



ACOS 6.0.7

SSL Insight (SSLi) Configuration Guide

March, 2025

© 2025 A10 Networks, Inc. All rights reserved.

Information in this document is subject to change without notice.

PATENT PROTECTION

A10 Networks, Inc. products are protected by patents in the U.S. and elsewhere. The following website is provided to satisfy the virtual patent marking provisions of various jurisdictions including the virtual patent marking provisions of the America Invents Act. A10 Networks, Inc. products, including all Thunder Series products, are protected by one or more of U.S. patents and patents pending listed at: [a10-virtual-patent-marking](#).

TRADEMARKS

A10 Networks, Inc. trademarks are listed at: [a10-trademarks](#)

CONFIDENTIALITY

This document contains confidential materials proprietary to A10 Networks, Inc. This document and information and ideas herein may not be disclosed, copied, reproduced or distributed to anyone outside A10 Networks, Inc. without prior written consent of A10 Networks, Inc.

DISCLAIMER

This document does not create any express or implied warranty about A10 Networks, Inc. or about its products or services, including but not limited to fitness for a particular use and non-infringement. A10 Networks, Inc. has made reasonable efforts to verify that the information contained herein is accurate, but A10 Networks, Inc. assumes no responsibility for its use. All information is provided "as-is." The product specifications and features described in this publication are based on the latest information available; however, specifications are subject to change without notice, and certain features may not be available upon initial product release. Contact A10 Networks, Inc. for current information regarding its products or services. A10 Networks, Inc. products and services are subject to A10 Networks, Inc. standard terms and conditions.

ENVIRONMENTAL CONSIDERATIONS

Some electronic components may possibly contain dangerous substances. For information on specific component types, please contact the manufacturer of that component. Always consult local authorities for regulations regarding proper disposal of electronic components in your area.

FURTHER INFORMATION

For additional information about A10 products, terms and conditions of delivery, and pricing, contact your nearest A10 Networks, Inc. location, which can be found by visiting www.a10networks.com.

Table of Contents

Getting Started	17
Overview	18
Architecture	18
Features	21
Limitations	21
Terminology	22
Real Server	22
Virtual Server and Virtual IP (VIP)	23
Service Groups	23
Wildcard VIPs, Ports, Virtual Ports, and ACL	24
Normal VIP with ACL	25
ACOS_decrypt and ACOS_encrypt Partition or Device	26
Topologies	27
Overview	28
SSLi in L2 Mode	28
SSLi in L3 Mode	30
Deployments	32
Single ACOS Device with One Partition Deployment	33
Features	34
Single ACOS Device with Two Partitions Deployment	36
Features	37
Two ACOS Devices, Each with One Partition Deployment	39
Features	39
Single Partition Deployment	42
Architecture	43
Deployment Types	45
L2 Deployment with Tagged VLANs	45

CLI Configuration	46
Step 1: Initial Configuration	46
Step 2: Configuring the Network VLANs	47
Step 3: Configuring the SSLi Services	48
Step 4: Configuring Network IP Addresses	50
Step 5: Configuring the Security Device	51
Step 6: Configuring Handling of Incoming Traffic	52
Step 7: Configuring Handling of Outgoing Traffic	53
Consolidated Configuration	55
GUI Configuration	60
Step 1: Configuring the Network VLANs	60
Step 2: Configuring the SSLi Services	61
Step 3: Configuring the VIPs	65
Step 4: Configuring the Security Device	66
Step 5: Configuring Handling of Incoming Traffic	67
Step 6: Configuring Handling of Outgoing Traffic	69
L2 Deployment with Untagged VLANs	71
CLI Configuration	72
Step 1: Initial Configuration	72
Step 2: Configuring the Default VLAN	73
Step 3: Configuring the SSLi services	74
Step 4: Configuring Network IP Addresses	76
Step 5: Configuring the Security Device	76
Step 6: Configuring Handling of Incoming Traffic	78
Step 7: Configuring Handling of Outgoing Traffic	79
Consolidated Configuration	81
Outbound Static Port Type HTTPS	86
Prerequisites	87
Two ACOS Devices, Each With Single Partition Deployment	87
CLI Configuration	90

ACOS_decrypt Configuration	90
ACOS_encrypt Configuration	95
GUI Configuration	99
ACOS_decrypt Configuration	99
ACOS_encrypt Configuration	106
Consolidated Configuration	108
Checking the Status and Operation	112
Single ACOS Device With Two Partitions Deployment	113
CLI Configuration	113
GUI Configuration	113
Single vThunder Device With Two Partitions Deployment	114
Outbound Static Port Type STARTTLS	116
Deployment Example	117
CLI Configuration	119
ACOS_decrypt Configuration	120
Step 1. Configuring the Network VLANs	120
Step 2. Configuring the Network IP Addresses	121
Step 3. Configuring the SSLi Services	121
Step 4. Configuring the SSLi Service Groups	124
Step 5. Configuring the Virtual Server	125
ACOS_encrypt Configuration	127
Step 1. Configuring the Network VLANs	128
Step 2. Configuring the Network IP Addresses	128
Step 3. Configuring the SSLi Services	128
Step 4. Configuring the SSLi Service Groups	130
Step 5. Configuring the Virtual Server	131
Consolidated Configuration Examples	134
Inbound Static-Port Type HTTPS	142
Deployment Example	143
CLI Configuration	144

Configure the External Inbound ACOS device	144
Configure the Internal Inbound ACOS device	149
Dynamic-Port Inspection	154
Configuration Workflow	154
ACOS_decrypt Configuration	155
ACOS_encrypt Configuration	156
CLI Configuration	156
Inside ACOS Configuration	156
ACOS_encrypt Configuration	159
ACOS_decrypt Configuration	161
Consolidated Configuration	164
DSCP Inspection	168
Deployment Example	168
Traffic WorkFlow	169
CLI Configuration	170
ACOS_decrypt Configuration	171
ACOS_encrypt Configuration	175
Consolidated Configuration	178
Static Port SSH Insight	183
Overview	184
Deployment	185
CLI Configuration	187
ACOS_decrypt Configuration	187
ACOS_encrypt Configuration	190
Certificate and Keys Management	197
SSL Certificate Management	198
CA Certificate Chaining	199
CA-Signed and Self-Signed Certificates	201
CA Certificate Versus SSL Certificate	202
Configuring Certificate Key Pair	203

SSL Handshake WorkFlow	203
Websites Workflow	207
Certificates in SSL Templates	208
Client-SSL Template Configuration and Usage Guidelines	209
Server-SSL Template Configuration and Usage Guidelines	212
Cipher Template Configuration and Usage Guidelines	214
Certificate Fetching and Forging	215
CLI Configuration	216
GUI Configuration	216
Certificate Pinning Candidate List	217
Configure Bypass Rule for Local Cert Pinning List	217
Central Cert Pinning List	219
TLS Server Name Indication (SNI) Support	223
Prerequisite	223
SNI Extension Support	224
SNI Support on vThunder	230
Configuring TLS SNI	231
Configuring Dynamic SNI Transmission for Inbound SSLi	232
TLS 1.3 Support for Software SSL	239
CAs and CSRs Management	240
Importing a Certificate and Key	240
Importing Individual Files	241
SSL Certificate and Key Files Bulk Import	242
Generating a Certificate Signing Request (CSR)	243
Generating a Self-Signed Certificate and Key	244
Generating an SSL Certificate – Private Key File with a CSR	246
Installing Certificates	250
Requesting and Installing a CA-Signed Certificate	250
Installing a Self-Signed Certificate	252
Implementing Certificates to SSL Templates	253
Creating Multiple CA Certificate in Server-SSL Templates	254

Multiple Certificates in Single File – Preparing the File	255
Binding Server-SSL Templates to Individual Real Ports	257
Server SSL Support for Data Plane Health Monitor	258
Configuring Email Notification for SSL Certificate Expiration	259
Converting Certificates and CRLs to PEM Format	260
Importing a Certificate Revocation List (CRL)	262
Exporting Certificates, Keys, and CRLs	263
Importing a CA Cert and Private Key	264
Configuring Forward Proxy Alternate Signing Cert	265
Deleting Certificate Files	266
Configuring Simple Certificate Enrollment Protocol (SCEP) Certificates	266
Key Considerations	267
Enrollment and Renewal Process	267
Configuring SCEP Certificate	268
Copying SCEP Certificate	269
Viewing SCEP Certificate	269
Configuration Example	269
Configuring Automatic Certificate Management Environment (ACME) Certificates	272
Key Considerations	273
ACME Directory URL	274
Enrollment and Renewal Process	274
Configuring ACME Certificate	276
Viewing ACME Certificate	277
OCSP Certificate Management	281
Overview	281
ACOS Server Certificate Verification	282
CLI Configuration	285
Certificate Revocation List	288
CLI Configuration	288
IP-less OCSP and CRL Requests	290
CLI Configuration	291

Invalid Certificates Customizable Message	292
CLI Configuration	292
Revoking Certificates	293
CLI Configuration	293
Explicit and Transparent Proxy	297
Overview	298
Deployment Example	298
CLI Configuration	299
ACOS_decrypt Configuration	300
ACOS_encrypt Configuration	302
Verifying the Configuration	302
Consolidated Configuration	304
Proxy Chaining SSLi	308
Configuration with Explicit Proxy	308
Configuration with Transparent Proxy	309
CLI Configuration	309
ACOS_decrypt configuration	309
ACOS_encrypt configuration	311
Explicit Proxy with Dual-Stack Forwarding Support	312
Deployment	313
Traffic Workflow	314
CLI Configuration	315
Drop and Drop-Redirect-URL Message Responses	321
ACOS_decrypt Configuration	322
Consolidated Configuration	324
Drop and Drop-Redirect-URL Priorities	326
Virtual Wire with SSLi Deployment	327
SSLi IP-Less Deployment	328
CLI Configuration	329
SSLi IP-Less Deployment with Explicit Proxy and Cert Validation	331

CLI Configuration	331
SSLi IP-Less Deployment with Layer 2 Health Monitoring	333
CLI Configuration	334
SSLi IP-Less Deployment with Multiple Ports for Failover	337
CLI Configuration	339
SSLi IP-Less Deployment with Layer 3 Health Monitoring	344
CLI Configuration	346
Limitation	353
AAM for SSLi Transparent Proxy	353
Topology Example	354
CLI Configuration	355
Decrypt_VIP Support	357
Forward Policy Support	358
HTTP Authenticate Logon Support	358
Transparent Proxy with Normal VIPs Associated to ACL	359
Overview	360
Key Considerations	360
Limitations	361
Virtual Wire Configuration	361
Virtual Wire Consolidated Configuration Example	364
Non-standard Port Support	366
CLI Configuration	366
GUI Configuration	367
Terminating SSLi Session Silently	367
Bypass, Inspect, and Exception	370
Overview	371
Priority of Rules	372
SNI and Server Certificate Based Inspection	375
Convert an SNI List to an AC Class List	376
User Name and Group Name Based Bypass	377

HTTPS Traffic Bypass	378
CLI Configuration	378
Creating a Class List	380
Importing a Class List	381
Showing the System Resource Usage	382
GUI Configuration	382
Creating a Class List	384
Importing a Class List	385
Client Authentication Bypass	386
WorkFlow	387
CLI Configuration	388
GUI Configuration	388
Consolidated Configuration	389
ACOS_decrypt	389
Outside ACOS device	391
Troubleshooting	392
IP Based Bypass	393
Configure Client IP Class List	394
CLI Configuration	394
GUI Configuration	395
Configure Server IP Class List	396
CLI Configuration	396
GUI Configuration	397
SSLi Bypass for "no shared cipher" Error	398
CLI Configuration	398
GUI Configuration for "no shared cipher" Error	399
Consolidated Configuration	399
"no-shared-cipher" Error	399
AAM, User Name, AD Group, Explicit Proxy, and SSLi	400
AAM, User Name, AD Group Name, Transparent Proxy, and SSLi	401

Web Category and Web Reputation Bypass	403
Web Category Bypass	404
Installing Web Category License	404
Step 1: Installing the Web Category License	404
Step 2: Verifying the Web Category License Installation	406
Step 3: Activating the Web Category License	406
Step 4: Verifying the Web Category Library	406
Step 5: Checking Web Category License Status and Expiration	407
Using a Proxy Server for BrightCloud Servers	407
Filtering Web Category for SSLi Bypass	408
Configuring SSLi Bypass Filtering	409
Viewing the Statistics	410
Deleting or Reimporting the Database	412
Troubleshooting	413
Logging	414
Implementing Lookup Enforcement	416
Implementing URL Filtering	416
Implementing SSLi Bypass	417
Web Reputation Bypass	417
Configuring the Forward Policy	418
Configuring the SSLi Bypass Filtering	419
Viewing the Statistics	420
URL Filtering	422
Overview	423
CLI Configuration	424
GUI Configuration	425
Consolidated Configuration	426
TLS Fingerprinting	430
JA3 Fingerprinting Overview	431
JA4+ Fingerprinting Overview	432

CLI Configuration	433
GUI Configuration	435
Show Commands	436
ICAP Services	437
Overview	438
Limitation	438
ICAP Template Options	438
Topology Example	439
CLI Configuration	441
Inside_SSLi	441
Outside_SSLi	443
GUI Configuration	443
Show Commands	445
ICAP Configuration Options	445
Including Protocol and Port in HTTP URI	446
CLI Configuration	447
Pre-Filtering ICAP Traffic	448
Disabling ICAP Templates	448
Configuring ACOS Logging	450
Log Example	451
Single Virtual Port Deployment	452
Key Considerations	452
Basic SSLi Deployment	453
CLI Configuration	454
Consolidated Configuration	456
Explicit Proxy Deployment	458
CLI Configuration	458
Consolidated Configuration	463
SSLi with IPv6 Deployment	465
Overview	466

Prerequisites	467
CLI Configuration	467
ACOS_decrypt Configuration	468
Step 1. Configuring the Network VLANs	469
Step 2. Configuring the Network IP Addresses	469
Step 3. Configuring the SSLi Services	470
Step 4. Configuring the SSLi Service Groups	470
Step 5. Configuring the Virtual Server	471
Consolidated Configuration for SSLi_Inside	473
ACOS_encrypt Configuration	476
Step 1. Configuring the Network VLANs	476
Step 2. Configuring the Network IP Addresses	476
Step 3. Configuring the SSLi Services	477
Step 4. Configuring the SSLi Service Groups	477
Step 5. Configuring the Virtual Server	478
Consolidated Configuration	479
GUI Configuration	482
SSLi in VRRP-A Deployment	483
Deployment Example	484
ACOS_decrypt CLI Configuration	485
Inside Primary ACOS device Configuration	485
Inside Secondary ACOS device Configuration	490
ACOS_encrypt CLI Configuration	495
Outside Primary ACOS device Configuration	495
Outside Secondary ACOS device Configuration	500
SSLi in Multi-PU Deployment	505
Supported Features	506
Key Considerations	506
Limitations	507
CLI Configuration	507

Basic SSLi Configuration	508
CLI Configuration	508
Explicit and Transparent Proxy Configuration	513
CLI Configuration	513
Certificate Fetching and Forging Configuration	518
CLI Configuration	518
Show Commands	519
Miscellaneous Features	530
File Inspection	531
CLI Configuration	531
Verifying the Device has a Cylance License	531
Creating a File Inspection Template	531
Binding the File Inspection Template to a Port	533
Importing a Cylance BW List	533
Implementing File Inspection on ADP	534
SSLi Source NAT	534
Static Source NAT CLI Configuration	534
Auto Source NAT CLI Configuration	537
Consolidated Configuration	540
Self-Signed Certificates	542
CLI Configuration	542
Show Configuration	543
Persistent Proxied Certificates	543
Creating a Persistent Forward-Proxy Class List	544
Binding a Separate Client-SSL Template	545
SSL CPU Offload	545
AutoUpdateO365 Script	547
Chrome Browser Configuration Options	553
Error Logging	554
Overview	555

CLI Configuration	557
Failure Event Error Reasons	558
Examples	584
CEF Error Log	584
Generic Failure Logs	584
SSLi Bypass Logs	585
SSL CA Verification Failure Logs	585
Example	585
Additional Failure Logs	586
Event-based Logging	587
Logging and Exporting SSL Session Secrets	587
Configuration Example	588
Limitation	589
Glossary	590

Getting Started

This section cover how to get started with SSL Insight (SSLi).

The following topics are covered:

Overview	18
Architecture	18
Features	21
Limitations	21
Terminology	22

Overview

Traditional security devices have the ability to inspect HTTP traffic, however, such devices cannot inspect SSL or encrypted traffic without incurring heavy CPU resources. This limited functionality of traditional security devices is a concern as the volume of encrypted traffic is increasing and is expected to surpass the volume of unencrypted traffic. Considering the immense possibility of cyber threats propagating through encrypted traffic, it is essential that organizations configure their security devices to inspect both encrypted and unencrypted traffic.

Deploy SSLi in your organization to dedicatedly decrypt SSL traffic, which can then be analyzed by a security device. Since the encryption and decryption functions are performed by the SSLi device, there is minimum latency in the network.

SSLi is configurable by using any of the supported ACOS devices. SSLi can detect and decrypt encryption on even non-proprietary TCP protocols. SSLi is deployable in a number of different ways, customizable for your network environment, with added HA. SSLi is also scalable to address the requirements of an expanding organization. The integrated load balancing capability of SSLi helps to optimize the SSLi performance.

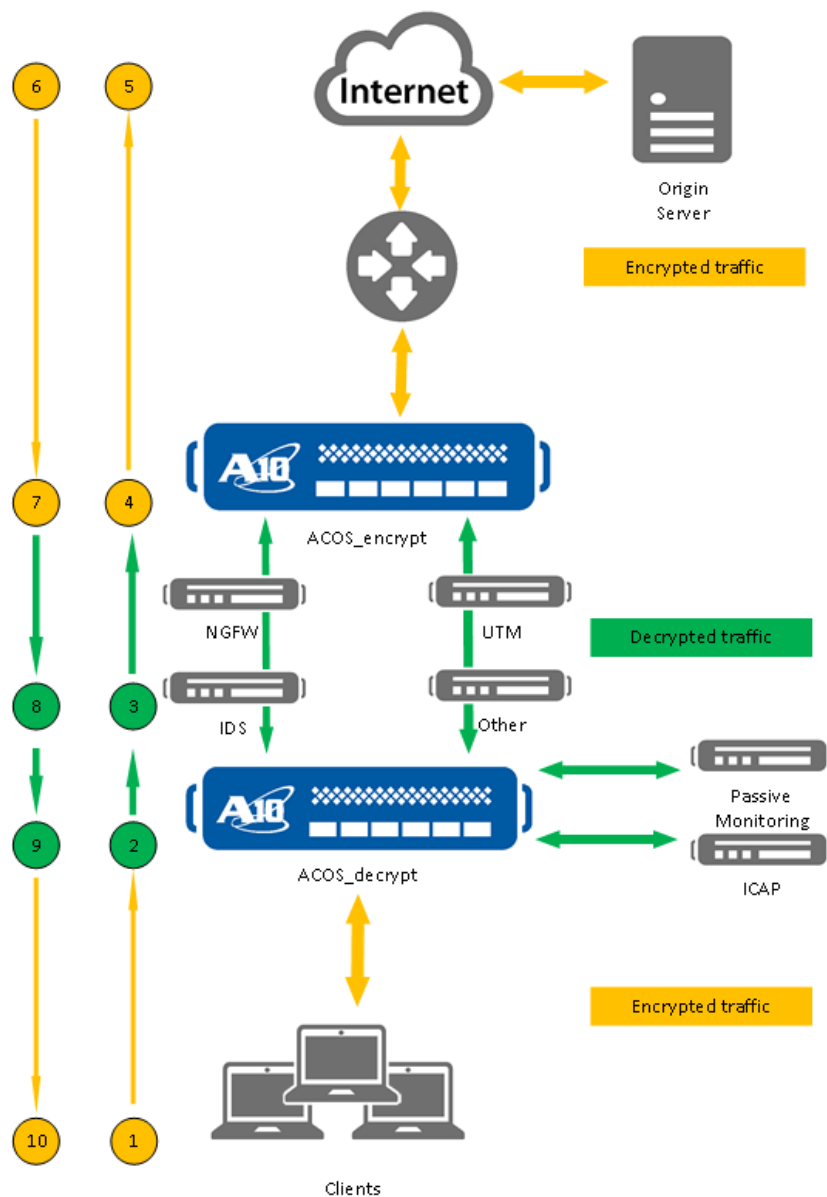
For more information on the supported ACOS devices for deploying SSLi, refer to the [SSLi Technical Specifications document](#).

Architecture

In the following deployment example, the client network is connected to the SSLi solution which is then connected through a gateway to the external network such as the Internet. All the encrypted traffic between the Internet and the client network is passed through the SSLi solution for inspection.

In the following deployment example, the client network is connected to the SSLi solution which is then connected through a gateway to the external network such as the Internet. All the encrypted traffic between the Internet and the client network is passed through the SSLi solution for inspection.

Figure 1 : SSLi Architecture



Deploy the SSLi solution in a number of ways by using one or more supported ACOS devices, reducing the disruption to your existing network to a minimum. In this example, the SSLi solution consists of two ACOS devices and a number of sample security devices that perform the traffic inspection on the clear decrypted text. Some examples of sample security devices are a next-generation firewall (NGFW), an intrusion detection system (IDS), a unified threat management (UTM), and so on. The

ACOS devices can also be configured as an ICAP client to offload traffic inspection to an ICAP server.

NOTE: While configuring SSLi, it is recommended to have separate interfaces for management and data in your network, as the management network frequently uses SSL.

You can deploy the SSLi solution with a single ACOS device or multiple ACOS devices. The ACOS devices in the SSLi solution consists of two parts:

- **ACOS_decrypt**—The ACOS partition or ACOS device(s) that connects to the client network. This part of the SSLi solution decrypts the traffic from the client and passes the clear traffic to the security devices for inspection. In some implementations, this part is also referred to as ACOS_inside.
- **ACOS_encrypt**—The ACOS partition or ACOS device(s) that connects to the server network. This part of the SSLi solution re-encrypts the clear traffic which it receives from the security device and passes it to the external server network by using SLB operations. In some implementations, this part is also referred to as ACOS_outside.

The following is an explanation of the workflow of the SSLi solution:

- The client network sends an encrypted request to a remote server.
- After a session is established, the traffic is intercepted and decrypted by the SSLi solution (ACOS_decrypt). Clear-text traffic is sent to the security devices.
- The security device inspects the clear-text request data and, if approved, forwards it to the SSLi solution to be re-encrypted (ACOS_encrypt).
- The traffic is intercepted by ACOS_encrypt, re-encrypted, and sent to the default gateway.
- The remote server receives an encrypted request.
- The remote server sends back an encrypted response.
- The SSLi solution (ACOS_encrypt) decrypts the response and forwards it to the same security device that sourced the request.
- The security device inspects the clear-text response data and, if approved, forwards it to the SSLi solution to be re-encrypted (ACOS_decrypt).

- The traffic is intercepted by ACOS_decrypt, encrypted again, and sent to the client.
- The client receives the encrypted response.

Features

As discussed previously, the SSLi solution is a requirement of organizations to decrypt traffic for analyzing the security devices data. SSLi has a number of advantages and high performance ratio as compared to other available similar products.

Here are few advantages available when deploying the SSLi:

- Transparent or Explicit proxy deployment in the network
- URL classification services for meeting compliance standards
- Dynamic port inspection of SSL and TLS traffic
- ICAP client to an ICAP server for DLP and AV security devices
- Utilize extensive SSL cipher including support for ECDHE and DHE
- Load balancing capabilities to support scaling of the security infrastructure
- End-to-end HTTP/2 support to enable full request and response multiplexing
- SSLi support for Intel QuickAssist Technology (QAT) and Nitrox V (N5) Hardware Acceleration modules
- SSLi intercepts SMTP, POP, FTP, LDAP, and XMPP sessions that are running over SSL

Limitations

SSLi has the following limitations.

- ACOS device cannot pass packets when the device has a failure or is powered down. To configure this functionality, a second ACOS device or a bypass switch is required.
- HTTP/2 support is available for static-port SSLi with and without forward-proxy, and dynamic-port SSLi. It is not supported for STARTTLS SSLi.

- Use of a native VLAN with tagged VLANs is not supported
- Sites which use hard certificate pinning cannot be decrypted

Terminology

Before deploying SSLi, there are some terms provided in the following sections to help you understand how SSLi functions. For more information, see *Application Delivery and Server Load Balancing Guide*.

The following topics are covered:

Real Server	22
Virtual Server and Virtual IP (VIP)	23
Service Groups	23
Wildcard VIPs, Ports, Virtual Ports, and ACL	24
Normal VIP with ACL	25
ACOS_decrypt and ACOS_encrypt Partition or Device	26

Real Server

A real server is the logical representation of physical servers (either individual servers, or servers in a server farm) connected to an ACOS device, or to another router in the network. To configure a real server, a name, an IP address, and a port are required.

In SSLi operation, the security device or collection of security devices is configured as a real server.

The following is an example of configuring a security device in an SSLi solution as a real server:

```
ACOS_decrypt(config)# slb server GW 1.1.1.254
ACOS_decrypt(config-slb server)# port 0 tcp
```

Virtual Server and Virtual IP (VIP)

A virtual server is the combination of real servers and an ACOS device(s), which together appear as a single server to the client.

A virtual IP (VIP) is the IP address of the virtual server. The VIP is used to access a group of servers or it can be a default gateway for users accessing the Internet. To configure a virtual server, a name, an IP address, and a port are required.

In SSLi operation, the security device or collection of security devices together with the ACOS device or devices is configured as a virtual server. The virtual server port or port 0 is configured for a virtual server with the `no-destination-nat` option enabled. This configuration enables SSLi to accept traffic for any destination port and send it to any destination port.

The following is an example of configuring a virtual server for incoming traffic:

```
ACOS_decrypt(config)# slb virtual-server decrypt_VIP 0.0.0.0 acl 100
ACOS_decrypt(config-slb vserver)# port 443 https
ACOS_decrypt(config-slb vserver)# port 0 tcp
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat port-translation
```

If the `port-translation` option is used, and the response traffic passes through the ACOS device, the ACOS device translates the source port of the server-reply back into the destination port to which the client sent the request, before forwarding the reply to the client. The `port-translation` option is supported only for the following virtual port types: TCP, UDP, and HTTP/HTTPS.

Service Groups

A service group is a group of servers that fulfil a service. Service groups are where load balancing algorithms are applied. The minimum configuration for a service group includes a name, the type of protocol, the load balancing algorithm, and at least one real server and port.

In SSLi operations, configure service groups to handle different types of encrypted traffic that is intercepted by the SSLi solution. In the following configuration example, a real server `FW1_Inspect` is created on `ACOS_decrypt`. A service group named `FW1_Inspect_sg` is also created on `ACOS_decrypt` to forward decrypted traffic over protocol TCP on port 8080.

In the following configuration example, a real server **FW1_Inspect** is created and added to the also created service group **FW1_Inspect_SG**. All the traffic will be decrypted and forwarded to members of the group (in this case) over protocol TCP on port 8080.

```
ACOS_decrypt(config)# slb server FW1_Inspect 10.15.1.12
ACOS_decrypt(config-real server)# port 8080 tcp
ACOS_decrypt(config-real server)# exit
ACOS_decrypt(config)# slb service-group FW1_Inspect_SG tcp
ACOS_decrypt(config-slb svc group)# member FW1_Inspect 8080
```

Wildcard VIPs, Ports, Virtual Ports, and ACL

A wildcard VIP is a VIP that does not have a specific IP address. Instead, wildcard VIPs have IP address 0.0.0.0 (for IPv4) or :: (for IPv6). The client requests sent to any IP address are accepted when they are received at a wildcard VIP.

Wildcard VIPs enable you to configure a feature that applies to multiple VIPs, without the need to reconfigure the feature separately for each VIP. To specify the subset of VIP addresses and ports for which a feature is applicable, use an Access Control List (ACL). ACLs also specify the subset of clients allowed to access the VIPs, thus ensuring that only legitimate requests are allowed through. Wildcard VIPs can be used for any type of load balancing. Port 0 is used as a wildcard port to match any port number.

In SSLi operations, a wildcard VIP is configured to intercept supported encrypted traffic such as HTTPS, STARTTLS, IMAPS, SSH, and so on, on any port. Use ACLs to specify the clients whose traffic is to be intercepted. The virtual server port or port 0 is configured for a virtual server with the **no-destination-nat** option enabled. This configuration enables SSLi to accept traffic for any destination port and send it to any destination port.

The following is an example configuration for a wildcard VIP that accepts HTTPS requests on port 443:

```
ACOS_decrypt(config)# slb virtual-server decrypt_VIP 0.0.0.0 acl 100
ACOS_decrypt(config-slb vserver)# port 443 https
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat port-translation
```


The following is an example configuration where on VLAN 10, all IP traffic is intercepted by ACOS_decrypt by using an ACL 100:

```
ACOS_decrypt(config)# access-list 100 permit ip any any vlan 10
ACOS_decrypt(config)# slb virtual-server ACOS_decrypt 0.0.0.0 acl 100
```

The maximum wildcard VIP limit is 200 for each partition. However, the number of wildcard VIPs per device is based on the system memory size. Also, the IPv4 and IPv6 VIPs are calculated separately. For example, with 64 GB of memory, you can configure 1200 IPv4 VIP and 1200 IPv6 VIP. For more information, see [Table 1](#) and [Table 1](#).

Table 1 : Wildcard VIPs Limit on vThunder

Memory (GB RAM)	Maximum Wildcard VIP ACL
1 - 16	200
32	800
64	1200

Table 2 : Wildcard VIPs Limit on Hardware Platform

Memory (GB RAM)	Maximum Wildcard VIP
1 - 24	200
32	800
64 - 128	1200

Normal VIP with ACL

A normal VIP with ACL functions like a [Wildcard VIPs](#) but has a specific IP address that is associated to an ACL. For example, 192.20.10.1 acl name myacl1. In this example, the VIP serves the same purpose as the destination IP in the ACL and is used for traffic matching. The source IP defined in the associated ACL is used to determine if the traffic matches the defined virtual port and accordingly permits or denies the traffic.

You can configure multiple VIPs associated with ACLs in ACOS alongside wildcard VIPs associated with ACLs. The maximum configurable number depends on the platform and system memory size. The maximum configurable number of normal VIPs is

greater than the maximum configurable number of wildcard VIPs. If traffic does not match any virtual port in a VIP with ACL, it uses the wildcard VIP for matching.

For configuration details, see [Transparent Proxy with Normal VIPs Associated to ACL](#).

ACOS_decrypt and ACOS_encrypt Partition or Device

The SSLi solution sandwiches the security device or devices between the ACOS_decrypt and ACOS_encrypt partition or device. ACOS_decrypt and ACOS_encrypt can be configured on separate ACOS devices or in a single ACOS device by using partitions.

There are also examples of single partition SSLi deployments where ACOS_decrypt and ACOS_encrypt zones are created by using a combination of virtual servers and ACLs. In a single partition deployment, a VIP represents the client and server sides.

Some guidelines for configuring ACOS_decrypt and ACOS_encrypt are as follows:

- Provision ACOS_decrypt with either a CA or a subordinate CA certificate and the accompanying private key. Refer to [CA Certificate Chaining](#).
- With HTTPS to HTTP conversion, the destination port is changed from 443 to any other port such as 8080.
- Create a client-SSLi template with **forward-proxy-enable** configured.
- Any TCP or UDP traffic that is intercepted must have an access control list (ACL) configured within the wildcard VIP to define the traffic flow.
- Incoming HTTPS sessions that are intercepted and decrypted are forwarded as clear text over HTTP on a configurable port such as 8080 through a third-party security device.
- ACOS_decrypt decrypts all SSL traffic originating from the client. All clear-text traffic decrypted by ACOS_decrypt is passed to the security device.
- ACOS_encrypt zone re-encrypts the HTTP traffic received on the port such as 8080 from the security device after inspection. The clear-text traffic is encrypted to HTTPS 443 and sent to the default router or Internet by using the port 443. You must configure a server-SSLi template with **forward-proxy-enable** for this zone.

Topologies

This section provides an overview of the different types topologies for SSLi.

The following topics are covered:

Overview	28
SSLi in L2 Mode	28
SSLi in L3 Mode	30

Overview

SSLi can be deployed in different topologies. Topologies can differ based on the mode of the SSLi deployment. The security device can be either in-line or in a passive mode.

For in-line deployment of the security device(s), the following topological combinations are supported:

- SSLi in L2 mode and the in-line security device in L2 mode
- SSLi in L2 mode and the in-line security device in L3 mode
- SSLi in L3 mode and the in-line security device in L2 mode
- SSLi in L3 model and the in-line security device in L3 mode

Security devices can be deployed in passive (tap) mode by using a mirror port on the SSLi device. This deployment is independent of whether the security device or the SSLi device is in L2 or L3 mode. In this mode, the physical link is established between ACOS_decrypt and ACOS_encrypt appliances and the decrypted traffic is mirrored out to the passive security device. The tap mode supports up to eight security devices. Support for RST from the security device (over a separate link) to terminate compromised connections is also included.

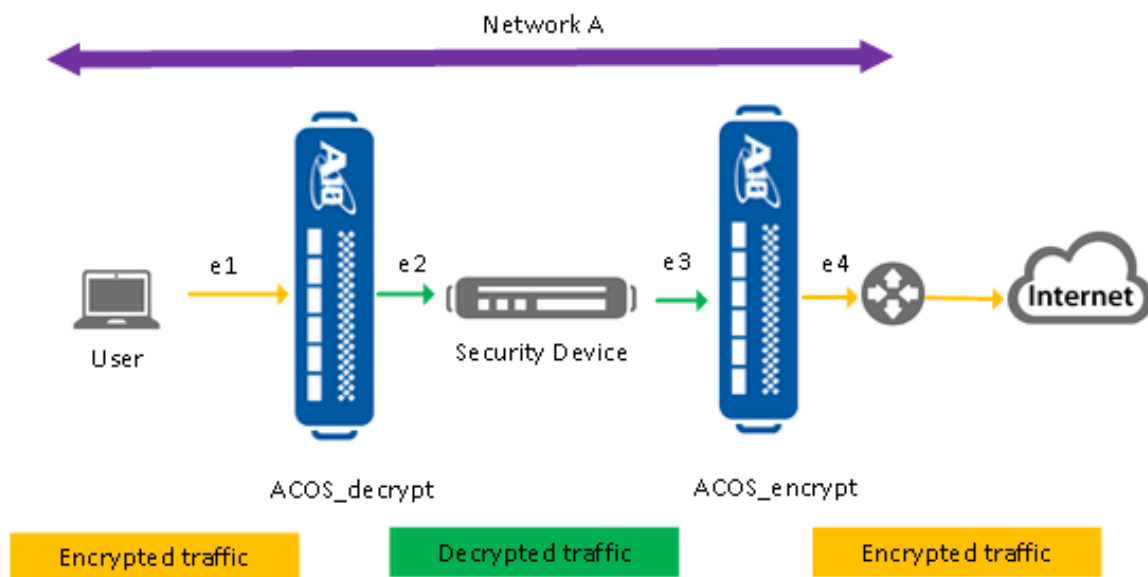
If you are configuring SSLi on a single vThunder device, then only two bi-directional or four unidirectional ports are required. For configuring SSLi on two vThunder devices, four bi-directional ports or 8 unidirectional ports are required.

SSLi in L2 Mode

In this topology, the SSLi solution consist of the ACOS device(s) in L2 mode and the security device(s) in L2 mode or L3 mode and these devices sit between the client and the external gateway. All of the devices are in the same subnet. For a single security device, four physical interfaces are required on the ACOS device, as shown in [Figure 2](#).

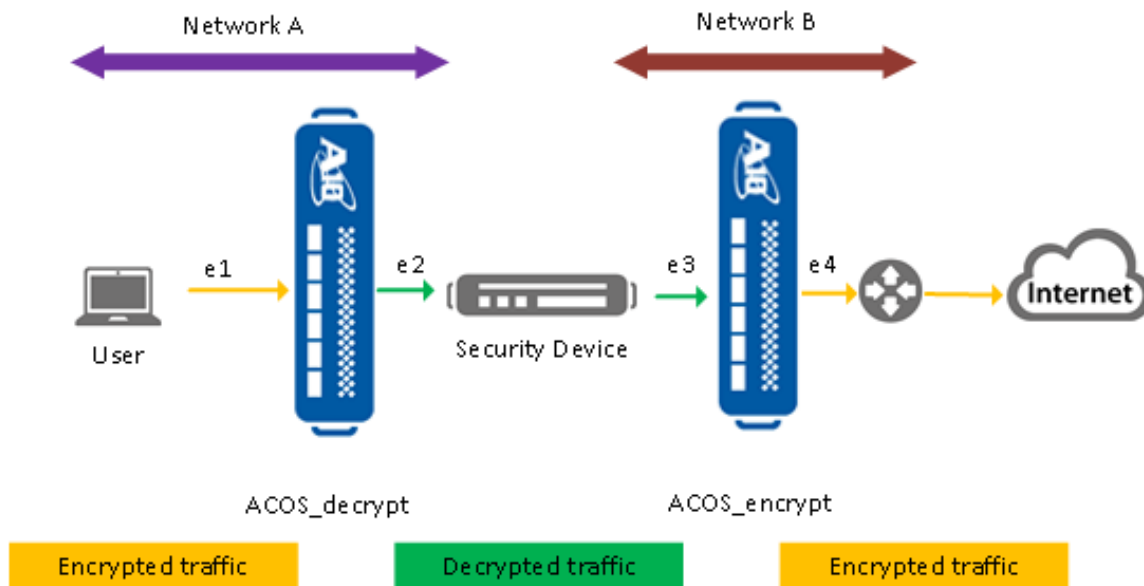
NOTE: On Thunder platforms with the older version of the FTA chipset, a `cpu-process` command must be run for the L2 mode to work. For more information, see “Configuring L2 SSLi on FTA-enabled ACOS Devices” on page 37.

Figure 2 : SSLi Deployment in L2 Mode, Security Device in L2 Mode



In this topology, there is minimal change to the existing IP network. Each additional security device requires two more physical interfaces on the ACOS device. Each additional security device must be in a separate subnet for load balancing purposes. If the security device is in L3 mode, two separate subnets are required.

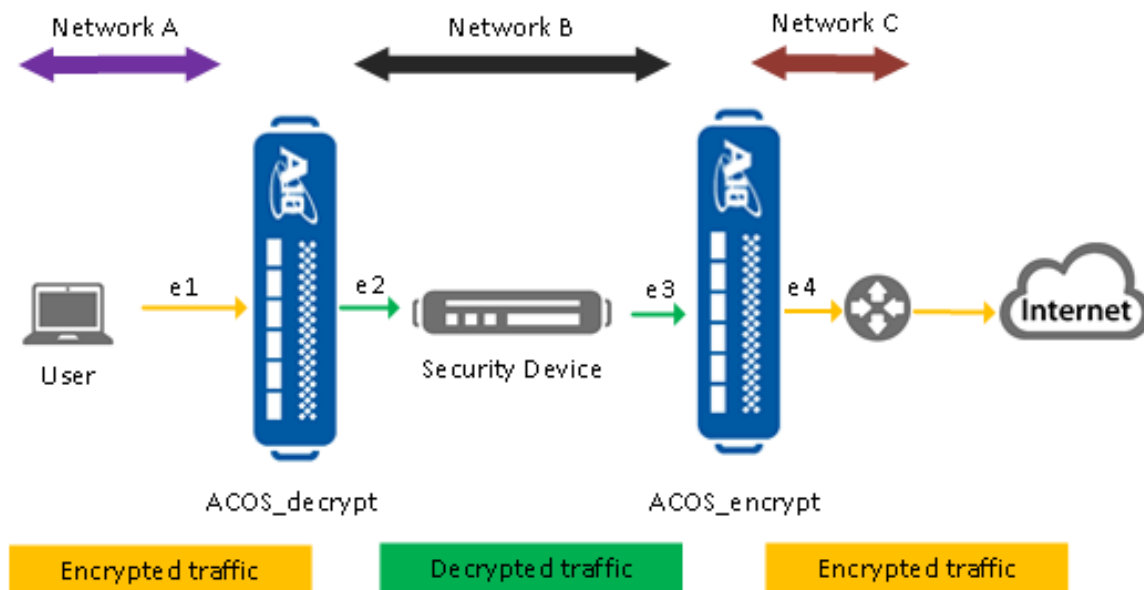
Figure 3 : SSLi Deployment in L2 Mode, Security Device in L3 Mode



SSLi in L3 Mode

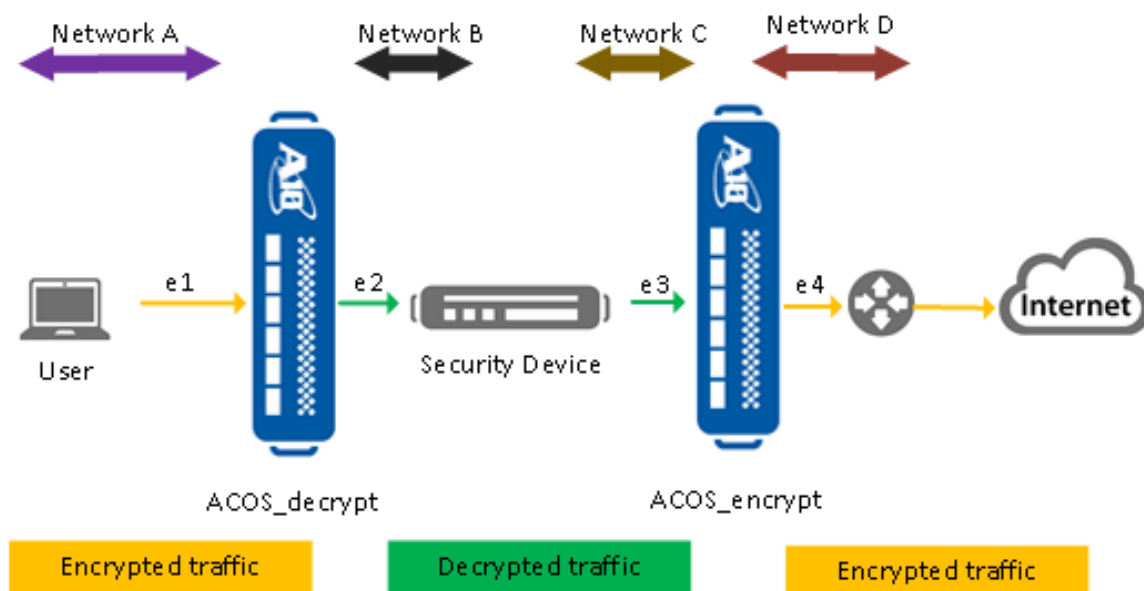
This topology configures the SSLi solution as a routed hop between the client network and the external gateway, which are on different subnets. The security device can either be deployed in an L2 or L3 mode. For a single security device, four physical interfaces are required on the ACOS device. Separate IP addresses are required for each interface. With a single security device in L2 mode, this topology requires three subnets, as shown in [Figure 4](#)

Figure 4 : SSLi Deployment in L3 Mode, Security Device in L2 Mode



For each additional security device, two more physical interfaces are required on the ACOS device. Each additional security device must be in a separate subnet for load balancing purposes. With a single security device in L3 mode, this topology requires four subnets.

Figure 5 : SSLi Deployment in L3 Mode, Security Device in L3 Mode



Deployments

This section provides an overview of the different types of deployments SSLi.

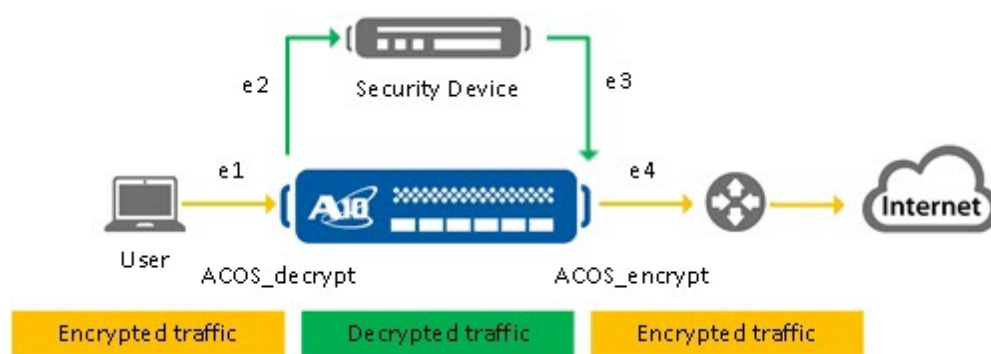
The following topics are covered:

Single ACOS Device with One Partition Deployment	33
Single ACOS Device with Two Partitions Deployment	36
Two ACOS Devices, Each with One Partition Deployment	39

Single ACOS Device with One Partition Deployment

In this deployment, a single ACOS device with one partition is configured as part of the SSLi solution. In a single partition deployment, the ACOS device is in L2 mode and requires one IP address at the minimum irrespective of the number of VLANs to be inspected. All interfaces used for the SSLi deployment must be assigned the same VLANs.

Figure 6 : Deployment of a Single ACOS Device with One Partition



In the sample deployment as shown in [Figure 6](#), the client device is connected to the SSLi solution, which is then connected to the external gateway. The SSLi solution consists of an ACOS device in L2 mode and a single security device in L2 mode. The encrypted traffic from the client is passed to the ACOS device on interface e1. The ACOS device decrypts the traffic and forwards the clear traffic to the security device on interface e2. After inspection, the security device passes the clear traffic to the ACOS device on interface e3. The ACOS device re-encrypts the traffic and passes it to the external gateway on interface e4.

Features

The following table lists the features for a single ACOS device with one partition deployment.

Table 4 :

Table 3 : Features for Single ACOS Device with One Partition

Features	Description	Notes
General Features	- Supported across all ACOS releases - SSLi Solution delivered in a single device - Web-category license add-on for the same device	- L3 firewalls supported across all ACOS releases. - L2 firewalls supported from ACOS 4.1.1-P3 version onwards. - Number of physical ports available to the solution is roughly halved.
SSLi Features	Static port inspection: - SNI-based bypass - Web category-based bypass - URL Filtering - Explicit proxy - Proxy chaining - ICAP Dynamic port inspection: - STARTTLS inspection	- Firewall Load Balancing (FWLB) is not supported. - URL filtering, explicit proxy, and proxy chaining are available with L3 firewall only.
Security Devices	- Inline L2 or vWire transparent firewalls - Inline L3 or NAT'ed transparent firewalls - Inline L7 or transparent proxy	- For inline L2 and L3 security devices, both tagged and untagged VLANs are supported. - For inline L7 security devices, only transparent proxy is supported. - One-armed transparent proxy is

Table 4 :

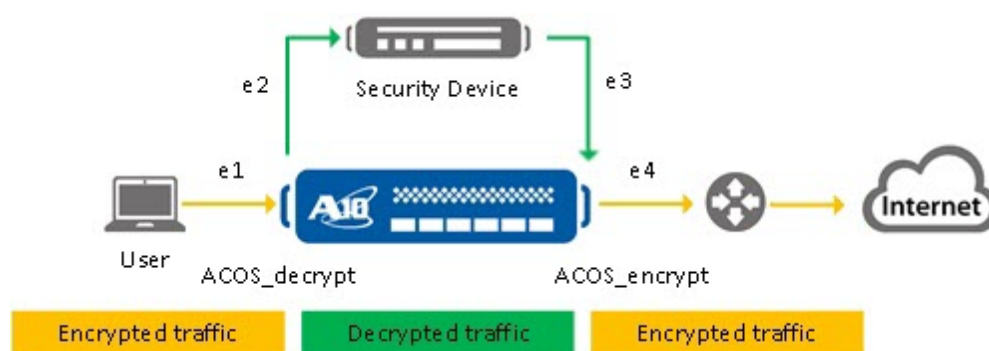
Table 3 : Features for Single ACOS Device with One Partition

Features	Description	Notes
	- One-armed transparent proxy - Non-inline passive IDS - ICAP-based DLP/AV	- supported with L3 firewalls only. - For non-inline passive IDs, up to four passive devices are supported.
Topologies	- Full L2 with the deployment behind SSLi and STP-based active-standby HA - L2 with L3 security device and VRRP-A based active-standby HA - L3 with A10 Thunder SSLi and VRRP-A based active-standby HA - Explicit proxy with A10 Thunder SSLi as the explicit proxy for client web browsers - Explicit proxy with upstream explicit proxy set on client web browsers	- For an L2 deployment, both tagged and untagged VLANs are supported. - L2 deployment does not support VRRP-A. - L3 deployment and both types of explicit proxy deployments are supported with L3 firewalls only. - Explicit proxy with upstream explicit proxy set on client web browsers require two IP addresses from the network.

Single ACOS Device with Two Partitions Deployment

In this deployment, two L3V partitions are configured in the ACOS device. The partition ACOS_decrypt is connected to the client and the partition ACOS_encrypt is connected to the external network by using a gateway. Configure `system ve-mac-scheme system-mac` on the shared partition to eliminate the MAC address duplication across partitions. If the ACOS device is a vThunder, also configure `system promiscuous-mode` on the shared partition.

Figure 7 : Deployment of a Single ACOS Device with a Two-Partition SSLi Solution



In the sample deployment as shown in [Figure 7](#), the client device is connected to the SSLi solution, which is then connected to the external gateway. The SSLi solution consists of an ACOS device and a single security device. The ACOS device has two partitions, ACOS_decrypt is connected to the client network and ACOS_encrypt is connected to the server network. The encrypted traffic from the client is passed to the ACOS_decrypt partition on interface e1. The ACOS_decrypt partition decrypts the traffic and forwards the clear traffic to the security device on interface e2. After inspection, the security device passes the clear traffic to the ACOS_encrypt partition on interface e3. The ACOS_encrypt partition re-encrypts the traffic and passes it to the external gateway on interface e4.

Features

The following table lists the features for a single ACOS device with two partitions deployment.

Table 6 :

Table 5 : Features for Single ACOS Device with Two Partitions

Features	Description	Notes
General Features	- Supported across all ACOS releases - SSLi solution delivered in a single ACOS device - Web-category license add-on for the same device - Full separation of L2 and L3 in ADPs - Firewall Load Balancing (FWLB) support	Number of physical ports available to the solution is roughly halved.
SSLi Features	Static port inspection: - SNI-based bypass - Web category-based bypass - URL Filtering - Explicit proxy - Proxy chaining - ICAP Dynamic port inspection: - STARTTLS inspection	For dynamic port inspection, a special header 'A10FP' gets pre-pended to client requests and is visible to the security device.
Security Devices	Inline untagged L2 or vWire transparent	For inline L2 deployment, only untagged VLANs are supported.

Table 6 :

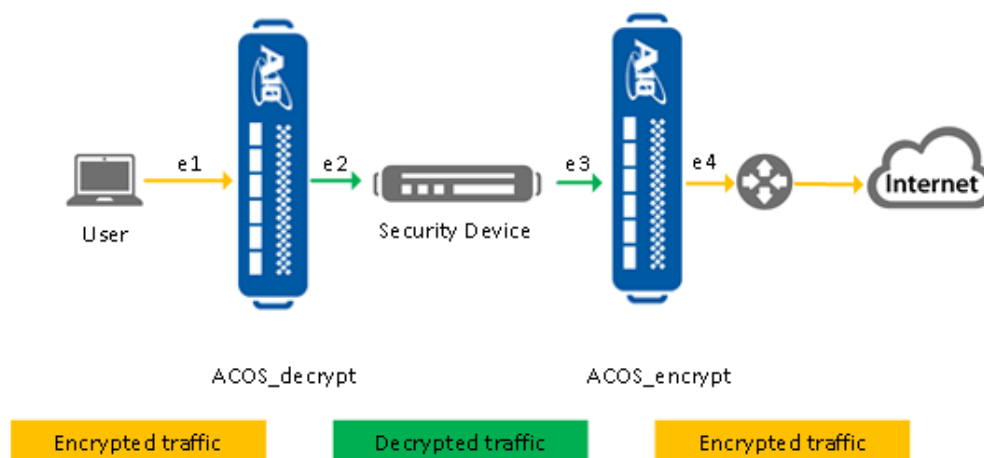
Table 5 : Features for Single ACOS Device with Two Partitions

Features	Description	Notes
	firewalls • Inline L3 or NAT'ed transparent firewalls • Inline L7 or transparent proxy • One-armed transparent proxy • Non-inline passive IDS • ICAP-based DLP/AV	• For inline L3, both tagged and untagged VLANs are supported. • For inline L7, only transparent proxy is supported. • For non-inline passive IDs, up to two passive devices are supported.
Topologies	• Full L2 with the deployment behind SSLi and STP-based active-standby HA • L2 with L3 security device as the deployment and VRRP-A based active-standby HA • L3 with A10 Thunder SSLi as the deployment and VRRP-A based active-standby HA • Explicit proxy with Thunder SSLi as the explicit proxy for client web browsers • Explicit proxy with upstream explicit proxy set on client web browsers	• For a full L2 deployment, only untagged VLANs are supported. VRRP-A is not supported. • For explicit proxy, two IP addresses are required from the network segment in which the Thunder SSi is deployed.

Two ACOS Devices, Each with One Partition Deployment

In this deployment, a dedicated ACOS device is configured each for the ACOS_decrypt and ACOS_encrypt partitions. This deployment provides a greater throughput than a single device deployment.

Figure 8 : Deployment of a Double ACOS Device SSLi Solution



In the sample deployment as shown in [Figure 8](#), the client device is connected to the SSLi solution, which is then connected to the external gateway. The SSLi solution consists of two ACOS devices and a single security device. The ACOS device connected to the client has a partition called ACOS_decrypt. The ACOS device connected to the external gateway has a partition called ACOS_encrypt. The encrypted traffic from the client is passed to the ACOS_decrypt partition on interface e1. The ACOS_decrypt partition decrypts the traffic and forwards the clear traffic to the security device on interface e2. After inspection, the security device passes the clear traffic to the ACOS_encrypt partition on interface e3. The ACOS_encrypt partition re-encrypts the traffic and passes it to the external gateway on interface e4.

Features

The following table lists the features for two ACOS devices, each with one partition deployment.

Table 8 :

Table 7 : Features for Two ACOS Devices, Each With One Partition

Features	Description	Notes
General Features	- Supported across all ACOS releases - Throughput is about 1.8x more than that of a single-device deployment - SSLi Solution is delivered with two ACOS devices - Web-category license add-on only for one device - Full separation of L2/L3 in two physical devices - Firewall Load Balancing (FWLB) support	Number of physical ports available to the solution is roughly doubled.
SSLi Features	Static Port inspection: - SNI-based bypass - Web category-based bypass - URL Filtering - Explicit proxy - Proxy chaining - ICAP Dynamic port inspection: - STARTTLS inspection	For dynamic port inspection, a special header 'A10FP' gets pre-pended to client request and is visible to the security device.
Security Devices	- Inline L2 or vWire transparent firewalls - Inline L3 or NAT'ed	- For inline L2 and L3, both tagged and untagged VLANs are supported. - For inline L7, only transparent proxy

Table 8 :

Table 7 : Features for Two ACOS Devices, Each With One Partition

Features	Description	Notes
	transparent firewalls • Inline L7 or transparent proxy • One-armed transparent proxy • Non-inline passive IDS • ICAP-based DLP/AV	is supported. • For non-inline passive IDs, up to four passive devices are supported.
Topologies	• Full L2 with the deployment behind SSLi and STP-based active-standby HA • L2 with L3 security device and VRRP-A based active-standby HA • L3 with A10 Thunder SSLi as the deployment and VRRP-A based active-standby HA • Explicit proxy with A10 Thunder SSLi as the explicit proxy for client web browsers • Explicit proxy with upstream explicit proxy set on client web browsers	• For a full L2 deployment, only untagged VLANs are supported. VRRP-A is not supported. • For explicit proxy, two IP addresses are required from the network segment in which the Thunder SSi is deployed.

Single Partition Deployment

This section describes the how to deploy SSLi by using a single partition instead of two partitions. The single partition approach allows for a bump-in-the-wire deployment that requires minimal changes to the existing network infrastructure.

The following topics are covered:

- [Architecture](#) 43
- [Deployment Types](#) 45
- [L2 Deployment with Tagged VLANs](#) 45
- [L2 Deployment with Untagged VLANs](#) 71

Architecture

In a single partition deployment, the ACOS device is in L2 mode and requires one IP address irrespective of the number of VLANs to be inspected. The VLAN ID and the source and destination MAC addresses of the incoming packets are completely preserved as the traffic passes through the ACOS device. For this type of deployment, all the four interfaces, e1, e2, e3, and e4 (as shown in [A Single Partition Deployment for SSLi](#)), related to the SSLi deployment must be assigned the same set of VLANs.

In the following example deployment, as shown in [Figure 9](#), the client network is connected through a layer 3 switch to the ACOS device. The ACOS device, which has a single partition, is in turn connected to a security device for traffic inspection purposes. The ACOS device is then connected through a layer 3 switch to the Internet.

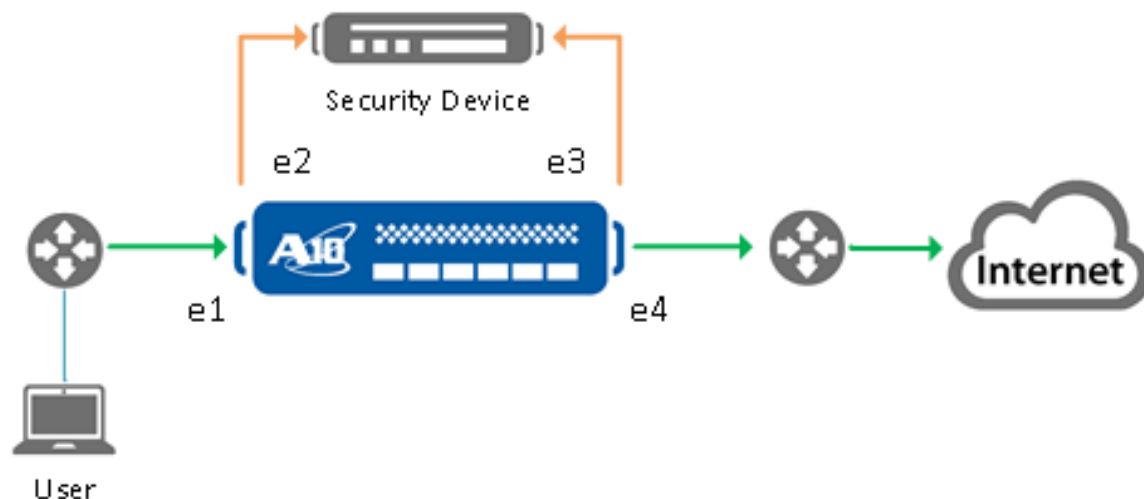
The traffic flows for the single partition deployment is described in the following section:

- **Traffic flows from the client network to the Internet**—The traffic flow from the client network is sent to the ACOS device on the e1 interface. The traffic flow is decrypted by the ACOS device. The traffic from the ACOS device is redirected to the security device in the forward direction. The traffic flow is forwarded from e1 to e2 by using the `redirect-fwd` command. From the security device, the traffic is directed back to the ACOS device on the e3 interface. The ACOS device re-encrypts the traffic and forwards the traffic to the gateway by using normal SLB operation.
- **Traffic flows from the Internet to the client network**—The traffic from the gateway is sent to the ACOS device on the e4 interface. The traffic flow is decrypted by the ACOS device. The traffic flow is then directed from e4 to e3 by using the `redirect-rev` command. From the security device, the traffic flow is directed back to the ACOS device on the e2 interface. The ACOS device re-encrypts the traffic and forwards the traffic to the client network on the e1 interface.

The security device is a L2 transparent device that preserves the L2 header while processing the traffic flows. For both scenarios, the L2 header is also preserved for the following traffic flows:

- Traffic flows between the client and the security device, on interfaces (e1 <- -> e2).
- Traffic flows between the security device and the gateway (e3 <- -> e4).

Figure 9 : A Single Partition Deployment for SSLi



The single partition SSLi deployment requires the ACOS device to have four interfaces. The functions of the interfaces is explained in the following list by using the logic of the traffic flow from the client network to the Internet:

- **e1**—This interface connects the layer 3 switch and the ACOS device. Traffic from the user network is channeled through the layer 3 switch to the ACOS device by using e1. An ACL rule is applied at e1 to forward only relevant traffic that is required to be inspected.
- **e2**—This interface connects the ACOS device and the security device. Decrypted traffic from the ACOS device is forwarded to the security device by using e2.
- **e3**—This interface connects the ACOS device and the security device. The inspected traffic from the security device is forwarded to the ACOS device by using e3. An ACL rule is applied at e3 to forward only relevant traffic.
- **e4**—This interface connects the ACOS device to another layer 3 switch. The inspected traffic from the user network is forwarded to the Internet by using e4.

The `redirect-fwd` and `redirect-rev` commands disable MAC learning on the interfaces specified in these commands and instead forwards packets to the specified ethernet port. The `redirect-fwd` configuration command redirects the client traffic to the security device. The `redirect-rev` configuration command redirects server

traffic back to the security device. See the port command in the *Command Line Interface Reference*.

NOTE: To ensure that all traffic is routed to the security device for inspection, you must define the traffic flow with respect to port-0-tcp, port-0-udp, and port-0-others as shown in the following configuration examples. Undefined traffic flows bypass the security device. Instead, configure SSLi Bypass to govern traffic that is not required to be inspected. See [Bypass, Inspect, and Exception](#).

Deployment Types

In single partition deployment, two types are supported and described in subsequent sections:

- L2 deployment with tagged VLANs
- L2 deployment with untagged VLANs

Tagged ports can be members of multiple VLANs. The port can recognize the VLAN to which a packet belongs based on the VLAN tag included in the packet. In the deployment scenario involving tagged VLANs, you can specify multiple VLANs for traffic inspection. All the ports of the security device are tagged.

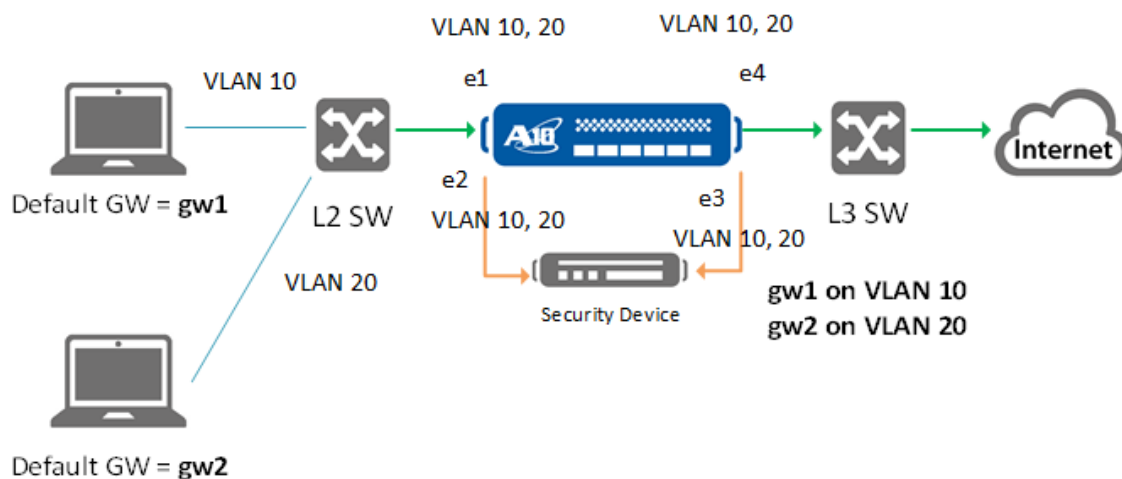
Untagged ports can belong to only a single VLAN. By default, all Ethernet data ports are untagged members of a default VLAN.

If there is only one VLAN, whether tagged or untagged, Source-NAT is supported if the Source-NAT pool belongs to the same subnet as the VEs.

L2 Deployment with Tagged VLANs

[Figure 10](#) is an example of an SSLi L2 deployment by using tagged VLANs. In this example, the traffic from tagged VLANs 10 and 20 is inspected by the security device.

Figure 10 : L2 Deployment with Tagged VLANs



CLI Configuration

The following sections describe how to configure SSLi for this deployment by using the ACOS CLI. The work-flow includes the following:

The following topics are covered:

Step 1: Initial Configuration	46
Step 2: Configuring the Network VLANs	47
Step 3: Configuring the SSLi Services	48
Step 4: Configuring Network IP Addresses	50
Step 5: Configuring the Security Device	51
Step 6: Configuring Handling of Incoming Traffic	52
Step 7: Configuring Handling of Outgoing Traffic	53
Consolidated Configuration	55

Step 1: Initial Configuration

1. Enter the configuration mode for the ACOS device:

```
ACOS>
ACOS>enable
```

```
Password:
ACOS#config
ACOS(config)#
```

The configuration mode is denoted by the ACOS(config)# prompt.

2. (Applicable to deployments using vThunder) The single-partition configuration for SSLi requires VE MAC address assignment changes, and since vThunder does not support VE MAC address assignment scheme changes in non-promiscuous mode, you must enable promiscuous mode.
3. To avoid a duplicate MAC address because of the VLAN that is shared, add the global command of `system ve-mac-scheme system-mac`.

```
ACOS(config)# system ve-mac-scheme system-mac
```

4. Assign an IP address and default gateway to the management interface:

```
ACOS(config)# interface management
ACOS(config-if:management)# ip address 10.101.7.103 255.255.252.0
ACOS(config-if:management)# ip default-gateway 10.101.4.1
ACOS(config-if:management)# exit
```

Step 2: Configuring the Network VLANs

1. Configure VLAN 10. Bind ethernet ports 1 to 4 to VLAN 10. Also, bind a virtual interface VE 10 to VLAN 10.

```
ACOS(config)# vlan 10
ACOS(config-vlan:10)# tagged ethernet 1 to 4
ACOS(config-vlan:10)# router-interface ve 10
ACOS(config-vlan:10)# exit
```

2. Configure VLAN 20. Bind ethernet port 1 to 4 to VLAN 20. Also, bind a virtual interface VE 20 to VLAN 20.

```
ACOS(config) #vlan 20
ACOS(config-vlan:20)# tagged ethernet 1 to 4
ACOS(config-vlan:20)# router-interface ve 20
ACOS(config-vlan:20)# exit
```

3. Enable the ethernet interfaces 1 to 4 on the ACOS device that are associated with the VLANs:

```
ACOS(config)# interface ethernet 1
ACOS(config-if:ethernet:1)# enable
ACOS(config-if:ethernet:1)# exit
ACOS(config)# interface ethernet 2
ACOS(config-if:ethernet:2)# enable
ACOS(config-if:ethernet:2)# exit
ACOS(config)# interface ethernet 3
ACOS(config-if:ethernet:3)# enable
ACOS(config-if:ethernet:3)# exit
ACOS(config)# interface ethernet 4
ACOS(config-if:ethernet:4)# enable
ACOS(config-if:ethernet:4)# exit
```

4. Verify the operational state of the interfaces by running the show interfaces command.

```
ACOS(config)# show interfaces brief
```

Step 3: Configuring the SSLi Services

1. Configure a cipher settings template called `cl_cipher_template`. This template is associated with the SSL client template.

```
ACOS(config)# slb template cipher cl_cipher_template
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_128_SHA
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_256_SHA
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_128_SHA256
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_128_GCM_SHA256
ACOS(config-cipher)# exit
```

2. Configure a cipher settings template called `sr_cipher_template`. This template is associated with the SSL server template.

```
ACOS(config)# slb template cipher sr_cipher_template
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_128_SHA
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_256_SHA
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_128_SHA256
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_128_GCM_SHA256
ACOS(config-cipher)# exit
```

3. Create a server SSL template called `sr_ssl` so that the VIP on the SSLi device can

operate as an SSL client and handshake with an external server. Enable forward proxy services on the template to enable SSLi operation on the VIP. Associate the `sr_cipher_template` with the server SSL template.

```
ACOS(config)# slb template server-ssl sr_ssl
ACOS(config-server ssl)# forward-proxy-enable
ACOS(config-server ssl)# cipher sr_cipher_template
```

4. Traffic selected to be forwarded to the security device is governed by the **redirect-fwd** configuration. All the IP traffic passing the vport that has the **redirect-fwd** command configured is redirected to the security device. Configure the client SSL template to provide the attributes which enable SSLi, specify the SSLi self-signed certificate, and private key. Associate the `cl_cipher_template` with the client SSL template.

```
ACOS(config)# slb template client-ssl cl_ssl
ACOS(config-client ssl)# template cipher cl_cipher_template
ACOS(config-client ssl)# forward-proxy-ca-certificate Cert123.pem key
key123
ACOS(config-client ssl)# forward-proxy-enable
```

5. Within the client SSL template, disable OCSP Stapling for SSL forward proxy.

```
ACOS(config-client ssl)# forward-proxy-ocsp-disable
```

6. Within the client SSL template, disable Certificate Revocation List (CRL) services for SSLi (forward-proxy).

```
ACOS(config-client ssl)# forward-proxy-crl-disable
```

7. Within the client SSL template, disable support for SSLv3.

```
ACOS(config-client ssl)# disable-ssl3
ACOS(config-client ssl)# exit
```

8. Configure an ACL called `ssli_in` for incoming traffic to the ACOS device. Configure the ACL to permit IP traffic from any source to any destination for VLAN 10 and VLAN 20 on the interface Ethernet 1:

```
ACOS(config)# access-list 190 remark ssli_in
ACOS(config)# access-list 190 permit ip any any vlan 10 ethernet 1
ACOS(config)# access-list 190 permit ip any any vlan 20 ethernet 1
```

9. Configure an ACL for dropping traffic called `block_quic`. Configure the ACL to

drop UDP-based traffic from any source to any destination on ports 80 and 443. If the traffic is IP-based, it is allowed to be forwarded.

```
ACOS(config)# access-list 191 remark block_quic
ACOS(config)# access-list 191 deny udp any any eq 80
ACOS(config)# access-list 191 deny udp any any eq 443
ACOS(config)# access-list 191 permit ip any any
```

10. Configure an ACL for outgoing traffic from the ACOS device called `ssli_out`. Configure the ACL to permit IP traffic from any source to any destination for VLAN 10 and VLAN 20 on the interface Ethernet 3:

```
ACOS(config)# access-list 192 remark ssli_out
ACOS(config)# access-list 192 permit ip any any vlan 10 ethernet 3
ACOS(config)# access-list 192 permit ip any any vlan 20 ethernet
```

Step 4: Configuring Network IP Addresses

On each virtual interface, enable promiscuous VIP support. When you enable promiscuous VIP support on a VE, the option is automatically enabled on each ethernet data port in the VE. Provision the virtual interfaces to allow promiscuous IP in order to subject traffic to the rules enabled on each interface. In addition, for any of the VLANs, assign an IP address and a default gateway. In this example, we assign the IP address and gateway to interface ve 10 associated with VLAN 10. Additionally, bind ACL 191 to the interfaces.

```
ACOS(config)# interface ve 10
ACOS(config-if:ve10)# access-list 191 in
ACOS(config-if:ve10)# ip address 1.1.1.1 255.255.255.0
ACOS(config-if:ve10)# ip allow-promiscuous-vip
ACOS(config-if:ve10)# exit

ACOS(config)# interface ve 20
ACOS(config-if:ve20)# access-list 191 in
ACOS(config-if:ve20)# ip allow-promiscuous-vip
ACOS(config-if:ve20)# exit
```

Step 5: Configuring the Security Device

1. Configure a server GW and its ports.

```
ACOS(config)# slb server GW 1.1.1.254
ACOS(config-real server)# port 0 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit

ACOS(config-real server)# port 0 udp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit

ACOS(config-real server)# port 443 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit

ACOS(config-real server)# port 8080 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit
ACOS(config-real server)# exit
```

2. Configure the server service group called GW_TCP_0 of type TCP. Associate GW and port 0 with the service group.

```
ACOS(config)# slb service-group GW_TCP_0 tcp
ACOS(config-slb svc group)# member GW 0
ACOS(config-slb svc group-member:0)# exit
ACOS(config-slb svc group)# exit
```

3. Configure the server service group called GW_TCP_8080 of type TCP. Associate GW and port 443 with the service group.

```
ACOS(config)# slb service-group GW_TCP_8080 tcp
ACOS(config-slb svc group)# member GW 443
ACOS(config-slb svc group-member:443)# exit
ACOS(config-slb svc group)# exit
```

4. Configure the server service group called SSLi_TCP_443 of type TCP. Associate GW and port 8080 with the service group.

```
ACOS(config)# slb service-group SSLi_TCP_443 tcp
ACOS(config-slb svc group)# member GW 8080
```

```
ACOS(config-slb svc group-member:8080)# exit
ACOS(config-slb svc group)# exit
```

5. Configure the server service group called `SSLi_TCP_0` of type TCP. Associate GW and port 0 with the service group.

```
ACOS(config)# slb service-group SSLi_TCP_0 tcp
ACOS(config-slb svc group)# member GW 0
ACOS(config-slb svc group)# exit
```

6. Configure the server service group called `SSLi_UDP_0` of type UDP. Associate GW and port 0 with the service group.
7. Configure the server service group called `GW_UDP_0` of type UDP. Associate GW and port 0 with the service group.

```
ACOS(config)# slb service-group GW_UDP_0 udp
ACOS(config-slb svc group)# member GW 0
ACOS(config-slb svc group)# exit
```

Step 6: Configuring Handling of Incoming Traffic

1. Create the wildcard VIP called `SSLi_in_ingress` at IP address 0.0.0.0 to handle traffic from the client network to the ACOS device. The `ACL 190` is bound to the wildcard VIP.

```
ACOS(config)# slb virtual-server SSLi_in_ingress 0.0.0.0 acl 190
```

2. Associate port 0 of type TCP with service group `SSLi_TCP_0`. Disable destination NAT. Within the virtual server command level, use the `redirect-fwd` command to select the forward direction for steering the IP traffic from the client destined for the security device through `ethernet 2`. Use the `use-rcv-hop-for-resp` command to send reply traffic for the session back through the same hop where the traffic was received.

```
ACOS(config-slb vserver)# port 0 tcp
ACOS(config-slb vserver-vport)# service-group SSLi_TCP_0
ACOS(config-slb vserver-vport)# no-dest-nat
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# redirect-rev ethernet 2
ACOS(config-slb vserver-vport)# exit
```

3. Within the virtual server command level, associate port 443 of type HTTPS with

the service group `SSLi_TCP_443` and the client SSL template `cl_ssl`. Disable destination NAT. Within the virtual server command level, use the `redirect-fwd` command to select the forward direction for steering the layer 2 traffic from the security device to the Internet through `ethernet 3`. Use the `use-rcv-hop-for-resp` command to send reply traffic for the session back through the same hop where the traffic was received.

```
ACOS(config-slb vserver)# port 443 https
ACOS(config-slb vserver-vport)# service-group SSLi_TCP_443
ACOS(config-slb vserver-vport)# template client-ssl cl_ssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# redirect-fwd ethernet 2
ACOS(config-slb vserver-vport)# exit
```

4. Enable similar configurations for the other ports.

```
ACOS(config-slb vserver)# port 0 udp
ACOS(config-slb vserver-vport)# service-group SSLi_UDP_0
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# redirect-fwd ethernet 2
ACOS(config-slb vserver-vport)# no-dest-nat
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# port 0 others
ACOS(config-slb vserver-vport)#service-group SSLi_UDP_0
ACOS(config-slb vserver-vport)#use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)#redirect-fwd ethernet 2
ACOS(config-slb vserver-vport)# no-dest-nat
ACOS(config-slb vserver-vport)# exit
```

Step 7: Configuring Handling of Outgoing Traffic

1. Create the wildcard VIP called `SSLi_out_ingress` at IP address `0.0.0.0` to handle traffic from the ACOS device to the outside network. The `ACL 192` is bound to the wildcard VIP.

```
ACOS(config)# slb virtual-server SSLi_out_ingress 0.0.0.0 acl 192
```

2. Associate port 0 of type TCP with service group `GW_TCP_0`. Disable destination NAT. Within the virtual server command level, use the `redirect-rev` command to

select the reverse direction for steering the layer 2 traffic from the security device to the ACOS device through ethernet 3. Use the `use-rcv-hop-for-resp` command to send reply traffic for the session back through the same hop where the traffic was received.

```
ACOS(config-slb vserver)# port 0 tcp
ACOS(config-slb vserver-vport)# service-group GW_TCP_0
ACOS(config-slb vserver-vport)# no-dest-nat
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# redirect-rev ethernet 3
ACOS(config-slb vserver-vport)# exit
```

3. Associate port 443 of type TCP with service group `GW_TCP_0`. Disable destination NAT. Within the virtual server command level, use the `redirect-rev` command to select the reverse direction for steering the layer 2 traffic from the security device to the ACOS device through ethernet 3. Use the `use-rcv-hop-for-resp` command to send reply traffic for the session back through the same hop where the traffic was received.

```
ACOS(config-slb vserver)# port 443 tcp
ACOS(config-slb vserver-vport)# service-group GW_TCP_0
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# redirect-rev ethernet 3
ACOS(config-slb vserver-vport)# exit
```

4. Enable similar configurations for the other ports.

```
ACOS(config-slb vserver)# port 0 udp
ACOS(config-slb vserver-vport)# service-group GW_UDP_0
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# redirect-rev ethernet 3
ACOS(config-slb vserver-vport)# no-dest-nat
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# port 0 others
ACOS(config-slb vserver-vport)# service-group GW_UDP_0
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# redirect-rev ethernet 3
ACOS(config-slb vserver-vport)# no-dest-nat
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# port 8080 http
```

```
ACOS(config-slb vserver-vport)# service-group GW_TCP_8080
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# redirect-rev ethernet 3
ACOS(config-slb vserver-vport)# template server-ssl sr_ssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
```

Consolidated Configuration

```
ACOS(config)# show run
!Current configuration: 2593 bytes
!Configuration last updated at 17:01:10 PDT Fri May 19 2017
!Configuration last saved at 14:15:38 PDT Wed May 17 2017
!64-bit Advanced Core OS (ACOS) version 4.1.1-P3, build 28 (May-12-
2017,04:15)
!
access-list 190 remark ssli_in
!
access-list 190 permit ip any any vlan 10 ethernet 1
!
access-list 190 permit ip any any vlan 20 ethernet 1
!
access-list 191 remark block_quic
!
access-list 191 deny udp any any eq 80
!
access-list 191 deny udp any any eq 443
!
access-list 191 permit ip any any
!
access-list 192 remark ssli_out
!
access-list 192 permit ip any any vlan 10 ethernet 3
!
access-list 192 permit ip any any vlan 20 ethernet 3
!
multi-config enable
!
system ve-mac-scheme system-mac
!
vlan 10
```

```
    tagged ethernet 1 to 4
    router-interface ve 10
!
vlan 20
    tagged ethernet 1 to 4
    router-interface ve 20
!
interface management
    ip address 10.101.7.103 255.255.252.0
    ip default-gateway 10.101.4.1
!
interface ethernet 1
    enable
!
interface ethernet 2
    enable
!
interface ethernet 3
    enable
!
interface ethernet 4
    enable
!
interface ethernet 5
!
interface ethernet 6
!
interface ethernet 7
!
interface ethernet 8
!
interface ve 10
    access-list 191 in
    ip address 1.1.1.1 255.255.255.0
    ip allow-promiscuous-vip
!
interface ve 20
    access-list 191 in
    ip allow-promiscuous-vip
!
```



```
!  
ip route 0.0.0.0 /0 1.1.1.254  
!  
slb template cipher cl_cipher_template  
    TLS1_ECDHE_RSA_AES_128_SHA  
    TLS1_ECDHE_RSA_AES_256_SHA  
    TLS1_ECDHE_RSA_AES_128_SHA256  
    TLS1_ECDHE_RSA_AES_128_GCM_SHA256  
    user-tag Security,ssli_in  
!  
slb template cipher sr_cipher_template  
    TLS1_ECDHE_RSA_AES_128_SHA  
    TLS1_ECDHE_RSA_AES_256_SHA  
    TLS1_ECDHE_RSA_AES_128_SHA256  
    TLS1_ECDHE_RSA_AES_128_GCM_SHA256  
    user-tag Security,ssli_out  
!  
slb template server-ssl sr_ssl  
    forward-proxy-enable  
    template cipher sr_cipher_template  
!  
slb server GW 1.1.1.254  
    user-tag Security,ssli_in  
    port 0 tcp  
        health-check-disable  
        user-tag Security,ssli_in_srv_port_0_tcp  
    port 0 udp  
        health-check-disable  
        user-tag Security,ssli_in_srv_port_0_udp  
    port 443 tcp  
        health-check-disable  
        user-tag Security,ssli_in_srv_port_443_tcp  
    port 8080 tcp  
        health-check-disable  
        user-tag Security,ssli_in_srv_port_8080_tcp  
!  
  
slb server GW 1.1.1.254  
    user-tag Security,ssli_in  
    port 0 tcp
```

```
    health-check-disable
    user-tag Security,ssli_in_srv_port_0_tcp
port 0 udp
    health-check-disable
    user-tag Security,ssli_in_srv_port_0_udp
port 443 tcp
    health-check-disable
    user-tag Security,ssli_in_srv_port_443_tcp
port 8080 tcp
    health-check-disable
    user-tag Security,ssli_in_srv_port_8080_tcp

slb service-group GW_TCP_0 tcp
    member GW 0
!
slb service-group GW_TCP_8080 tcp
    member GW 443
!
slb service-group GW_UDP_0 udp
    member GW 0
!
slb service-group SSLi_TCP_0 tcp
    member GW 0
!
slb service-group SSLi_TCP_443 tcp
    member GW 8080
!
slb service-group SSLi_UDP_0 udp
    member GW 0
!
slb template client-ssl cl_ssl
    template cipher cl_cipher_template
    forward-proxy-ca-certificate Cert123.pem key key123
    forward-proxy-ocsp-disable
    forward-proxy-crl-disable
    forward-proxy-enable
    disable-sslv3
!
slb virtual-server SSLi_in_ingress 0.0.0.0 acl 190
    port 0 tcp
```

```
    service-group SSLi_TCP_0
    use-rcv-hop-for-resp
    redirect-fwd ethernet 2
    no-dest-nat
port 0 udp
    service-group SSLi_UDP_0
    use-rcv-hop-for-resp
    redirect-fwd ethernet 2
    no-dest-nat
port 0 others
    service-group SSLi_UDP_0
    use-rcv-hop-for-resp
    redirect-fwd ethernet 2
    no-dest-nat
port 443 https
    service-group SSLi_TCP_443
    use-rcv-hop-for-resp
    redirect-fwd ethernet 2
    template client-ssl cl_ssl
    no-dest-nat port-translation
!
slb virtual-server SSLi_out_ingress 0.0.0.0 acl 192
port 0 tcp
    service-group GW_TCP_0
    use-rcv-hop-for-resp
    redirect-rev ethernet 3
    no-dest-nat
port 0 udp
    service-group GW_UDP_0
    use-rcv-hop-for-resp
    redirect-rev ethernet 3
    no-dest-nat
port 0 others
    service-group GW_UDP_0
    use-rcv-hop-for-resp
    redirect-rev ethernet 3
    no-dest-nat
port 443 tcp
    service-group GW_TCP_0
    use-rcv-hop-for-resp
```

```

    redirect-rev ethernet 3
    no-dest-nat
    port 8080 http
    service-group GW_TCP_8080
    use-rcv-hop-for-resp
    redirect-rev ethernet 3
    template server-ssl sr_ssl
    no-dest-nat port-translation
!
end

```

GUI Configuration

This section describe how to configure SSLi for this deployment by using the AOCS GUI.

The following topics are covered:

Step 1: Configuring the Network VLANs	60
Step 2: Configuring the SSLi Services	61
Step 3: Configuring the VIPs	65
Step 4: Configuring the Security Device	66
Step 5: Configuring Handling of Incoming Traffic	67
Step 6: Configuring Handling of Outgoing Traffic	69

Step 1: Configuring the Network VLANs

In this section, first create the VLANs 10 and 20 and the interfaces e1, e2, e3, and e4. Associate the e1, e2, e3, and e4 interfaces with the VLANs. Finally, enable the interfaces.

To create the VLANs:

1. Navigate to **Network > VLAN**.
2. Click **Create**.
The **Create VLAN** page is displayed.
3. Enter the following details to create VLAN 10.

VLAN ID: 10

Name: VLAN10

4. Select **Create Virtual Interface**. For **Tagged Ethernet**, select 1,2,3, and 4.
5. Click **Create VLAN**.
VLAN10 is created.
6. Similarly follow the above steps to create VLAN 20.

The tagged VLANs are created. You must now enable the interfaces associated with the VLANs.

To enable the network interfaces associated with the tagged VLANs:

1. Navigate to **Network > Interfaces**.
2. Select **e1**, **e2**, **e3**, and **e4**.
3. Click **Enable** to enable the interfaces.
The icons for the interfaces change to a green up-arrow.

You can now proceed to configuring the SSLi services.

Step 2: Configuring the SSLi Services

In this section, create the two cipher templates to be associated with the SSL templates. Next, create the server SSL and client SSL templates. Associate the client cipher template with the client SSL template. Associate the server cipher template with the server SSL template. Finally, create the ACL lists to define how to handle incoming traffic, outgoing traffic, and which traffic to drop for inspection.

Creating the Client and Server Cipher Templates

A cipher template contains a list of ciphers. A client or server, that connects to a virtual port, can use only the ciphers that are listed in the template. A cipher template must be bound to a client or server SSL template.

1. Navigate to **ADC > Templates > SSL**.
2. Select **Create > SSL Cipher**.
The **Create SSL Cipher Template** is displayed.
3. Enter the name as `cl_cipher_template`.

- For **Cipher Config**, click **Add**.
 - For **Cipher Suite**, select **TLS1_ECDHE_RSA_AES_128_SHA**.
4. Add the following ciphers by clicking **Add** for each cipher and selecting the appropriate one from the drop-down menu:
- TLS1_ECDHE_RSA_AES_128_SHA
 - TLS1_ECDHE_RSA_AES_256_SHA
 - TLS1_ECDHE_RSA_AES_128_SHA256
 - TLS1_ECDHE_RSA_AES_128_GCM_SHA256

NOTE: Priority values are supported only for client-SSL templates. If a cipher template is used by a server-SSL template, the priority values in the cipher template are ignored. In this example, since all the ciphers have equal priority, ACOS selects the strongest available cipher.

5. Click **Create**.

The `cl_cipher_template` cipher template is created.

6. Repeat the procedure to create a server cipher template called `sr_cipher_template` and configured with the following ciphers:
- TLS1_RSA_AES_128_SHA
 - TLS1_RSA_AES_128_GCM_SHA256
 - TLS1_ECDHE_RSA_AES_128_SHA
 - TLS1_ECDHE_RSA_AES_128_SHA256

Proceed to create the client SSL template and the server SSL template and associating these templates with the correct SSL cipher template.

Creating the Client SSL Template

1. Navigate to **Security > SSLi > Templates**.
2. Select **Create > Client SSL**.
The **Create Client SSL Template** is displayed.
3. For **Name**, enter `cl_ssl`.

4. Under the **Basic** tab, select **Forward Proxy Enable** for SSLi.
5. For **SSLi Forward Proxy CA Cert**, select your appropriate certificate.
6. For **SSLi Forward Proxy CA Key**, select your appropriate key.
7. Under **Ciphers**, select **Template**. From the drop-down menu, select **cl_cipher_template**.

NOTE: You had already created the client cipher template in [Creating the Client and Server Cipher Templates](#).

8. Under **Advanced** tab, select **Forward Proxy OCSP Disable**.
9. Click **OK** to create the client SSL template.

Creating the Server SSL Template

1. Navigate to **Security > SSLi > Templates**.
2. Select **Create > Server SSL**.
The **Create Server SSL Template** is displayed.
3. For **Name**, enter `sr_ssl`.
4. Select **SSL Forward Proxy Enable**.
5. For **Cipher**, select **Template**.
From the drop-down menu, select `sr_cipher_template`.

NOTE: You had already created the server cipher template in [Creating the Client and Server Cipher Templates](#).

6. Click **Create**.
The server SSL template is created.

Creating an ACL

You must create three ACLs to govern three types of traffic: incoming traffic, traffic to be dropped, and outgoing traffic. To create the `ACL 190` for incoming traffic:

1. Navigate to **Security > Access List > Extended**.
2. Click **Create**.
The **Create Extended Access List** page is displayed.

3. For **ID**, enter 190.
 4. For **Sequence number**, enter 1.
 5. Select **Remark**.
 6. For **Remark**, enter `ssli_in`.
 7. Select **Create**.
- The `ACL 190` is created.

You can now add rules to the ACL.

Adding Rules to an ACL

To add a rule to `ACL 190` that allows IP traffic on VLAN 10 and on e1 to pass through.

1. Select `ACL 190` and click **Add New Rule**.
2. Enter the **Sequence Number** as 2.
3. Select **Entry**.
4. For **Action**, select **Permit**.
5. For **Service**, select **Protocol** and **IP**.
6. For **Source Address**, select **Source Address** and **Any**.
7. For **Destination Address**, select **Destination Address** and **Any**.
8. For **Match Type**, select **VLAN**.
9. Enter **VLAN** value as 10.
10. For **Interface Type**, select **Ethernet**.
11. Select the **Ethernet number** from the drop down as 1.
12. Click **Create**.

A new rule is added to `ACL 190`.

You can repeat the procedure to add another rule for `ACL 190` that allows IP traffic on VLAN 20 e1 to pass through.

Similarly, create `ACL 191` and `ACL 192`.

The configuration statements are provided for reference:


```
access-list 191 remark block_quic
!
access-list 191 deny udp any any eq 80
!
access-list 191 deny udp any any eq 443
!
access-list 191 permit ip any any
!
access-list 192 remark ssli_out
!
access-list 192 permit ip any any vlan 10 ethernet 3
!
access-list 192 permit ip any any vlan 20 ethernet 3
```

You can now associate the ACLs with the VIPs.

Step 3: Configuring the VIPs

The virtual interfaces or VIPs are already created as VE 10 and VE 20 in section [Step 1: Configuring the Network VLANs](#). The following section modifies the properties of the VIPs.

1. Navigate to **Network > Interfaces > Virtual Ethernets**.
2. Select **10** and click **Edit**.
3. Under **IP**, add in IP address as **1.1.1.1** and gateway as **255.255.255.0**.
4. Enable **Allow Promiscuous VIP**.
5. Select **Access List** as **191**.
You created the Access List in [Creating an ACL](#).
6. Click **Update**.
Interface **VE 10** is updated.
7. Select **VE 20** and click **Edit**.
8. Select **Access List** as **191**.
9. Enable **Allow Promiscuous VIP**.
10. Click **Update**.
Interface **VE 20** is updated.

You are now ready to define the real server and its ports.

Step 4: Configuring the Security Device

In this section, first create the real server `GW`. Then, create service groups and associate the real server and a port to each of the service groups.

Creating the Real Server and its Ports

To create the real server `GW` and its ports:

1. Navigate to **ADC > SLB > Servers**.
2. Click **Create** and configure the following real server settings:
 - Name: `GW`
 - Type: **IPv4**
 - Host: `1.1.1.254`
 - Action: **Enable**
3. Select **Disable Health Check**.
4. Under **Port**, click **Create**, and configure the following port settings:
5. Port Number: **0**
6. Protocol: **TCP**
7. Select **Disable Health Check**.
8. Click **Create**.
Port 0 of type TCP is now associated with `GW`.
9. Similarly, associate the following ports with `GW`:
10. Port **0** of type UDP.
11. Port **443** of type TCP.
12. Port **8080** of type TCP.
13. Click **Update** to create the real server `GW`.

Proceed to creating the service groups.

Creating the Service Groups

To create and associate the service group `GW_TCP_0` with `GW` and **port 0**:

1. Navigate to **ADC > SLB > Service Groups**.
2. Click **Create** and configure the following settings:
3. **Name:** `GW_TCP_0`
4. **Protocol:** `TCP`
5. Under **Member**, select **Create**.
The **Create Member** page is displayed.
6. Under **Choose Creation Type**, select **Existing Server**.
7. For **Server**, select `GW` from the drop-down menu.
8. For **Port**, select **0**.
9. Select **State** as **Enable**.
10. Click **Create**.
`GW` and **port 0** are now associated with the service group `GW_TCP_0 tcp`.
11. Repeat the procedure to configure the following:
 - Service group `GW_TCP_8080` of type **TCP**.
Associate `GW` and **port 8080** with this service group.
 - Service group `SSLi_TCP_443` of type **TCP**.
Associate `GW` and **port 443** with this service group.
 - Service group `SSLi_TCP_0` of type **TCP**.
Associate `GW` and **port 0** with this service group.
 - Service group `SSLi_UDP_0` of type **UDP**.
Associate `GW` and **port 0** with this service group.
 - Service group `GW_UDP_0` of type **UDP**.
Associate `GW` and **port 0** with this service group.

Step 5: Configuring Handling of Incoming Traffic

Create a virtual server for incoming traffic called `SSLi_in_ingress`.

1. Navigate to **ADC > SLB > Virtual Servers**.
2. Click **Create**.
The **Create Virtual Server** page is displayed.
3. For **Name**, enter `SSLi_in_ingress`, and configure the following

4. Select **Wildcard**.
5. For **Address Type**, select **IPv4**.
6. For **Action**, select **Enable**.
7. For **Access List**, select **190**.
8. Under **Virtual Port**, click **Create**.
The **Create Virtual Port** page is displayed. Configure the following:
9. For **Protocol**, select **TCP**.
10. For **Port**, select **0**.
11. For **Action**, select **Enable**.
12. For **Service Group**, select **SSLi_TCP_0**.
13. Expand **General Fields** and select the following:
 - No Dest NAT
 - Use Rcv Hop For Resp
 - For Redirect Forward, select Ethernet
14. Click **Create**.
The **Virtual Port** is created and added to the virtual server.
15. Similarly, create and add ports of the following properties:

```
port 0 udp
    service-group SSLi_UDP_0
    use-rcv-hop-for-resp
    redirect-fwd ethernet 2
    no-dest-nat
port 0 others
    service-group SSLi_UDP_0
    use-rcv-hop-for-resp
    redirect-fwd ethernet 2
    no-dest-nat
port 443 https
    service-group SSLi_TCP_443
    use-rcv-hop-for-resp
    redirect-fwd ethernet 2
    template client-ssl cl_ssl
```

```
no-dest-nat port-translation
```

Step 6: Configuring Handling of Outgoing Traffic

Create a virtual server for outgoing traffic called `SSLi_out_ingress`.

1. Navigate to **ADC > SLB > Virtual Servers**.
2. Click **Create**.
The **Create Virtual Server** page is displayed.
3. For **Name**, enter `SSLi_out_ingress`, and configure the following:
 - a. Navigate to **ADC > SLB > Virtual Servers**.
 - b. Click **Create**.
The **Create Virtual Server** page is displayed.
 - c. For **Name**, enter `SSLi_out_ingress`, and configure the following:
 - i. Select **Wildcard**.
 - ii. For **Address Type**, select **IPv4**.
 - iii. For **Action**, select **Enable**.
 - iv. For **Access List**, select **192**.
 - d. Under **Virtual Port**, click **Create**.
The **Create Virtual Port** page is displayed. Configure the following:
 - i. For **Protocol**, select **TCP**.
 - ii. For **Port**, select **0**.
 - iii. For **Action**, select **Enable**.
 - iv. For **Service Group**, select **GW_TCP_0**.
 - e. Expand **General Fields** and select the following:
 - i. No Dest NAT.
 - ii. Use Rcv Hop For Resp.
 - iii. For **Redirect Reverse**, select **Ethernet**, and then select **3**.
 - f. Click **Create**.
The **Virtual Port** is created and added to the virtual server.

g. Similarly, create and add ports of the following properties:

```
port 0 udp
    service-group GW_UDP_0
    use-rcv-hop-for-resp
    redirect-rev ethernet 3
    no-dest-nat
port 0 others
    service-group GW_UDP_0
    use-rcv-hop-for-resp
    redirect-rev ethernet 3
    no-dest-nat
port 443 tcp
    service-group GW_TCP_0
    use-rcv-hop-for-resp
    redirect-rev ethernet 3
    no-dest-nat
port 8080 http
    service-group GW_TCP_8080
    use-rcv-hop-for-resp
    redirect-rev ethernet 3
    template server-ssl sr_ssl
    no-dest-nat port-translation
```

h. Finally, click **Update** to complete creating the virtual server.

4. Under **Virtual Port**, click **Create**.

The **Create Virtual Port** page is displayed. Configure the following:

- a. For **Protocol**, select **TCP**.
- b. For **Port**, select **0**.
- c. For **Action**, select **Enable**.
- d. For **Service Group**, select **GW_TCP_0**.

5. Expand **General Fields** and select the following:

- a. No Dest NAT.
- b. Use Rcv Hop For Resp.
- c. For **Redirect Reverse**, select **Ethernet**, and then select **3**.

6. Click **Create**.

The **Virtual Port** is created and added to the virtual server.

7. Similarly, create and add ports of the following properties:

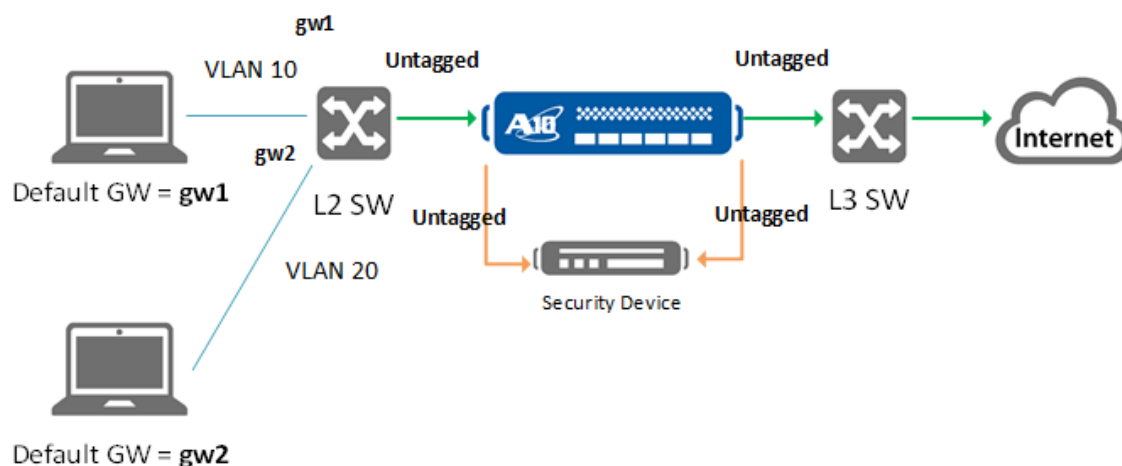
```
port 0 udp
  service-group GW_UDP_0
  use-rcv-hop-for-resp
  redirect-rev ethernet 3
  no-dest-nat
port 0 others
  service-group GW_UDP_0
  use-rcv-hop-for-resp
  redirect-rev ethernet 3
  no-dest-nat
port 443 tcp
  service-group GW_TCP_0
  use-rcv-hop-for-resp
  redirect-rev ethernet 3
  no-dest-nat
port 8080 http
  service-group GW_TCP_8080
  use-rcv-hop-for-resp
  redirect-rev ethernet 3
  template server-ssl sr_ssl
  no-dest-nat port-translation
```

8. Finally, click **Update** to complete creating the virtual server.

L2 Deployment with Untagged VLANs

[Figure 11](#) is an example of an SSLi L2 deployment by using untagged VLANs.

Figure 11 : L2 Deployment with Untagged VLANs



CLI Configuration

This section describes how to configure SSLi for this deployment by using the ACOS CLI.

For GUI configuration, see [GUI Configuration](#).

The following topics are covered:

Step 1: Initial Configuration	72
Step 2: Configuring the Default VLAN	73
Step 3: Configuring the SSLi services	74
Step 4: Configuring Network IP Addresses	76
Step 5: Configuring the Security Device	76
Step 6: Configuring Handling of Incoming Traffic	78
Step 7: Configuring Handling of Outgoing Traffic	79
Consolidated Configuration	81

Step 1: Initial Configuration

1. Enter the configuration mode for the ACOS device:

```
ACOS>
```



```
ACOS>enable
Password:
ACOS#config
ACOS(config)#
```

The configuration mode is denoted by the `ACOS(config)#` prompt.

2. To avoid a duplicate MAC address because of the VLAN that is shared, add the global command of `system ve-mac-scheme system-mac`.

```
ACOS(config)# system ve-mac-scheme system-mac
```

3. Assign an IP address and default gateway to the management interface:

```
ACOS(config)# interface management
ACOS(config-if:management)# ip address 10.101.7.103 255.255.252.0
ACOS(config-if:management)# ip default-gateway 10.101.4.1
ACOS(config-if:management)# exit
```

Step 2: Configuring the Default VLAN

1. Configure the default VLAN. Bind ethernet ports 1 to 4 to the VLAN. Also, bind a virtual interface `ve` to the VLAN. In this example, a default VLAN of 850 is configured.

```
ACOS(config)# vlan 850
ACOS(config-vlan:850)# untagged ethernet 1 to 4
ACOS(config-vlan:850)# router-interface ve 850
ACOS(config-vlan:850)# exit
```

2. Enable the ethernet interfaces 1 to 4 on the ACOS device that are associated with the VLAN:

```
ACOS(config)# interface ethernet 1
ACOS(config-if:ethernet:1)# enable
ACOS(config-if:ethernet:1)# exit

ACOS(config)# interface ethernet 2
ACOS(config-if:ethernet:2)# enable
ACOS(config-if:ethernet:2)# exit

ACOS(config)# interface ethernet 3
ACOS(config-if:ethernet:3)# enable
```

```
ACOS(config-if:ethernet:3)# exit

ACOS(config)# interface ethernet 4
ACOS(config-if:ethernet:4)# enable
ACOS(config-if:ethernet:4)# exit
```

3. Verify the operational state of the interfaces by running the show interfaces command.

```
ACOS(config)# show interfaces brief
```

Step 3: Configuring the SSLi services

1. Configure a cipher settings template called `cl_cipher_template`. This template is associated with the SSL client template.

```
ACOS(config)# slb template cipher cl_cipher_template
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_128_SHA
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_256_SHA
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_128_SHA256
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_128_GCM_SHA256
ACOS(config-cipher)# exit
```

2. Configure a cipher settings template called `sr_cipher_template`. This template is associated with the SSL server template.

```
ACOS(config)# slb template cipher sr_cipher_template
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_128_SHA
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_256_SHA
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_128_SHA256
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_128_GCM_SHA256
ACOS(config-cipher)# exit
```

3. Create a server SSL template called `sr_ssl` so that the VIP on the SSLi device can operate as an SSL client and handshake with an external server. Enable forward proxy services on the template to enable SSLi operation on the VIP. Associate the `sr_cipher_template` with the server SSL template.

```
ACOS(config)# slb template server-ssl sr_ssl
ACOS(config-server ssl)# forward-proxy-enable
ACOS(config-server ssl)# cipher sr_cipher_template
```

4. Configure an SLB template of type TCP.

```
ACOS(config)# slb template tcp tcp
```

5. Configure an SLB template of type tcp-proxy.

```
ACOS(config)# slb template tcp-proxy tcp-proxy
```

6. Traffic selected to be forwarded to the security device is governed by the **redirect-fwd** configuration. All the IP traffic passing the vport that has the **redirect-fwd** command configured is redirected to the security device. Configure the client SSL template to provide the attributes which enable SSLi, specify the SSLi self-signed certificate, and private key. Associate the `cl_cipher_template` with the client SSL template.

```
ACOS(config)# slb template client-ssl cl_ssl
ACOS(config-client ssl)# template cipher cl_cipher_template
ACOS(config-client ssl)# forward-proxy-ca-certificate Cert123.pem key
key123
ACOS(config-client ssl)# forward-proxy-enable
```

7. Within the client SSL template, disable OCSP Stapling for SSL forward proxy.

```
ACOS(config-client ssl)# forward-proxy-ocsp-disable
```

8. Within the client SSL template, disable Certificate Revocation List (CRL) services for SSLi (forward-proxy).

```
ACOS(config-client ssl)# forward-proxy-crl-disable
```

9. Within the client SSL template, disable support for SSLv3.

```
ACOS(config-client ssl)# disable-ssl3
ACOS(config-client ssl)# exit
```

10. Configure the ACL to permit IP traffic from any source to any destination for the VLAN on the interface Ethernet 1:

```
ACOS(config)# access-list 190 remark ssl_i_in
ACOS(config)# access-list 190 permit ip any any vlan 850 ethernet 1
```

11. Configure an ACL for dropping traffic called `block_quic`. Configure the ACL to drop UDP-based traffic from any source to any destination on ports 80 and 443. If the traffic is IP-based, it is allowed to be forwarded.

```
ACOS(config)# access-list 191 remark block_quic
ACOS(config)# access-list 191 deny udp any any eq 80
```

```
ACOS(config)# access-list 191 deny udp any any eq 443
ACOS(config)# access-list 191 permit ip any any
```

12. Configure an ACL for outgoing traffic from the ACOS device called `ssli_out`. Configure the ACL to permit IP traffic from any source to any destination for the VLAN on the interface Ethernet 3:

```
ACOS(config)# access-list 192 remark ssli_out
ACOS(config)# access-list 192 permit ip any any vlan 850 ethernet 3
```

Step 4: Configuring Network IP Addresses

On the virtual interface 850, enable promiscuous VIP support. When you enable promiscuous VIP support on a VE, the option is automatically enabled on each ethernet data port in the VE. Provision the virtual interfaces to allow promiscuous IP in order to subject traffic to the rules enabled on each interface. In addition, assign an IP address and a default gateway to the VLAN. In this example, we assign the IP address and gateway to interface ve 850. Additionally, bind ACL 191 to the interface.

```
ACOS(config)# interface ve 850
ACOS(config-if:ve850)# access-list 191 in
ACOS(config-if:ve850)# ip address 1.1.1.1 255.255.255.0
ACOS(config-if:ve850)# ip allow-promiscuous-vip
ACOS(config-if:ve850)# exit
```

Step 5: Configuring the Security Device

1. Configure the server `GW` and its ports.

```
ACOS(config)# slb server GW 1.1.1.254
ACOS(config-real server)# port 0 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit

ACOS(config-real server)# port 0 udp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit

ACOS(config-real server)# port 443 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit
```

```
ACOS(config-real server)# port 8080 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit
ACOS(config-real server)# exit
```

2. Configure the server service group called `GW_TCP_0` of type TCP. Associate GW and port 0 with the service group.

```
ACOS(config)# slb service-group GW_TCP_0 tcp
ACOS(config-slb svc group)# member GW 0
ACOS(config-slb svc group-member:0)# exit
ACOS(config-slb svc group)# exit
```

3. Configure the server service group called `GW_TCP_8080` of type TCP. Associate GW and port 443 with the service group.

```
ACOS(config)# slb service-group GW_TCP_8080 tcp
ACOS(config-slb svc group)# member GW 443
ACOS(config-slb svc group-member:443)# exit
ACOS(config-slb svc group)# exit
```

4. Configure the server service group called `SSLi_TCP_443` of type TCP. Associate GW and port 8080 with the service group.

```
ACOS(config)# slb service-group SSLi_TCP_443 tcp
ACOS(config-slb svc group)# member GW 8080
ACOS(config-slb svc group-member:8080)# exit
ACOS(config-slb svc group)# exit
```

5. Configure the server service group called `SSLi_TCP_0` of type TCP. Associate GW and port 0 with the service group.

```
ACOS(config)# slb service-group SSLi_TCP_0 tcp
ACOS(config-slb svc group)# member GW 0
ACOS(config-slb svc group)# exit
```

6. Configure the server service group called `SSLi_UDP_0` of type UDP. Associate GW and port 0 with the service group.

```
ACOS(config)# slb service-group SSLi_UDP_0 udp
ACOS(config-slb svc group)# member GW 0
ACOS(config-slb svc group)# exit
```

7. Configure the server service group called `GW_UDP_0` of type UDP. Associate `GW` and port 0 with the service group.

```
ACOS(config)# slb service-group GW_UDP_0 udp
ACOS(config-slb svc group)# member GW 0
ACOS(config-slb svc group)# exit
```

Step 6: Configuring Handling of Incoming Traffic

1. Create the wildcard VIP called `SSLi_in_ingress` at IP address 0.0.0.0 to handle traffic from the client network to the ACOS device. The `ACL 190` is bound to the wildcard VIP.

```
ACOS(config)# slb virtual-server SSLi_in_ingress 0.0.0.0 acl 190
```

2. Associate port 0 of type TCP with service group `SSLi_TCP_0`. Disable destination NAT.

```
ACOS(config-slb vserver)# port 0 tcp
ACOS(config-slb vserver-vport)# service-group SSLi_TCP_0
ACOS(config-slb vserver-vport)# no-dest-nat
```

3. Within the virtual server command level, use the `redirect-fwd` command to select the forward direction for steering the layer 2 traffic from the client destined for the security device through `ethernet 2`. Use the `use-rcv-hop-for-resp` command to send reply traffic for the session back through the same hop where the traffic was received.

```
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# redirect-fwd ethernet 2
ACOS(config-slb vserver-vport)# exit
```

4. Within the virtual server command level, associate port 443 of type HTTPS with the service group `SSLi_TCP_443` and the client SSL template `cl_ssl`. Disable destination NAT.

```
ACOS(config-slb vserver)# port 443 https
ACOS(config-slb vserver-vport)# service-group SSLi_TCP_443
ACOS(config-slb vserver-vport)# template client-ssl cl_ssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
```

5. Within the virtual server command level, use the `redirect-fwd` command to

select the forward direction for steering the layer 2 traffic from the security device to the Internet through `ethernet 3`. Use the `use-rcv-hop-for-resp` command to send reply traffic for the session back through the same hop where the traffic was received.

```
ACOS(config-slbf vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slbf vserver-vport)# redirect-fwd ethernet 2
ACOS(config-slbf vserver-vport)# exit
```

6. Enable similar configurations for the other ports.

```
ACOS(config-slbf vserver)# port 0 udp
ACOS(config-slbf vserver-vport)# service-group SSLi_UDP_0
ACOS(config-slbf vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slbf vserver-vport)# redirect-fwd ethernet 2
ACOS(config-slbf vserver-vport)# no-dest-nat
ACOS(config-slbf vserver-vport)# exit
ACOS(config-slbf vserver)# port 0 others
ACOS(config-slbf vserver-vport)# service-group SSLi_UDP_0
ACOS(config-slbf vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slbf vserver-vport)# redirect-fwd ethernet 2
ACOS(config-slbf vserver-vport)# no-dest-nat
ACOS(config-slbf vserver-vport)# exit
```

Step 7: Configuring Handling of Outgoing Traffic

1. Create the wildcard VIP called `SSLi_out_ingress` at IP address `0.0.0.0` to handle traffic from the ACOS device to the outside network. The `ACL 192` is bound to the wildcard VIP.

```
ACOS(config)# slbf virtual-server SSLi_out_ingress 0.0.0.0 acl 192
```

2. Associate port 0 of type TCP with service group `GW_TCP_0`. Disable destination NAT.

```
ACOS(config-slbf vserver)# port 0 tcp
ACOS(config-slbf vserver-vport)# service-group GW_TCP_0
ACOS(config-slbf vserver-vport)# no-dest-nat
```

3. Within the virtual server command level, use the `redirect-rev` command to select the reverse direction for steering the layer 2 traffic from the security device

to the ACOS device through ethernet 3. Use the `use-rcv-hop-for-resp` command to send reply traffic for the session back through the same hop where the traffic was received.

```
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# redirect-rev ethernet 3
ACOS(config-slb vserver-vport)# exit
```

4. Associate port 443 of type TCP with service group `GW_TCP_0`. Disable destination NAT.

```
ACOS(config-slb vserver)# port 443 tcp
ACOS(config-slb vserver-vport)# service-group GW_TCP_0
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
```

5. Within the virtual server command level, use the `redirect-rev` command to select the reverse direction for steering the layer 2 traffic from the security device to the ACOS device through ethernet 3. Use the `use-rcv-hop-for-resp` command to send reply traffic for the session back through the same hop where the traffic was received.

```
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# redirect-rev ethernet 3
ACOS(config-slb vserver-vport)# exit
```

6. Enable similar configurations for the other ports.

```
ACOS(config-slb vserver)# port 0 udp
ACOS(config-slb vserver-vport)# service-group GW_UDP_0
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# redirect-rev ethernet 3
ACOS(config-slb vserver-vport)# no-dest-nat
ACOS(config-slb vserver-vport)# exit

ACOS(config-slb vserver)# port 0 others
ACOS(config-slb vserver-vport)# service-group GW_UDP_0
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# redirect-rev ethernet 3
ACOS(config-slb vserver-vport)# no-dest-nat
ACOS(config-slb vserver-vport)# exit

ACOS(config-slb vserver)# port 8080 http
```



```
ACOS(config-slb vserver-vport)# service-group GW_TCP_8080
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# redirect-rev ethernet 3
ACOS(config-slb vserver-vport)# template server-ssl sr_ssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
```

Consolidated Configuration

```
TH3230S#show run
!Current configuration: 2333 bytes
!Configuration last updated at 17:03:06 PDT Fri May 19 2017
!Configuration last saved at 14:15:38 PDT Wed May 17 2017
!64-bit Advanced Core OS (ACOS) version 4.1.1-P3, build 28 (May-12-
2017,04:15)
!
    access-list 190 remark ssli_in
    !
    access-list 190 permit ip any any vlan 850 ethernet 1
    !
    access-list 191 remark block_quic
    !
    access-list 191 deny udp any any eq 80
    !
    access-list 191 deny udp any any eq 443
    !
    access-list 191 permit ip any any
    !
    access-list 192 remark ssli_out
    !
    access-list 192 permit ip any any vlan 850 ethernet 3
!
multi-config enable
!
!
system ve-mac-scheme system-mac
!
vlan 850
    untagged ethernet 1 to 4
    router-interface ve 850
!
```

```
!  
interface management  
    ip address 10.101.7.103 255.255.252.0  
    ip default-gateway 10.101.4.1  
!  
interface ethernet 1  
    enable  
!  
interface ethernet 2  
    enable  
!  
interface ethernet 3  
    enable  
!  
interface ethernet 4  
    enable  
!  
interface ethernet 5  
!  
interface ethernet 6  
!  
interface ethernet 7  
!  
interface ethernet 8  
!  
interface ve 850  
    access-list 191 in  
    ip address 1.1.1.1 255.255.255.0  
    ip allow-promiscuous-vip  
!  
ip route 0.0.0.0 /0 1.1.1.254  
!  
slb template cipher cl_cipher_template  
    TLS1_ECDHE_RSA_AES_128_SHA  
    TLS1_ECDHE_RSA_AES_256_SHA  
    TLS1_ECDHE_RSA_AES_128_SHA256  
    TLS1_ECDHE_RSA_AES_128_GCM_SHA256  
!  
slb template cipher sr_cipher_template
```

```
TLS1_ECDHE_RSA_AES_128_SHA
TLS1_ECDHE_RSA_AES_256_SHA
TLS1_ECDHE_RSA_AES_128_SHA256
TLS1_ECDHE_RSA_AES_128_GCM_SHA256
!
slb template server-ssl sr_ssl
  forward-proxy-enable
  template cipher sr_cipher_template
!
slb server GW 1.1.1.254
  user-tag Security,ssli_in
  port 0 tcp
    health-check-disable
  port 0 udp
    health-check-disable
  port 443 tcp
    health-check-disable
  port 8080 tcp
    health-check-disable
!
slb service-group GW_TCP_0 tcp
  member GW 0
!
slb service-group GW_TCP_8080 tcp
  member GW 443
!
slb service-group GW_UDP_0 udp
  member GW 0
!
slb service-group SSLi_TCP_0 tcp
  member GW 0
!
slb service-group SSLi_TCP_443 tcp
  member GW 8080
!
slb service-group SSLi_UDP_0 udp
  member GW 0
!
slb template client-ssl cl_ssl
  template cipher cl_cipher_template
```

```
forward-proxy-ca-certificate Cert123.pem key key123
forward-proxy-ocsp-disable
forward-proxy-crl-disable
forward-proxy-enable
disable-sslv3
!
slb virtual-server SSLi_in_ingress 0.0.0.0 acl 190
user-tag Security,ssli_in
port 0 tcp
    service-group SSLi_TCP_0
    use-rcv-hop-for-resp
    redirect-fwd ethernet 2
    no-dest-nat
port 0 udp
    service-group SSLi_UDP_0
    use-rcv-hop-for-resp
    redirect-fwd ethernet 2
    no-dest-nat
port 0 others
    service-group SSLi_UDP_0
    use-rcv-hop-for-resp
    redirect-fwd ethernet 2
    no-dest-nat
port 443 https
    service-group SSLi_TCP_443
    use-rcv-hop-for-resp
    redirect-fwd ethernet 2
    template client-ssl cl_ssl
    no-dest-nat port-translation
!
slb virtual-server SSLi_out_ingress 0.0.0.0 acl 192
user-tag Security,ssli_out
port 0 tcp
    service-group GW_TCP_0
    use-rcv-hop-for-resp
    redirect-rev ethernet 3
    no-dest-nat
port 0 udp
    service-group GW_UDP_0
    use-rcv-hop-for-resp
```

```
    redirect-rev ethernet 3
    no-dest-nat
port 0 others
    service-group GW_UDP_0
    use-rcv-hop-for-resp
    redirect-rev ethernet 3
    no-dest-nat
port 443 tcp
    service-group GW_TCP_0
    use-rcv-hop-for-resp
    redirect-rev ethernet 3
    no-dest-nat
port 8080 http
    service-group GW_TCP_8080
    use-rcv-hop-for-resp
    redirect-rev ethernet 3
    template server-ssl sr_ssl
    no-dest-nat port-translation
!
end
```

Outbound Static Port Type HTTPS

This section provides instructions on configuring SSLi by using an example configuration of an outbound SSLi with a static port type HTTPS deployment. To implement the configuration you can use both the GUI and CLI configuration examples explained in this section.

Although A10 Networks supports various SSLi deployments based on different SSLi features, the overall steps for configuring SSLi for each deployment are similar.

NOTE: Subsequent sections refer to the procedures documented for [Two ACOS Devices, Each With Single Partition Deployment](#). It is recommended that you understand the workflow described in this section, even if your SSLi deployment differs from this example.

The following topics are covered:

Prerequisites	87
Two ACOS Devices, Each With Single Partition Deployment	87
Single ACOS Device With Two Partitions Deployment	113
Single vThunder Device With Two Partitions Deployment	114

Prerequisites

- A10 Networks Advanced Core Operating System (ACOS®) 4.0.1 SP9 or higher. ACOS version 4.1.0 or higher is recommended.
- For single-partition SSLi deployments, ACOS version 4.1.1 or higher is required.
- Supported A10 Thunder or vThunder device(s). For more information on the supported ACOS devices for deploying SSLi, refer to the [SSLi Technical Specifications](#).
- Security appliance or ICAP-based (RFC3507) anti-virus or DLP solution.
- A self-signed certificate or a certification authority (CA) certificate with a known private key.

NOTE: If not already provisioned, push an internal PKI CA root certificate to all the client machines.

- The ACOS device supports both CLI and GUI for configuration. Change the default management port IP address for GUI or CLI access.
- If you are using two separate ACOS devices to deploy SSLi, make sure that both systems are configured with management addresses. For more information on how to access an ACOS device, refer to *System Configuration and Administration Guide*.
- Unless you are using a single ACOS device with a single partition to deploy SSLi, you require two partitions, one to decrypt SSL traffic and the second to encrypt SSL traffic. Make sure that you are on the correct partition when creating configurations.
- In a single device solution, use the command `system ve-mac-scheme system-mac` to support MAC address duplication .

Two ACOS Devices, Each With Single Partition Deployment

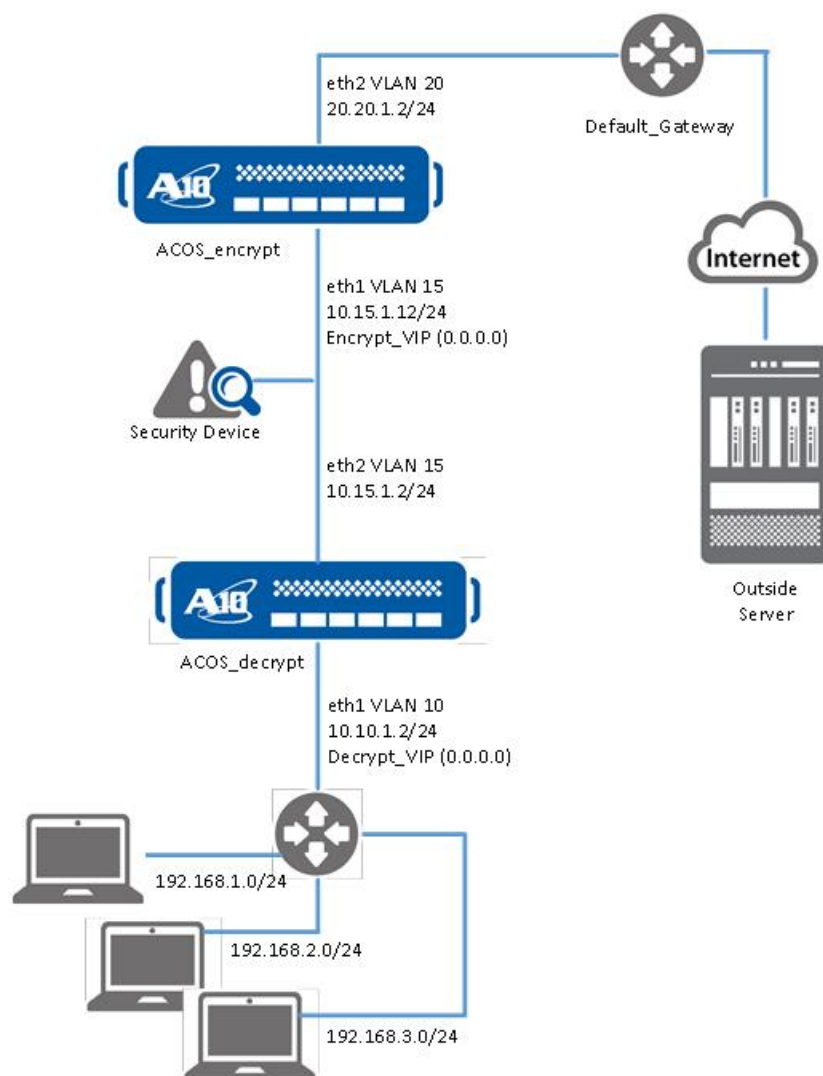
In a static-port type deployment, each intercepted protocol is configured with its own static virtual port enabled for SSLi. For example, to intercept SMTP running over SSL, the wildcard VIP configuration includes the command line **port 25 ssli** where 25 is

the port number identifying SMTP. For static port type SSLi deployment configured to intercept HTTPS traffic, the wildcard VIP includes the command line **port 443 https** where port 443 is the port number identifying HTTPS. In such deployments, only the traffic for the specified protocol is intercepted. All other SSL and non-SSL traffic is bypassed.

You can configure static port inspection for both inbound and outbound traffic. The intercepted and decrypted traffic is said to be outbound when it flows from clients in a private network to the SSL servers on the Internet. If the traffic is intercepted and decrypted as it flows from the Internet to the client network, it is called as inbound. Inbound and outbound SSLi can also be configured together. In such a deployment, traffic flowing in both directions is decrypted and re-encrypted. However, the command lines that configure the inbound virtual servers must go before the command lines that configure the outbound virtual servers.

Static port inspection is supported for all the three types of SSLi deployments.

Figure 12 : Static Port Type HTTPS in a Two ACOS Device each with Single Partition Deployment



The following table provides the VLAN IDs, Virtual Ethernet (VE) addresses, and interfaces used to configure the SSLi network topology illustrated in [Figure 12](#).

Table 10 :

Table 9 : Details of the SSLi Deployment

Partition	Tagged VLAN	VE IP Address	Ethernet Port Number
ACOS_decrypt	10	10.10.1.2 /24	eth1
	15	10.15.1.2 /24	eth2
ACOS_encrypt	20	20.1.1.2 /24	eth2

Table 10 :

Table 9 : Details of the SSLi Deployment

Partition	Tagged VLAN	VE IP Address	Ethernet Port Number
	15	10.15.1.12 /24	eth1

In this example, the outbound SSLi with static-port type HTTPS deployment consists of two ACOS devices, each with a single partition, and the security device set in between. The ACOS devices are in L3 mode, while the security device is in L2 mode.

The encrypted traffic from the client is passed to the ACOS_decrypt partition. The ACOS_decrypt partition decrypts the HTTPS traffic and forwards the clear traffic to the security device. After inspection, the security device passes the clear traffic to the ACOS_encrypt partition. The ACOS_encrypt partition re-encrypts the HTTPS traffic and passes it to the external gateway. All other SSL traffic is bypassed.

CLI Configuration

In order to configure SSLi for two ACOS devices each with a single partition deployment, you must first configure the two partitions, ACOS_decrypt and ACOS_encrypt. Also, for a list of prerequisites, see [Prerequisites](#).

The following topics are covered:

ACOS_decrypt Configuration	90
ACOS_encrypt Configuration	95

ACOS_decrypt Configuration

Perform the following steps for the ACOS_decrypt partition:

Step 1. Configuring the Network VLANs

1. Create tagged VLANs 10 and 15 on the ethernet 1 and ethernet 2 interfaces respectively.

Enable the interface ethernet 1 and 2 by running the following commands:

```
ACOS_decrypt(config)# interface ethernet 1
ACOS_decrypt(config-if:ethernet:1)# enable
ACOS_decrypt(config-if:ethernet:1)# exit
```

```
ACOS_decrypt(config)# interface ethernet 2
ACOS_decrypt(config-if:ethernet:2)# enable
ACOS_decrypt(config-if:ethernet:2)# exit
```

2. Create a tagged VLAN 10. Bind ethernet 1 to the tagged VLAN 10. Also, bind a virtual interface VE 10 to VLAN 10.

```
ACOS_decrypt(config)# vlan 10
ACOS_decrypt(config-vlan:10)# tagged ethernet 1
ACOS_decrypt(config-vlan:10)#router-interface ve 10
ACOS_decrypt(config-vlan:10)# exit
```

3. Create a tagged VLAN 15. Bind ethernet 2 to the tagged VLAN 15. Also, bind a virtual interface VE 15 to VLAN 15.

```
ACOS_decrypt(config)# vlan 15
ACOS_decrypt(config-vlan:15)# tagged ethernet 2
ACOS_decrypt(config-vlan:15)#router-interface ve 15
ACOS_decrypt(config-vlan:15)# exit
```

Step 2. Configuring the Network IP Addresses

On each VE, enable promiscuous VIP support, which is required for wildcard VIPs. When you enable promiscuous VIP support on a VE, the option is automatically enabled on each Ethernet data port associated with the VE. Perform the following steps:

```
ACOS_decrypt(config)# interface ve 10
ACOS_decrypt(config-if:ve10)# ip address 10.10.1.2 /24
ACOS_decrypt(config-if:ve10)# ip allow-promiscuous-vip
ACOS_decrypt(config-if:ve10)# exit
ACOS_decrypt(config)# interface ve 15
ACOS_decrypt(config-if:ve15)# ip address 10.15.1.2 /24
ACOS_decrypt(config-if:ve15)# exit
```

Step 3. Configuring the SSLi Services

1. Create a client SSL template with **forward-proxy enable** configured. This configuration enables the ACOS_decrypt device to proxy for the remote SSL servers and bring up SSL sessions with the clients.

Configure the client SSL template called `SSLInsight_DecryptSide` by running the following commands:

```
ACOS_decrypt(config)# slb template client-ssl SSLInsight_DecryptSide
ACOS_decrypt(config-client ssl)# forward-proxy-ca-certificate
Cert123.pem key key123
ACOS_decrypt(config-client ssl)# forward-proxy-enable
```

NOTE: There already may be a CA Root Certificate installed. If the CA has signed the A10 certificate as a subordinate, the certificate-chaining command is used to make the chain a trusted one.

2. Create a real server called **FW1_Inspect** on ACOS_decrypt. Configure the port 8080 for decrypted SSLi traffic.

```
ACOS_decrypt(config)# slb server FW1_Inspect 10.15.1.12
ACOS_decrypt(config-real server)# port 8080 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
```

3. Configure wildcard ports for all non-HTTPS traffic that is to be bypassed.

```
ACOS_decrypt(config-real server)# port 0 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit
ACOS_decrypt(config-real server)# port 0 udp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit
```

Step 4. Configuring the SSLi Service Groups

1. Configuring the SSLi service groups enable you to manage how the different types of traffic coming from the clients is handled by ACOS_decrypt.
2. Create a service group named **FW1_Inspect_SG** for decrypted SSL traffic. The **FW1_Inspect_SG** service group is configured on **FW1_Inspect** to forward HTTPS assigned over protocol 8080 to the ACOS_encrypt device.

```
ACOS_decrypt(config)# slb service-group FW1_Inspect_SG tcp
ACOS_decrypt(config-slb svc group)# member FW1_Inspect 8080
```

3. For the non-HTTPS traffic that is to be bypassed, configure two other service groups called **ALL_TCP_SG** for TCP and **ALL_UDP_SG** for UDP traffic.

```
ACOS_decrypt(config)# slb service-group ALL_TCP_SG tcp
ACOS_decrypt(config-slb svc group)# member FW1_Inspect 0
ACOS_decrypt(config-slb svc group)# exit
ACOS_decrypt(config)# slb service-group ALL_UDP_SG udp
```

```
ACOS_decrypt(config-slb svc group)# member FW1_Inspect 0
ACOS_decrypt(config-slb svc group)# exit
```

Step 5. Configuring the Virtual Server

A virtual server called **decrypt_vip** is created and is associated to the wildcard outbound VIP to intercept traffic from clients. The following virtual ports are configured on this VIP:

- **443 (HTTPS)**—Intercepts SSL-encrypted traffic from the clients. Port 443 on the wildcard outbound VIP is bound to a service group called **FW1_Inspect_SG** that contains the path through the security device to the ACOS_encrypt device. Consider the following information:
 - The destination NAT is disabled, and ACOS_decrypt does not change the source or destination IP addresses of the traffic.
 - Port translation is enabled and required because the ACOS device must change the destination protocol port from 443 to the port number on which the security device listens for traffic.
 - The client-SSL template is bound to the virtual port 443 HTTPS.
- **0 (TCP), 0 (UDP), and 0 (Others)** —Intercepts the client traffic that is not HTTPS in the following ways:
 - The TCP port intercepts all other TCP traffic from clients. The TCP wildcard port is bound to a TCP service group called **ALL_TCP_SG** that contains the path through the security device to the ACOS_encrypt device.
 - The UDP port intercepts all other UDP traffic from clients. The UDP wildcard port is bound to a UDP service group called **ALL_UDP_SG** that contains the path through the security device to the ACOS_encrypt device.
 - The Others port intercepts the client traffic types that are not listed. The Others port is for IP traffic not included by the TCP and UDP all-ports sections. The Others wildcard port is bound to a UDP service group called **ALL_UDP_SG** that contains the path through the security device to the ACOS_encrypt device.
 - The destination NAT and port translation are disabled for the aforementioned ports.

NOTE: If you replace a certificate and key in a client-SSL or server-SSL template, you must unbind the template from the virtual ports that use it and then rebind the template to the virtual ports.

1. Create an ACL to permit IP traffic from any source to any destination. Create the virtual server **Decrypt_VIP**. Bind the wildcard VIP to the virtual server and associate the ACL with the VIP.

```
ACOS_decrypt(config)# access-list 100 permit ip any any vlan 10
ACOS_decrypt(config)# slb virtual-server Decrypt_VIP 0.0.0.0 acl 100
```

2. Bind the port 443 to the wildcard outbound VIP and associate the port with the service group called **FW1_Inspect_SG** that contains the path through the security device to the ACOS_encrypt device.

```
ACOS_decrypt(config-slb vsrver)# port 443 https
ACOS_decrypt(config-slb vsrver-vport)# service-group FW1_Inspect_SG
ACOS_decrypt(config-slb vsrver-vport)# no-dest-nat port-translation
```

3. Bind the client SSL template to the virtual port.

```
ACOS_decrypt(config-slb vsrver-vport)# template client-ssl
SSLInsight_DecryptSide
ACOS_decrypt(config-slb vsrver-vport)# no-dest-nat port-translation
ACOS_decrypt(config-slb vsrver-vport)# exit
```

4. Configure the virtual server to assign wildcard ports to incoming non-HTTPS traffic and to forward that traffic over the non-HTTPS service groups.

```
ACOS_decrypt(config-slb vsrver)# port 0 tcp
ACOS_decrypt(config-slb vsrver-vport)# service-group ALL_TCP_SG
ACOS_decrypt(config-slb vsrver-vport)# no-dest-nat
ACOS_decrypt(config-slb vsrver-vport)# exit
ACOS_decrypt(config-slb vsrver)# port 0 udp
ACOS_decrypt(config-slb vsrver-vport)# service-group ALL_UDP_SG
ACOS_decrypt(config-slb vsrver-vport)# no-dest-nat
ACOS_decrypt(config-slb vsrver-vport)# exit
ACOS_decrypt(config-slb vsrver)# port 0 other
ACOS_decrypt(config-slb vsrver-vport)# service-group ALL_UDP_SG
ACOS_decrypt(config-slb vsrver-vport)# no-dest-nat
```

ACOS_encrypt Configuration

Perform the following steps for the ACOS_encrypt partition:

Step 1. Configuring the Network VLANs

1. Create tagged VLANs 15 and 20 on the ethernet 1 interface. Perform the following steps:
2. Enable the interface ethernet 1 by running the following commands:

```
ACOS_encrypt(config)# interface ethernet 1
ACOS_encrypt(config-if:ethernet:1)# enable
ACOS_encrypt(config-if:ethernet:1)# exit
ACOS_encrypt(config)# interface ethernet 2
ACOS_encrypt(config-if:ethernet:2)# enable
ACOS_encrypt(config-if:ethernet:2)# exit
```

3. Create a tagged VLAN 20. Bind ethernet 2 to the tagged VLAN 20. Also, bind a virtual interface VE 20 to VLAN 20.

```
ACOS_encrypt(config)# vlan 20
ACOS_encrypt(config-vlan:10)# tagged ethernet 2
ACOS_encrypt(config-vlan:10)#router-interface ve 20
ACOS_encrypt(config-vlan:10)# exit
```

4. Create a tagged VLAN 15. Bind ethernet 1 to the tagged VLAN 15. Also, bind a virtual interface VE 15 to VLAN 15.

```
ACOS_encrypt(config)# vlan 15
ACOS_encrypt(config-vlan:15)# tagged ethernet 1
ACOS_encrypt(config-vlan:15)#router-interface ve 15
ACOS_encrypt(config-vlan:15)# exit
```

Step 2. Configuring the Network IP Addresses

On each VE, enable promiscuous VIP support, which is required for wildcard VIPs. When you enable promiscuous VIP support on a VE, the option is automatically enabled on each Ethernet data port associated with the VE. Perform the following steps:

```
ACOS_encrypt(config)# interface ve 20
ACOS_encrypt(config-if:ve20)# ip address 20.1.1.2 /24
ACOS_encrypt(config-if:ve20)# exit
```

```
ACOS_encrypt(config)# interface ve 15
ACOS_encrypt(config-if:ve15)# ip address 10.15.1.12 /24
ACOS_encrypt(config-if:ve15)# ip allow-promiscuous-vip
ACOS_encrypt(config-if:ve15)# exit
```

Step 3. Configuring the SSLi Services

1. Create an SSL server template called **SSLInsight_EncryptSide** on ACOS_encrypt so that the VIP on ACOS_encrypt can operate as an SSL client and handshake with the **ExternalABC** server. Enable forward proxy services on the template to allow SSLi operation on the VIP.

```
ACOS_encrypt(config)# slb template server-ssl SSLInsight_EncryptSide
ACOS_encrypt(config-server ssl)# forward-proxy-enable
```

2. Create a real server called **Default_Gateway** on ACOS_encrypt. Configure port 443 for the intercepted HTTPS traffic. ACOS_encrypt forwards the traffic on these ports over VLAN 20 to the default gateway at IP address 20.1.1.10. The default gateway has a route to the **ExternalABC** server.

```
ACOS_encrypt(config)# slb server Default_Gateway 20.1.1.10
ACOS_encrypt(config-real server)# port 443 tcp
ACOS_encrypt(config-real server-node port)# health-check-disable
ACOS_encrypt(config-real server-node port)# exit
```

3. Configure wildcard ports for all non-HTTPS traffic.

```
ACOS_encrypt(config-real server)# port 0 tcp
ACOS_encrypt(config-real server-node port)# health-check-disable
ACOS_encrypt(config-real server-node port)# exit
ACOS_encrypt(config-real server)# port 0 udp
ACOS_encrypt(config-real server-node port)# health-check-disable
ACOS_encrypt(config-real server-node port)# exit
```

Step 4. Configuring the SSLi Service Groups

1. Create a service group called **DG_SSL_SG** and provide a path for the intercepted HTTPS traffic by binding the service group to ports 443 of the real server **Default_Gateway**.

```
ACOS_encrypt(config)# slb service-group DG_SSL_SG tcp
ACOS_encrypt(config-slb svc group)# member Default_Gateway 443
```

2. Create a service group called **DG_TCP_SG** and provide a path to **Default_Gateway**

for all other TCP traffic by binding the service group to the wildcard `port 0 tcp`.

```
ACOS_encrypt(config)# slb service-group DG_TCP_SG tcp
ACOS_encrypt(config-slb svc group)# member Default_Gateway 0
```

3. Create a service group called `DG_UDP_SG` and provide a path to `Default_Gateway` for all UDP traffic by binding the service group to the wildcard `port 0 udp`.

```
ACOS_encrypt(config)# slb service-group DG_UDP_SG udp
ACOS_encrypt(config-slb svc group)# member Default_Gateway 0
```

Step 5. Configuring the Virtual Server

A virtual server called `Encrypt_VIP` is created and is associated to the wildcard VIP to intercept traffic from the security device. The following virtual ports are configured on this VIP:

- **8080 (HTTP)**—Intercepts decrypted client traffic that is allowed by the security devices. Port 8080 is bound to a service group called `DG_SSL_SG` that contains a member for the gateway router to the Internet. This member consists of the router's IP address and protocol port 443. Consider the following information:
 - The destination NAT is disabled, but port translation is enabled.
 - Port translation is required because `ACOS_encrypt` must change the destination protocol port to 443 before sending the re-encrypted traffic to the gateway router.
- **0 (TCP), 0 (UDP), and 0 (Others)**—Intercepts all client traffic that is not SSL-encrypted traffic in the following ways:
 - The TCP port intercepts all other TCP traffic from clients. The TCP port is bound to a TCP service group called `DG_TCP_SG` that contains a member for the gateway router to the Internet.
 - The UDP port intercepts all other UDP traffic from clients.
 - The Others port intercepts client traffic of types other than those listed above. The UDP wildcard port and others wildcard port is bound to a UDP service group called `DG_UDP_SG` that contains a member for the gateway router.
 - The destination NAT and port translation are disabled for the aforementioned ports.

1. Create an ACL to permit IP traffic from any source to any destination for VLAN 15. Create a virtual server called **Encrypt_VIP** and associate the ACL to the virtual server.

```
ACOS_encrypt(config)# access-list 101 permit ip any any vlan 15
ACOS_encrypt(config)# slb virtual-server Encrypt_VIP 0.0.0.0 acl 101
```

2. Bind the port 8080 to the wildcard VIP and associate the port with the service group called **DG_SSL_SG** that contains the path through from ACOS_encrypt to the gateway router.

```
ACOS_encrypt(config-slb vsrver)# port 8080 http
ACOS_encrypt(config-slb vsrver-vport)# no-dest-nat port-translation
ACOS_encrypt(config-slb vsrver-vport)# service-group DG_SSL_SG
```

3. Bind the server SSL template to the virtual port.

```
ACOS_encrypt(config-slb vsrver-vport)# template server-ssl
SSLInsight_EncryptSide
ACOS_encrypt(config-slb vsrver-vport)# exit
```

4. Create wildcard ports for all other traffic. Disable destination NAT to preserve the destination IP address on load-balanced traffic. Bind the wildcard virtual port 0 tcp to the **DG_TCP_SG** service-group. Bind the wildcard virtual port 0 udp to the **DG_UDP_SG** service-group. Bind the wildcard virtual port 0 others to any wildcard service group such as **DG_UDP_SG**.

```
ACOS_encrypt(config-slb vsrver)# port 0 tcp
ACOS_encrypt(config-slb vsrver-vport)# no-dest-nat
ACOS_encrypt(config-slb vsrver-vport)# service-group DG_TCP_SG
ACOS_encrypt(config-slb vsrver-vport)# exit
ACOS_encrypt(config-slb vsrver)# port 0 udp
ACOS_encrypt(config-slb vsrver-vport)# no-dest-nat
ACOS_encrypt(config-slb vsrver-vport)# service-group DG_UDP_SG
ACOS_encrypt(config-slb vsrver-vport)# exit
ACOS_encrypt(config-slb vsrver)# port 0 others
ACOS_encrypt(config-slb vsrver-vport)# no-dest-nat
ACOS_encrypt(config-slb vsrver-vport)# service-group DG_UDP_SG
ACOS_encrypt(config-slb vsrver-vport)# exit
ACOS_encrypt(config-slb vsrver)# exit
```

5. If you provision SSLi on an FTA-enabled ACOS device with any partition that is

deployed in a L2 mode, configure the interfaces by using the `cpu-process` command. For example, to enable ethernet 1, the following steps are applicable:

```
ACOS_decrypt(config)# interface ethernet 1
ACOS_decrypt(config-if:ethernet:1)# enable
ACOS_decrypt(config-if:ethernet:1)# cpu-process
```

GUI Configuration

In order to configure SSLi for a two ACOS device single partition deployment, you must first configure the two partitions.

Also, for a list of prerequisites, see [Prerequisites](#).

The following topics are covered:

ACOS_decrypt Configuration	99
ACOS_encrypt Configuration	106

ACOS_decrypt Configuration

Perform the following steps for the ACOS_decrypt partition:

Step 1. Configuring the Network VLANs

Create tagged VLANs 10 and 15 on the ethernet 1 and ethernet 2 interfaces respectively.

To create VLAN 10, perform the following steps:

1. Navigate to **Network > Interfaces > LAN**.
2. Click **Edit** in the Actions column for interface 1 (Interface field).
3. On the **Update Ethernet** page, select **Enable** in the **Status** field.
4. Click **Update**.
5. Navigate to **Network > VLAN**.
6. Click **+ Create**.
7. Enter 10 in the **VLAN ID** field.
8. Click the checkbox in the **Create Virtual Interface** field.

9. Select **1** from the list of interfaces in the **Tagged Ethernet** field
10. Click **Create VLAN**.
11. Repeat the steps to create VLAN 15 for interface 2.

Step 2. Configuring the Network IP Addresses

Configure the parameters for VE 10 by performing the following steps:

1. Navigate to **Network > Interfaces > Virtual Ethernets**.
2. Click **Edit** in the **Actions** column for virtual interface (ve)**10** (ifnum field).
3. Enter **10.10.1.2** in the **IPv4 Address** field
4. Enter **255.255.255.0** in the **NetMask** field.
5. Click the icon to save the new row.
6. Click the **Allow Promiscuous VIP** box.
7. Click **Update**.
8. Repeat the procedure for the ve **15** parameters, IPv4 address is **10.15.1.2**.

Step 3. Creating an Access List

1. Create an ACL to permit IP traffic from any source to any destination.
2. Click **Security >> Access List**.
3. Click **Create**.
4. The **Create Standard Access List** page is displayed.
5. Enter the details:
6. Access List Number: 100
7. Sequence Number: 1
8. Action: Permit Any for Entry
9. Click **Create** to create ACL 100.

Step 4. Configuring the SSLi Service

In the GUI configuration, the red asterisk (*) indicates a required parameter. Some required parameters are filled in automatically, while some must be manually configured. Before attempting to create an SSLi service, the CA certificate you

import and upon which your proxied certificates are based, must be imported. In the CLI, the `import cert` command imports certificates that can be used in the SSLi service.

NOTE: This example of GUI configuration covers only the SSLi VIP and all the other SSL ACOS objects that are needed for the basic **static-port https 443** configuration. For a complete list of available options and their associated descriptions, refer to the Online Help for the ACOS GUI.

1. Navigate to **Security > SSLi > Services > +Create** and click **+Create**.

The **Create SSLi Service** page is displayed.

2. Enter the following details:
3. **Type:** Inside(Decrypt)
4. **Name:** SSLInsight_DecryptSide
5. Enable static port.
6. Click **Next**.
7. Under **Basic**, select Forward Proxy Enable.
8. Under **SSLi proxy**, select the CA cert and Key.
9. Click **Next**.
10. Continue clicking **Next** till you get to the end page and then click **Done**.

Step 5. Configuring the Real Server

Create a real server called FW1_Inspect on ACOS_decrypt. Configure the port 8080 for decrypted SSLi traffic. Configure wildcard ports for all non-HTTPS traffic that is to be bypassed.

1. Go to **Security >> SSLi >> Servers**.
2. Click **Create**.
3. The **Create Server** page is displayed.
4. Enter the following details:

- **Name:** FW1_Inspect
 - **Type:** IPv4
 - **Host:** 10.15.1.12
5. Click **Add Port**, and enter the following details:
 - **Port:** 8080
 - **Protocol:** TCP
 6. Click **Apply** under Actions.
 7. Click **Add Port**, and enter the following details:
 - **Port:** 0
 - **Protocol:** TCP
 8. Click **Apply** under Actions.
 9. Click **Add Port**, and enter the following details:
 - **Port:** 0
 - **Protocol:** UDP
 10. Click **Apply** under Actions.
 11. Click **OK** to create the server FW1_Inspect.

Step 6 Creating the Service Group and its Members

Create a service group named **FW1_Inspect_SG** for decrypted SSL traffic. The **FW1_Inspect_SG** service group is configured on **FW1_Inspect** real server to forward HTTPS assigned over protocol **8080** to the **ACOS_encrypt** device. The real server is already created in the previous section.

1. Go to **Security >> SSLi >> Service Group**.
2. Click **Create**.
3. The **Create Service Group** page is displayed.
4. Enter the following details to create a service group named **FW1_Inspect_SG**:
 - **Name:** FW1_Inspect_SG
 - **Protocol:** TCP

5. Under **Members**, click **Add Member**.
6. The **Create Member** page is displayed.
7. Select **Existing**.
8. Select **FW1_Inspect** from the drop-down under **Name** and enter the **Port: 8080**.
9. Click **Apply**.

The member **FW1_Inspect** is added to the service group.

10. Click **Create** again to create the Service Group.
11. For the non-HTTPS traffic that is to be bypassed, configure two other service groups called **ALL_TCP_SG** for TCP and **ALL_UDP_SG** for UDP traffic.
12. Go to **Security >> SSLi >> Service Group**.
13. Click **Create**.
14. The **Create Service Group** page is displayed.
15. Enter the following details to create a service group named **ALL_TCP_SG**.
 - **Name:** ALL_TCP_SG
 - **Protocol:** TCP
16. Under **Members**, click **Add Member**.
17. The **Create Member** page is displayed.
18. Select **Existing**.
19. Select **FW1_Inspect** from the drop-down under **Name** and enter the following details:
20. Port: **0**
21. Click **Apply**.
22. The member **FW1_Inspect** is added to the service group.
23. Click **Create** again to create the Service Group.
24. To create the service group, **ALL_UDP_SG**, here are the details:

- **Name:** ALL_UDP_SG
- **Protocol:** UDP

25. Under **Members**, click **Add Member**.
26. The **Create Member** page is displayed.
27. Select **Existing**.
28. Select **FW1_Inspect** from the drop-down under **Name** and Port **0**.
29. Click **Apply**.

The member **FW1_Inspect** is added to the service group.

30. Click **Create** again to create the Service Group.

Step 7. Creating the Virtual Server

A virtual server called Decrypt_VIP is created and is associated to the wildcard outbound VIP to intercept traffic from clients. The following virtual ports are configured on this VIP:

- **443 (HTTPS)**—Intercepts SSL-encrypted traffic from the clients. Port 443 on the wildcard outbound VIP is bound to a service group called FW1_Inspect_SG that contains the path through the security device to the ACOS_encrypt device. Consider the following information:
 - The destination NAT is disabled, and ACOS_decrypt does not change the source or destination IP addresses of the traffic.
 - Port translation is enabled and required because the ACOS device must change the destination protocol port from 443 to the port number on which the security device listens for traffic.
 - The client-SSL template is bound to the virtual port 443 HTTPS.
- **0 (TCP), 0 (UDP), and 0 (Others)** —Intercepts the client traffic that is not HTTPS in the following ways:
 - The TCP port intercepts all other TCP traffic from clients. The TCP wildcard port is bound to a TCP service group called ALL_TCP_SG that contains the path through the security device to the ACOS_encrypt device.

- The UDP port intercepts all other UDP traffic from clients. The UDP wildcard port is bound to a UDP service group called ALL_UDP_SG that contains the path through the security device to the ACOS_encrypt device.
- The Others port intercepts the client traffic types that are not listed. The Others port is for IP traffic not included by the TCP and UDP all-ports sections. The Others wildcard port is bound to a UDP service group called ALL_UDP_SG that contains the path through the security device to the ACOS_encrypt device.
- The destination NAT and port translation are disabled for the aforementioned ports.

Perform the following steps:

1. Go to **ADC >> SLB >> Virtual Servers**.
2. Click **Create**.

The **Create Virtual Server** page is displayed.

3. Enter the **Name**: Decrypt_VIP.
4. Enable **Wildcard**.
5. Under Virtual Port, click **Create**.

The **Create Virtual Port** page is displayed.

6. Enter the following details:
 - **Protocol**: HTTPS
 - **Port**: 443
 - **Service Group**: FW1_Inspect_SG
 - **Template Client SSL**: SSLInsight_DecryptSide
7. Enable No Dest Nat and Port Translation.
8. Click **Create**.
9. Click **Create** to add another virtual port.

The **Create Virtual Port** page is displayed.

10. Enter the following details:

- **Protocol:** TCP
- **Port:** 0
- **Service Group:**ALL_TCP_SG

11. Click **Create**.

12. Click **Create** to add another virtual port.

The **Create Virtual Port** page is displayed.

13. Enter the following details:

- **Protocol:** UDP
- **Port:** 0
- **Service Group:**ALL_UDP_SG

14. Click **Create**.

15. Click **Create** to add another virtual port.

The **Create Virtual Port** page is displayed.

16. Enter the following details:

- **Protocol:** Other
- **Port:** 0
- **Service Group:**ALL_UDP_SG

17. Click **Create**.

ACOS_encrypt Configuration

Perform the following steps for the ACOS_encrypt partition:

Step 1. Configuring the Network VLANs

Create tagged VLANs 15 and 20 on the ethernet 1 interface and ethernet interface 2 respectively.

Follow the instructions in [Step 1. Configuring the Network VLANs](#).

Step 2. Configuring the Network IP Addresses

Assign IP address 20.1.1.2 to ve 20 and IP address 10.15.1.12 to ve 15 respectively. Follow the instructions in [Step 2. Configuring the Network IP Addresses](#).

Step 3. Configuring an Access List.

Create an ACL to permit IP traffic from any source to any destination for VLAN 15. Create a virtual server called Encrypt_VIP and associate the ACL to the virtual server. Follow the instructions in [Step 3. Creating an Access List](#).

Step 4. Configuring SSLi Services

Create an SSLi service called SSLInsight_EncryptSide. Enable forward proxy enable. Follow the instructions in [Step 4. Configuring the SSLi Service](#).

Step 5. Configuring the Real Server

Create a real server called Default_Gateway on ACOS_encrypt. Configure port 443 for the intercepted HTTPS traffic. ACOS_encrypt forwards the traffic on these ports over VLAN 20 to the default gateway at IP address 20.1.1.10. The default gateway has a route to the ExternalABC server. Follow the instructions in [Step 5. Configuring the Real Server](#).

Step 6. Configuring the Service Groups

- Create a service group called DG_SSL_SG and provide a path for the intercepted HTTPS traffic by binding the service group to ports 443 of the real server Default_Gateway.
- Create a service group called DG_TCP_SG and provide a path to Default_Gateway for all other TCP traffic by binding the service group to the wildcard port 0 tcp.
- Create a service group called DG_UDP_SG and provide a path to Default_Gateway for all UDP traffic by binding the service group to the wildcard port 0 udp.

Follow the instructions in [Step 6 Creating the Service Group and its Members](#).

Step 7. Creating the Virtual Server

A virtual server called `Encrypt_VIP` is created and is associated to the wildcard VIP to intercept traffic from the security device. The following virtual ports are configured on this VIP:

- 8080 (HTTP)—Intercepts decrypted client traffic that is allowed by the security devices. Port 8080 is bound to a service group called `DG_SSL_SG` that contains a member for the gateway router to the Internet. This member consists of the router's IP address and protocol port 443. Consider the following information:
 - The destination NAT is disabled, but port translation is enabled.
 - Port translation is required because `ACOS_encrypt` must change the destination protocol port to 443 before sending the re-encrypted traffic to the gateway router.
- 0 (TCP), 0 (UDP), and 0 (Others)—Intercepts all client traffic that is not SSL-encrypted traffic in the following ways:
 - The TCP port intercepts all other TCP traffic from clients. The TCP port is bound to a TCP service group called `DG_TCP_SG` that contains a member for the gateway router to the Internet.
 - The UDP port intercepts all other UDP traffic from clients.
- The Others port intercepts client traffic of types other than those listed above. The UDP wildcard port and others wildcard port is bound to a UDP service group called `DG_UDP_SG` that contains a member for the gateway router.
- The destination NAT and port translation are disabled for the aforementioned ports.

Follow the instructions in [Step 7. Creating the Virtual Server](#).

Consolidated Configuration

The configuration developed in the preceding section is the basic building block for other SSLi features. It is referred to as the reference configuration for Static-Port SSLi.

Use the `show running-config` command to check your configuration for both `ACOS_decrypt` and `ACOS_encrypt`.

```
ACOS_decrypt# show running-config
!
access-list 100 permit ip any any vlan 10
!
vlan 10
    tagged ethernet 1
    router-interface ve 10
!
vlan 15
    tagged ethernet 2
    router-interface ve 15
!
hostname ACOS_decrypt
!
interface ethernet 1
    enable
!
interface ethernet 2
    enable
!
interface ve 10
    ip address 10.10.1.2 255.255.255.0
    ip allow-promiscuous-vip
!
interface ve 15
    ip address 10.15.1.2 255.255.255.0
!
slb server FW1_Inspect 10.15.1.12
    port 8080 tcp
        health-check-disable
    port 0 tcp
        health-check-disable
    port 0 udp
        health-check-disable
!
slb service-group ALL_TCP_SG tcp
    member FW1_Inspect 0
!
slb service-group ALL_UDP_SG udp
```

```
member FW1_Inspect 0
!
slb service-group FW1_Inspect_SG tcp
  member FW1_Inspect 8080
!
slb template client-ssl SSLInsight_DecryptSide
  forward-proxy-ca-certificate Cert123.pem key key123
  forward-proxy-enable
!
slb virtual-server Decrypt_VIP 0.0.0.0 acl 100
  port 443 https
    service-group FW1_Inspect_SG
    template client-ssl SSLInsight_DecryptSide
    no-dest-nat port-translation
  port 0 tcp
    service-group ALL_TCP_SG
    no-dest-nat
  port 0 udp
    service-group ALL_UDP_SG
    no-dest-nat
  port 0 others
    service-group ALL_UDP_SG
    no-dest-nat
!
end

ACOS_encrypt# show running-config
!
access-list 101 permit ip any any vlan 15
!
vlan 20
  tagged ethernet 2
  router-interface ve 20
!
vlan 15
  tagged ethernet 1
  router-interface ve 15
!
hostname ACOS_encrypt
!
```

```
interface ethernet 1
  enable
!
interface ethernet 2
  enable
!
interface ve 20
  ip address 20.1.1.2 255.255.255.0
!
interface ve 15
  ip address 10.15.1.12 255.255.255.0
  ip allow-promiscuous-vip
!
slb server Default_Gateway 20.1.1.10
  port 443 tcp
    health-check-disable
  port 0 tcp
    health-check-disable
  port 0 udp
    health-check-disable
!
slb service-group DG_SSL_SG tcp
  member Default_Gateway 443

slb service-group DG_TCP_SG tcp
  member Default_Gateway 0

slb service-group DG_UDP_SG udp
  member Default_Gateway 0
!
slb template server-ssl SSLInsight_EncryptSide
  forward-proxy-enable
!
slb virtual-server Encrypt_VIP 0.0.0.0 acl 101
!
  port 8080 http
    no-dest-nat port-translation
    service-group DG_SSL_SG
    template server-ssl SSLInsight_EncryptSide
  port 0 tcp
```

```
no-dest-nat
service-group DG_TCP_SG
port 0 udp
no-dest-nat
service-group DG_UDP_SG
port 0 others
no-dest-nat
service-group DG_UDP_SG
!
end
```

Checking the Status and Operation

Run the **show slb ssl-forward-proxy-cert** command to check the status and operation of ACOS_decrypt.

```
ACOS_decrypt# show slb ssl-forward-proxy-cert Decrypt_VIP 443 all
Virtual server (inside: 443):

----Start One Certificate---
real server: 74.6.143.25:443 tcp
server name: yahoo.com
ALPN protocol: ALPN HTTP2
state: ready
hit times: 1
idle time: 3 seconds
timeout after 3597 seconds
expires after 604787 seconds
----End One Certificate---

1 hashed certificate retrieved
```

- Run the **show slb ssl-forward-proxy-stats** command to check the SSLi counters such as the certificates created and expired, hit times, idle times, the SSL connections that were inspected and those that were bypassed.
- Run the **clear slb ssl-forward-proxy-cert** command to reset the **ssl-forward-proxy-cert** counters.

Single ACOS Device With Two Partitions Deployment

You can implement SSLi in a single device by creating a separate partition for ACOS_decrypt and ACOS_encrypt. The deployment architecture and the flow of traffic is similar to that of [Two ACOS Devices, Each With Single Partition Deployment](#).

CLI Configuration

To configure SSLi for a single device two partition deployment, perform the following steps:

Follow the prerequisites discussed in [Prerequisites](#).

1. To avoid a duplicate MAC address because of the VLAN that is shared, add the global command of `system ve-mac-scheme system-mac` in the shared partition:

```
ACOS(config)# system ve-mac-scheme system-mac
```

2. Create the ACOS_decrypt and ACOS_encrypt partitions by running the following commands:

```
ACOS(config)# partition ACOS_encrypt id 1 application-type adc
ACOS(config-partition: ACOS_encrypt)# exit
ACOS(config)# active-partition ACOS_encrypt
ACOS[ACOS_encrypt](config)#
ACOS[ACOS_encrypt](config)# active-partition shared
ACOS(config)# partition ACOS_decrypt id 2 application-type adc
ACOS[ACOS_decrypt](config)#
```

3. Bind the VLANs as shown in [Step 1. Configuring the Network VLANs](#) and continue with the remaining steps shown in [CLI Configuration](#).

GUI Configuration

To configure SSLi for a single device two partition deployment, perform the following steps:

Follow the prerequisites discussed in [Prerequisites](#).

To create the ACOS_decrypt and ACOS_encrypt partitions, perform the following steps:

1. Navigate to **System >> Admin Partitions**.
2. Click **Create+**.
3. Specify **ACOS_encrypt** for Partition Name and **1** for the Partition ID.
4. Specify **ADC** for the Type.
5. Enable **Shared VLAN**.
6. Repeat the preceding steps for the ACOS_decrypt partition.
7. Continue with the configuration steps shown in [GUI Configuration](#).

Single vThunder Device With Two Partitions Deployment

The vThunder instance can run in promiscuous mode or non-promiscuous mode. By default, vThunder runs in non-promiscuous mode in order to help optimize system performance. However, the following limitations apply when running vThunder in non-promiscuous mode:

- VE interfaces can be bound to only 1 tagged or untagged physical interface.
- VE MAC address assignment scheme changes are not supported.

The two-partition configuration for SSLi requires VE MAC address assignment changes, and vThunder does not support VE MAC address assignment scheme changes in non-promiscuous mode. Therefore, run the vThunder instance in promiscuous mode. Perform the following steps:

1. To change the vThunder mode to promiscuous mode, use the following command:

```
ACOS(config)# system promiscuous-mode
Settings will take effect on reload. Please save the configuration by
issuing the "write
memory" command followed by the "reload" command
ACOS(config)# write memory
Building configuration...
Write configuration to primary default startup-config
```

```
[OK]
ACOS(config)# exit
ACOS# exit
WARNING: System configuration has been modified
```

2. When the reload completes, enter the following command to permit VE MAC address assignment scheme changes:

```
ACOS# config
ACOS(config)# system ve-mac-scheme system-mac
```

3. Create the ACOS_decrypt and ACOS_encrypt partitions by running the following commands:

```
ACOS(config)# partition ACOS_encrypt id 1 application-type adc
ACOS(config-partition:ACOS_encrypt)# exit
ACOS(config)# active-partition ACOS_encrypt
ACOS[ACOS_encrypt](config)#
ACOS[ACOS_encrypt](config)# active-partition shared
ACOS(config)# partition ACOS_decrypt id 2 application-type adc
ACOS[ACOS_decrypt](config)#
```

4. Bind the VLANs as shown in [Step 1. Configuring the Network VLANs](#) and continue with the remaining steps shown in [CLI Configuration](#).

Outbound Static Port Type STARTTLS

This section describes how to configure outbound SSLi for static port type STARTTLS. Inbound and outbound SSLi can be configured together. In such a deployment, traffic flowing in both directions is decrypted and re-encrypted. However, the command lines that configure the inbound virtual servers must go before the command lines that configure the outbound virtual servers.

The following topics are covered:

Deployment Example	117
CLI Configuration	119
Consolidated Configuration Examples	134

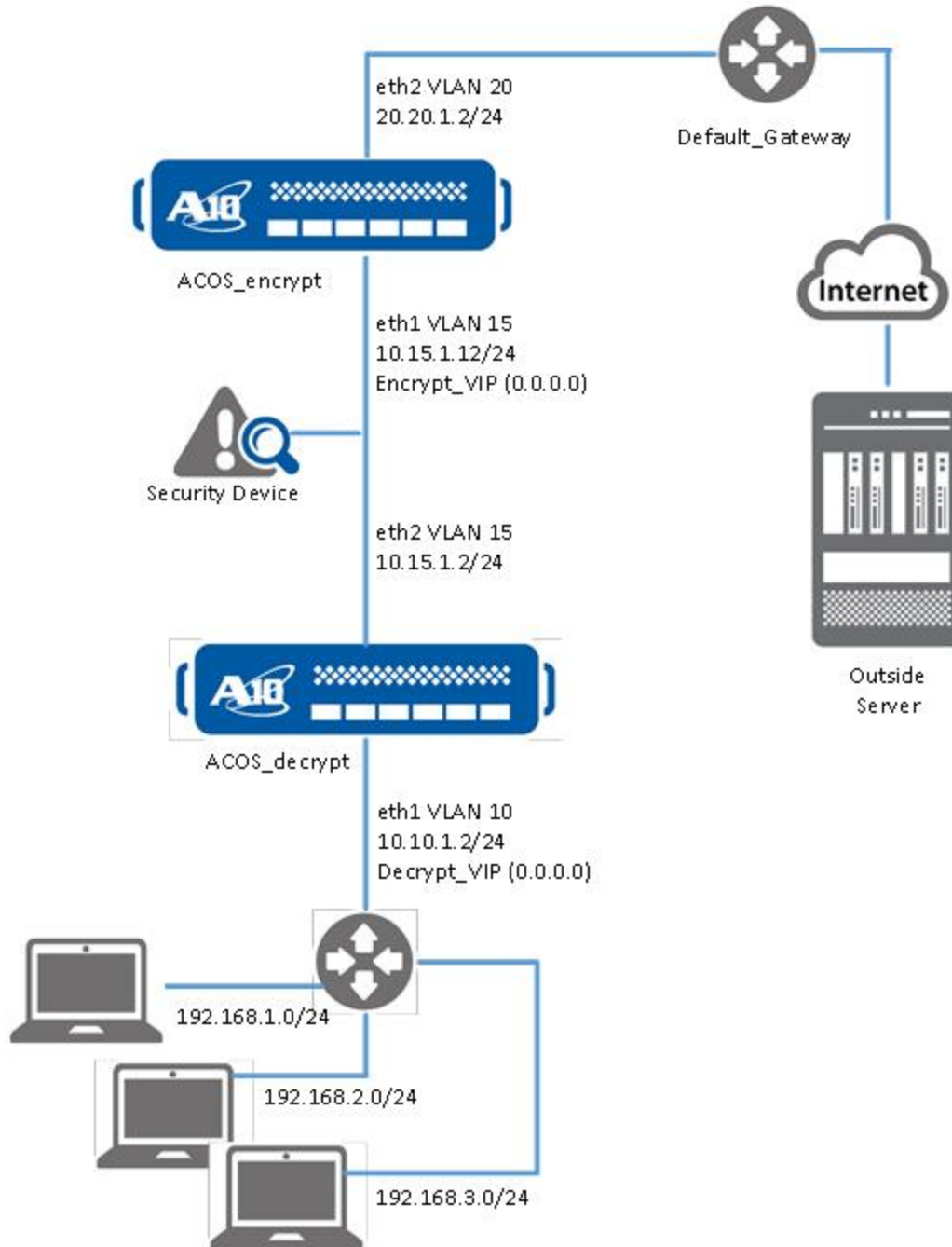
Deployment Example

All the SSLi deployments discussed in [SSLi-Deployments](#) supports static port inspection. The SSLi deployment for static port type STARTTLS intercepts SMTP, POP, FTP, LDAP, and XMPP sessions. The `port port-number ssl` command is used to specify the virtual ports. The term `ssl` indicates that the port is a STARTTLS type. In addition, each STARTTLS type port is defined in an SLB SSLi template which is bound to an SSLi port with the keyword `type`.

In static port type SSLi, each intercepted protocol is configured with its own static virtual port enabled for SSLi. For example, to intercept SMTP running over SSL, the wildcard VIP configuration includes the command line `port 25 ssl` where 25 is the port number identifying SMTP.

In this example, the outbound SSLi with static port type STARTTLS deployment consists of two ACOS devices, each with a single partition, and the security device set in between. The ACOS devices are in L2 mode, while the security device is in L3 mode. In this example, SSLi intercepts SMTP, POP, FTP, LDAP, and XMPP sessions that are running over SSL.

Figure 13 : Static Port Type STARTLS in a Two-Device Deployment, Each with Single Partition



The following is the traffic flow:

- The client's encrypted traffic is sent to the ACOS_decrypt partition.
- The ACOS_decrypt partition decrypts STARTTLS communication and routes it to the security device.
- After inspecting the clear traffic, the security device sends it to the ACOS_encrypt partition.
- The ACOS_encrypt partition re-encrypts HTTPS traffic before sending it to the external gateway.

All other HTTPS traffic is routed around it.

The following table provides the VLAN IDs, Virtual Ethernet (VE) addresses, and interfaces used to configure the SSLi network topology illustrated in [Figure 13](#).

Table 12 :

Table 11 : SSLi Deployment Details

Partition	Tagged VLAN	VE IP Address	Ethernet Port Number
ACOS_decrypt	10	10.10.1.2 /24	eth 1
	15	10.15.1.2 /24	eth 2
ACOS_encrypt	20	20.1.1.2 /24	eth 2
	15	10.15.1.12 /24	eth 1

CLI Configuration

This section describes the CLI configuration examples to configure outbound SSLi for static port type STARTTLS.

To configure SSLi for a two ACOS device single partition deployment, you must first configure the two partitions, ACOS_decrypt and ACOS_encrypt. For more information, see [Prerequisites](#).

The following topics are covered:

ACOS_decrypt Configuration	120
ACOS_encrypt Configuration	127

ACOS_decrypt Configuration

Perform the steps for the ACOS_decrypt partition:

The following topics are covered:

Step 1. Configuring the Network VLANs	120
Step 2. Configuring the Network IP Addresses	121
Step 3. Configuring the SSLi Services	121
Step 4. Configuring the SSLi Service Groups	124
Step 5. Configuring the Virtual Server	125

Step 1. Configuring the Network VLANs

For an explanation of the procedure, refer to a similar procedure discussed in [Step 1. Configuring the Network VLANs](#).

```
ACOS(config)# interface ethernet 1
ACOS(config-if:ethernet:1)# enable
ACOS(config-if:ethernet:1)# exit
!
ACOS(config)# interface ethernet 2
ACOS(config-if:ethernet:2)# enable
ACOS(config-if:ethernet:2)# exit
!
ACOS(config)# hostname ACOS_decrypt
ACOS_decrypt(config)# vlan 10
ACOS_decrypt(config-vlan:10)# tagged ethernet 1
ACOS_decrypt(config-vlan:10)#router-interface ve 10
ACOS_decrypt(config-vlan:10)# exit

ACOS_decrypt(config)# vlan 15
ACOS_decrypt(config-vlan:15)# tagged ethernet 2
ACOS_decrypt(config-vlan:15)#router-interface ve 15
ACOS_decrypt(config-vlan:15)# exit
```


Step 2. Configuring the Network IP Addresses

For an explanation of the procedure, refer to a similar procedure discussed in [Step 2. Configuring the Network IP Addresses](#).

```
ACOS_decrypt(config)# interface ve 10
ACOS_decrypt(config-if:ve10)# ip address 10.10.1.2 /24
ACOS_decrypt(config-if:ve10)# ip allow-promiscuous-vip
ACOS_decrypt(config-if:ve10)# exit
ACOS_decrypt(config)# interface ve 15
ACOS_decrypt(config-if:ve15)# ip address 10.15.1.2 /24
ACOS_decrypt(config-if:ve15)# exit
```

Step 3. Configuring the SSLi Services

1. Configure an SSLi client template, by running the following commands.

```
ACOS_decrypt(config)# slb template client-ssl SSLInsight_DecryptSide
ACOS_decrypt(config-client ssl)# forward-proxy-ca-certificate
Cert123.pem key key123
ACOS_decrypt(config-client ssl)# forward-proxy-enable
ACOS_decrypt(config-client ssl)# exit
```

- a. Create an SSLi client template to configure explicit FTPS for TLS data transmission.

```
ACOS_decrypt(config)# slb template client-ssl Explicit_FTPS
ACOS_decrypt(config-client ssl)# ssli-logging all
ACOS_decrypt(config-client ssl)# close-notify
ACOS_decrypt(config-client ssl)# forward-proxy-ca-certificate
Cert123.pem key key123
ACOS_decrypt(config-client ssl)# forward-proxy-enable
ACOS_decrypt(config-client ssl)# exit
```

- b. Create an SSLi client template to configure implicit FTPS for TLS data transmission, DSCP, and enable ssli FTP.

```
ACOS_decrypt(config)# slb template client-ssl Implicit_FTPS
ACOS_decrypt(config-client ssl)# ssli-logging all
ACOS_decrypt(config-client ssl)# close-notify
ACOS_decrypt(config-client ssl)# forward-proxy-ca-certificate
Cert123.pem key key123
ACOS_decrypt(config-client ssl)# forward-proxy-decrypted dscp 10 2
ACOS_decrypt(config-client ssl)# forward-proxy-enable
ACOS_decrypt(config-client ssl)# enable-ssli-ftp-alg 990
ACOS_decrypt(config-client ssl)# exit
```

NOTE: There already may be a CA Root Certificate installed. If the CA has signed the A10 certificate as a subordinate, the certificate-chaining command is used to make the chain a trusted one.

2. Configure a real server called **FW1_Inspect** with the IP address **10.15.1.12**. This IP address matches the virtual IP address of ACOS_decrypt so that the real server connects to ACOS_decrypt over VLAN 15. Bind **FW1_Inspect** interface to the following ports:

- TCP ports 25 to forward decrypted SMTP traffic.

```
ACOS_decrypt(config)# slb server FW1_Inspect 10.15.1.12
ACOS_decrypt(config-real server)# port 25 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit
ACOS_decrypt(config-real server)# exit
```

- TCP ports 110 to forward decrypted POP traffic.

```
ACOS_decrypt(config-real server)# port 110 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit
ACOS_decrypt(config-real server)# exit
```

- TCP ports 5522 to forward decrypted XMPP traffic.

```
ACOS_decrypt(config-real server)# port 5522 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit
ACOS_decrypt(config-real server)# exit
```

- TCP ports 21 to forward decrypted FTP traffic.

```
ACOS_decrypt(config-real server)# port 21 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit
ACOS_decrypt(config-real server)# exit
```

- TCP ports 389 to forward decrypted LDAP traffic.

```
ACOS_decrypt(config-real server)# port 389 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit
ACOS_decrypt(config-real server)# exit
```

- UDP and TCP wildcard ports 0 to forward all other decrypted UDP and TCP traffic.

```
ACOS_decrypt(config-real server)# port 0 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit

ACOS_decrypt(config-real server)# port 0 udp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit
```

NOTE: You can configure ACOS_decrypt to bypass the security devices based on the website category, client authentication, or the domain SNI (Service Name Indication). For more information, see the relevant section for the specific SSLi feature.

3. Create an SSLi template for each non-HTTP protocol (listed below) running over SSL that ACOS_decrypt must intercept.

- SSLi template for SMTP.

```
ACOS_decrypt(config)# slb template ssli smtp_insight
ACOS_decrypt(config-ssli)# type smtp
ACOS_decrypt(config-ssli)# exit
```

- SSLi template for POP.

```
ACOS_decrypt(config)# slb template ssli pop_insight
ACOS_decrypt(config-ssli)# type pop
ACOS_decrypt(config-ssli)# exit
```

- SSLi template for XMPP.

```
ACOS_decrypt(config)# slb template ssli xmpp_insight
ACOS_decrypt(config-ssli)# type xmpp
ACOS_decrypt(config-ssli)# exit
```

- SSLi template for FTP.

```
ACOS_decrypt(config)# slb template ssli ftp_insight
ACOS_decrypt(config-ssli)# type ftp
ACOS_decrypt(config-ssli)# exit
```

- SSLi template for LDAP.

```
ACOS_decrypt(config)# slb template ssli ldap_insight
ACOS_decrypt(config-ssli)# type ldap
ACOS_decrypt(config-ssli)# exit
```

Step 4. Configuring the SSLi Service Groups

For an explanation of the procedure, refer to a similar procedure discussed in [Step 4. Configuring the SSLi Service Groups](#).

1. Configure one service-group **FW1_Inspect** and associate ports 25, 5522, and 110 to inspect SMTP, POP, and XMPP traffic. You can create separate service-groups for each ports as well.

```
ACOS_decrypt(config)# slb service-group FW1_Inspect_SG tcp
ACOS_decrypt(config-slb svc group)# member FW1_Inspect 25
ACOS_decrypt(config-slb svc group)# member FW1_Inspect 5522
ACOS_decrypt(config-slb svc group)# member FW1_Inspect 110
ACOS_decrypt(config-slb svc group)# exit
```

2. Configure one service-group **FW1_Inspect_FTP** and associate member port 21 to inspect FTP traffic.

```
ACOS_decrypt(config)# slb service-group FW1_Inspect_FTP tcp
ACOS_decrypt(config-slb svc group)# member FW1_Inspect 21
```

3. Configure one service-group **FW1_Inspect_LDAP** and associate member port 389

to inspect LDAP traffic.

```
ACOS_decrypt(config)# slb service-group FW1_Inspect_LDAP tcp
ACOS_decrypt(config-slb svc group)# member FW1_Inspect 389
```

4. Configure two service-groups **ALL_TCP_SG** and **ALL_UDP_SG** and associate wildcard member ports 0 to inspect all other UDP or TCP traffic.

```
ACOS_decrypt(config)# slb service-group ALL_TCP_SG tcp
ACOS_decrypt(config-slb svc group)# member FW1_Inspect 0
ACOS_decrypt(config-slb svc group)# exit

ACOS_decrypt(config)# slb service-group ALL_UDP_SG udp
ACOS_decrypt(config-slb svc group)# member FW1_Inspect 0
ACOS_decrypt(config-slb svc group)# exit
```

Step 5. Configuring the Virtual Server

For an explanation of the procedure, refer to a similar procedure discussed in [Step 5. Configuring the Virtual Server](#).

NOTE: If you configure ssli virtual port along with **ssli-sni-hash-enable** on ACOS_decrypt or ACOS_encrypt, then you cannot configure the **source-nat** virtual port on the ACOS_decrypt. However, you can configure it on the ACOS-encrypt.

1. Configure the ACL and SLB virtual server.

```
ACOS_decrypt(config)# access-list 100 permit ip any any vlan 10
ACOS_decrypt(config)# slb virtual-server Decrypt_VIP 0.0.0.0 acl 100
```

2. Configure port 25 SSLi to intercept encrypted SMTP traffic from the clients. Bind the service-group **FW1_Inspect_SG** which has port 25, client-SSL template, and SMTP SSLi template.

```
ACOS_decrypt(config-slb vserver)# port 25 ssli
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat
ACOS_decrypt(config-slb vserver-vport)# service-group FW1_Inspect_SG
ACOS_decrypt(config-slb vserver-vport)# template client-ssl SSLInsight_
DecryptSide
ACOS_decrypt(config-slb vserver-vport)# template ssli smtp_insight
ACOS_decrypt(config-slb vserver-vport)# exit
```

3. Configure port 110 SSLi to intercept encrypted POP traffic from the clients. Bind the service-group **FW1_Inspect_SG** which has port 110, client-SSL template, and POP SSLi template.

```
ACOS_decrypt(config-slb vserver)# port 110 ssli
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat
ACOS_decrypt(config-slb vserver-vport)# service-group FW1_Inspect_SG
ACOS_decrypt(config-slb vserver-vport)# template client-ssl SSLInsight_
DecryptSide
ACOS_decrypt(config-slb vserver-vport)# template ssli pop_insight
ACOS_decrypt(config-slb vserver-vport)# exit
```

4. Configure port 5522 SSLi to intercept encrypted XMPP traffic from the clients. Bind the service-group **FW1_Inspect_SG** which has port 5522, client-SSL template, and XMPP SSLi template.

```
ACOS_decrypt(config-slb vserver)# port 5522 ssli
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat
ACOS_decrypt(config-slb vserver-vport)# service-group FW1_Inspect_SG
ACOS_decrypt(config-slb vserver-vport)# template client-ssl SSLInsight_
DecryptSide
ACOS_decrypt(config-slb vserver-vport)# template ssli xmpp_insight
ACOS_decrypt(config-slb vserver-vport)# exit
```

5. Configure port 389 SSLi to intercept encrypted LDAP traffic from the clients. Bind the service-group **FW1_Inspect_LDAP** with member port 389, client-SSL template, and LDAP SSLi template.

```
ACOS_decrypt(config-slb vserver)# port 389 ssli
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat
ACOS_decrypt(config-slb vserver-vport)# service-group FW1_Inspect_LDAP
ACOS_decrypt(config-slb vserver-vport)# template client-ssl SSLInsight_
DecryptSide
ACOS_decrypt(config-slb vserver-vport)# template ssli ldap_insight
ACOS_decrypt(config-slb vserver-vport)# exit
```

6. Configure port 21 SSLi to intercept encrypted explicit FTP traffic from the clients. Bind the service-group **FW1_Inspect_FTP** with member port 21, explicit FTPS client-SSL template, and SSLi template.

```
ACOS_decrypt(config-slb vserver)# port 21 ssli
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat
```

```
ACOS_decrypt(config-slb vserver-vport)# service-group FW1_Inspect_FTP
ACOS_decrypt(config-slb vserver-vport)# template client-ssl Explicit_
FTPS
ACOS_decrypt(config-slb vserver-vport)# template ssli ftp_insight
ACOS_decrypt(config-slb vserver-vport)# exit
```

7. Configure port 0 SSLi to intercept encrypted implicit FTP traffic from the clients. Bind the service-group **ALL_TCP_SG** with member port 0 and implicit FTPS client-SSL template.

```
ACOS_decrypt(config-slb vserver)# port 0 ssl-proxy
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat
ACOS_decrypt(config-slb vserver-vport)# service-group ALL_TCP_SG tcp
ACOS_decrypt(config-slb vserver-vport)# template client-ssl Implicit_
FTPS
ACOS_decrypt(config-slb vserver-vport)# exit
```

8. Configure port 0 TCP, UDP, and Others to intercept all encrypted UDP and TCP traffic from the clients. Bind the service-group **ALL_TCP_SG** and **ALL_UDP_SG** with member port 0.

```
ACOS_decrypt(config-slb vserver)# port 0 tcp
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat
ACOS_decrypt(config-slb vserver-vport)# service-group ALL_TCP_SG
ACOS_decrypt(config-slb vserver-vport)# exit

ACOS_decrypt(config-slb vserver)# port 0 udp
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat
ACOS_decrypt(config-slb vserver-vport)# service-group ALL_UDP_SG
ACOS_decrypt(config-slb vserver-vport)# exit

ACOS_decrypt(config-slb vserver)# port 0 others
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat
ACOS_decrypt(config-slb vserver-vport)# service-group ALL_UDP_SG
ACOS_decrypt(config-slb vserver-vport)# exit
ACOS_decrypt(config-slb vserver)# exit
```

ACOS_encrypt Configuration

Perform the following steps for the ACOS_encrypt partition:

The following topics are covered:

Step 1. Configuring the Network VLANs	128
Step 2. Configuring the Network IP Addresses	128
Step 3. Configuring the SSLi Services	128
Step 4. Configuring the SSLi Service Groups	130
Step 5. Configuring the Virtual Server	131

Step 1. Configuring the Network VLANs

```
ACOS(config)# hostname ACOS_encrypt
ACOS_encrypt(config)# vlan 20
ACOS_encrypt(config-vlan:20)# tagged ethernet 2
ACOS_encrypt(config-vlan:20)#router-interface ve 20
ACOS_encrypt(config-vlan:20)# exit

ACOS_encrypt(config)# vlan 15
ACOS_encrypt(config-vlan:15)# tagged ethernet 1
ACOS_encrypt(config-vlan:15)#router-interface ve 15
ACOS_encrypt(config-vlan:15)# exit
```

Step 2. Configuring the Network IP Addresses

```
ACOS_encrypt(config)# interface ve 20
ACOS_encrypt(config-if:ve20)# ip address 20.1.1.2 /24
ACOS_encrypt(config-if:ve20)# exit

ACOS_encrypt(config)# interface ve 15
ACOS_encrypt(config-if:ve15)# ip address 10.15.1.12 /24
ACOS_encrypt(config-if:ve15)# exit
```

Step 3. Configuring the SSLi Services

1. Create an SSL server template on ACOS_encrypt so that the VIP on ACOS_encrypt can operate as an SSL client and handshake with the EnterpriseABC server.

```
ACOS_encrypt(config)# slb template server-ssl SSLInsight_EncryptSide
ACOS_encrypt(config-server ssl)# forward-proxy-enable
ACOS_encrypt(config-server ssl)# exit
```


- a. Create an SSL server template to configure implicit FTPS for TLS data transmission and enable SSLi FTP. If you want to enable TLS reuse then configure session-ticket-enable as well.

```
ACOS_encrypt(config)# slb template server-ssl Implicit_FTPS
ACOS_encrypt(config-server ssl)# close-notify
ACOS_encrypt(config-server ssl)# forward-proxy-enable
ACOS_encrypt(config-server ssl)# session-ticket-enable
ACOS_encrypt(config-server ssl)# enable-ssli-ftp-alg 990
ACOS_encrypt(config-server ssl)# exit
```

- b. Create an SSLi server template to configure explicit FTPS for TLS data transmission.

```
ACOS_encrypt(config)# slb template server-ssl Explicit_FTPS
ACOS_encrypt(config-server ssl)# ssli-logging all
ACOS_encrypt(config-server ssl)# close-notify
ACOS_encrypt(config-server ssl)# forward-proxy-enable
ACOS_encrypt(config-server ssl)# exit
```

2. Create the real server **Default_Gateway**. ACOS_encrypt forwards the traffic on these ports over VLAN 20 to the default gateway at IP address 20.1.1.10. The default gateway has a route to the EnterpriseABC server. Bind the SLB ports of the intercepted non-HTTP protocols to **Default_Gateway**.

- TCP ports 25 to forward encrypted SMTP traffic.

```
ACOS_encrypt(config)# slb server Default_Gateway 20.1.1.10
ACOS_encrypt(config-real server)# port 25 tcp
ACOS_encrypt(config-real server-node port)# health-check-disable
ACOS_encrypt(config-real server-node port)# exit
ACOS_encrypt(config-real server)# exit
```

- TCP ports 110 to forward encrypted POP traffic.

```
ACOS_encrypt(config-real server)# port 110 tcp
ACOS_encrypt(config-real server-node port)# health-check-disable
ACOS_encrypt(config-real server-node port)# exit
ACOS_encrypt(config-real server)# exit
```

- TCP ports 5522 to forward encrypted XMPP traffic.

```
ACOS_encrypt(config-real server)# port 5522 tcp
```

```
ACOS_encrypt(config-real server-node port)# health-check-disable
ACOS_encrypt(config-real server-node port)# exit
ACOS_encrypt(config-real server)# exit
```

- TCP ports 21 to forward decrypted FTP traffic.

```
ACOS_decrypt(config-real server)# port 21 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit
ACOS_decrypt(config-real server)# exit
```

- TCP ports 389 to forward decrypted LDAP traffic.

```
ACOS_decrypt(config-real server)# port 389 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit
ACOS_decrypt(config-real server)# exit
```

- Wildcard ports: **port 0 tcp** and **port 0 udp** to forward all other encrypted UDP and TCP traffic.

```
ACOS_encrypt(config-real server)# port 0 tcp
ACOS_encrypt(config-real server-node port)# health-check-disable
ACOS_encrypt(config-real server-node port)# exit
ACOS_encrypt(config-real server)# exit

ACOS_encrypt(config-real server)# port 0 udp
ACOS_encrypt(config-real server-node port)# health-check-disable
ACOS_encrypt(config-real server-node port)# exit
ACOS_encrypt(config-real server)# exit
```

Step 4. Configuring the SSLi Service Groups

1. Configure one service-group **DG_SSL_SG** and binding it to ports 25, 5522, and 110 to inspect SMTP, XMPP, and POP traffic. You can create separate service-groups for each ports as well.

```
ACOS_encrypt(config)# slb service-group DG_SSL_SG tcp
ACOS_encrypt(config-slbg svc group)# member Default_Gateway 25
ACOS_encrypt(config-slbg svc group)# member Default_Gateway 5522
ACOS_encrypt(config-slbg svc group)# member Default_Gateway 110
ACOS_encrypt(config-slbg svc group)# exit
```

2. Configure one service-group **DG_SSL_FTP** and associate member port 21 to inspect FTP traffic.

```
ACOS_decrypt(config)# slb service-group DG_SSL_FTP tcp
ACOS_decrypt(config-slb svc group)# member Default_Gateway 21
```

3. Configure one service-group **DG_SSL_LDAP** and associate member port 389 to inspect LDAP traffic.

```
ACOS_decrypt(config)# slb service-group DG_SSL_LDAP tcp
ACOS_decrypt(config-slb svc group)# member Default_Gateway 389
```

4. Configure two service groups **DG_TCP_SG** and **DG_UDP_SG** and bind it to port 0 to inspect all other traffic.

```
ACOS_encrypt(config)# slb service-group DG_TCP_SG tcp
ACOS_encrypt(config-slb svc group)# member Default_Gateway 0
ACOS_encrypt(config-slb svc group)# exit

ACOS_encrypt(config)# slb service-group DG_UDP_SG udp
ACOS_encrypt(config-slb svc group)# member Default_Gateway 0
ACOS_encrypt(config-slb svc group)# exit
```

Step 5. Configuring the Virtual Server

For an explanation of the procedure, refer to a similar procedure discussed in [Step 5. Configuring the Virtual Server](#).

The only deviation is that the **port 21 ssl**, **port 25 ssl**, **port 110 ssl**, **port 389 ssl**, and **port 5522 ssl** in this example must be configured as part of the virtual server **Encrypt_VIP**.

1. Configure the ACL and SLB virtual server.

```
ACOS_encrypt(config)# access-list 101 permit ip any any vlan 15
ACOS_encrypt(config)# slb virtual-server Encrypt_VIP 0.0.0.0 acl 101
```

2. Configure port 25 SSLi to intercept decrypted SMTP traffic from the clients. Bind the service-group **DG_SSL_SG** which has port 25, server-SSL template, and SMTP SSLi template.

```
ACOS_encrypt(config-slb vsrver)# port 25 ssl
ACOS_encrypt(config-slb vsrver-vport)# no-dest-nat
```

```
ACOS_encrypt(config-slb vserver-vport)# service-group DG_SSL_SG
ACOS_encrypt(config-slb vserver-vport)# template server-ssl SSLInsight_
EncryptSide
ACOS_encrypt(config-slb vserver-vport)# template ssls smtp_insight
ACOS_encrypt(config-slb vserver-vport)# exit
```

3. Configure port 110 SSLi to intercept decrypted POP traffic from the clients. Bind the service-group **DG_SSL_SG** which has port 110, server-SSL template, and SMTP SSLi template.

```
ACOS_encrypt(config-slb vserver)# port 110 ssls
ACOS_encrypt(config-slb vserver-vport)# no-dest-nat
ACOS_encrypt(config-slb vserver-vport)# service-group DG_SSL_SG
ACOS_encrypt(config-slb vserver-vport)# template server-ssl SSLInsight_
EncryptSide
ACOS_encrypt(config-slb vserver-vport)# template ssls pop_insight
ACOS_encrypt(config-slb vserver-vport)# exit
```

4. Configure port 5522 SSLi to intercept decrypted XMPP traffic from the clients. Bind the service-group **DG_SSL_SG** which has port 5522, server-SSL template, and XMPP SSLi template.

```
ACOS_encrypt(config-slb vserver)# port 5522 ssls
ACOS_encrypt(config-slb vserver-vport)# no-dest-nat
ACOS_encrypt(config-slb vserver-vport)# service-group DG_SSL_SG
ACOS_encrypt(config-slb vserver-vport)# template server-ssl SSLInsight_
EncryptSide
ACOS_encrypt(config-slb vserver-vport)# template ssls xmpp_insight
ACOS_encrypt(config-slb vserver-vport)# exit
```

5. Configure port 389 SSLi to intercept decrypted LDAP traffic from the clients. Bind the service-group **DG_SSL_LDAP** which has port 389, server-SSL template, and LDAP SSLi template.

```
ACOS_encrypt(config-slb vserver)# port 389 ssls
ACOS_encrypt(config-slb vserver-vport)# no-dest-nat
ACOS_encrypt(config-slb vserver-vport)# service-group DG_SSL_LDAP
ACOS_encrypt(config-slb vserver-vport)# template server-ssl SSLInsight_
EncryptSide
ACOS_encrypt(config-slb vserver-vport)# template ssls ldap_insight
ACOS_encrypt(config-slb vserver-vport)# exit
```

6. Configure port 0 TCP, UDP, and Others to intercept all UDP and TCP traffic from the clients.

```
ACOS_encrypt(config-slb vserver)# port 0 tcp
ACOS_encrypt(config-slb vserver-vport)# no-dest-nat
ACOS_encrypt(config-slb vserver-vport)# service-group DG_TCP_SG
ACOS_encrypt(config-slb vserver-vport)# exit

ACOS_encrypt(config-slb vserver)# port 0 udp
ACOS_encrypt(config-slb vserver-vport)# no-dest-nat
ACOS_encrypt(config-slb vserver-vport)# service-group DG_UDP_SG
ACOS_encrypt(config-slb vserver-vport)# exit

ACOS_encrypt(config-slb vserver)# port 0 others
ACOS_encrypt(config-slb vserver-vport)# no-dest-nat
ACOS_encrypt(config-slb vserver-vport)# service-group DG_UDP_SG
ACOS_encrypt(config-slb vserver-vport)# exit
ACOS_encrypt(config-slb vserver)# exit
```

7. Configure port 21 SSLi to intercept decrypted explicit FTP traffic from the clients. Bind the service-group **DG_SSL_FTP** which has port 21, explicit server-SSL template, and FTP SSLi template.

```
ACOS_encrypt(config-slb vserver)# port 21 ssli
ACOS_encrypt(config-slb vserver-vport)# source-nat auto
ACOS_encrypt(config-slb vserver-vport)# service-group DG_SSL_FTP
ACOS_encrypt(config-slb vserver-vport)# template server-ssl Explicit_FTPS
ACOS_encrypt(config-slb vserver-vport)# template ssli ftp_insight
ACOS_encrypt(config-slb vserver-vport)# no-dest-nat
ACOS_encrypt(config-slb vserver-vport)# exit
```

8. Configure the Differentiated Services Code Point (DSCP) for Implicit FTPS. Bind the service-group **DG_SSL_FTP** which has member port 0, implicit server-SSL template, and FTP SSLi template.

```
ACOS_encrypt(config)# access-list 102 permit ip any any dscp 10
ACOS_encrypt(config)# slb virtual-server implicit-dscp10 0.0.0.0 acl
102

ACOS_encrypt(config-slb vserver)# port 0 tcp-proxy
```

```
ACOS_encrypt(config-slb vserver-vport)# source-nat pool V4_20_NAT
ACOS_encrypt(config-slb vserver-vport)# service-group DG_TCP_SG
ACOS_encrypt(config-slb vserver-vport)# template server-ssl Implicit_
FTPS
ACOS_encrypt(config-slb vserver-vport)# no-dest-nat
ACOS_encrypt(config-slb vserver-vport)# exit
```

Consolidated Configuration Examples

Use the **show running-config** command to check your configuration for both ACOS_decrypt and ACOS_encrypt.

```
ACOS_decrypt# show running-config
!
access-list 100 permit ip any any vlan 10
!
vlan 10
    tagged ethernet 1
    router-interface ve 10
!
vlan 15
    tagged ethernet 2
    router-interface ve 15
!
hostname ACOS_decrypt
!
interface management
    ip address dhcp
!
interface ethernet 1
    enable
!
interface ethernet 2
    enable
!
interface ve 10
    ip address 10.10.1.2 255.255.255.0
    ip allow-promiscuous-vip
```

```
!  
interface ve 15  
  ip address 10.15.1.2 255.255.255.0  
!  
slb template ssli xmpp_insight  
  type xmpp  
!  
slb template ssli smtp_insight  
  type smtp  
!  
slb template ssli pop_insight  
  type pop  
!  
slb template ssli ftp_insight  
  type ftp  
!  
slb template ssli ldap_insight  
  type ldap  
!  
slb server FW1_Inspect 10.15.1.12  
  port 0 tcp  
    health-check-disable  
  port 0 udp  
    health-check-disable  
  port 25 tcp  
    health-check-disable  
  port 110 tcp  
    health-check-disable  
  port 21 tcp  
    health-check-disable  
  port 389 tcp  
    health-check-disable  
  port 5522 tcp  
    health-check-disable  
!  
slb service-group ALL_TCP_SG tcp  
  member FW1_Inspect 0  
!  
slb service-group ALL_UDP_SG udp  
  member FW1_Inspect 0
```

```
!  
slb service-group FW1_Inspect_SG tcp  
    member FW1_Inspect 25  
    member FW1_Inspect 5522  
    member FW1_Inspect 110  
!  
slb service-group FW1_Inspect_FTP tcp  
    member FW1_Inspect 21  
!  
slb service-group FW1_Inspect_LDAP tcp  
    member FW1_Inspect 389  
!  
slb template client-ssl SSLInsight_DecryptSide  
    forward-proxy-ca-certificate Cert123.pem key key123  
    forward-proxy-enable  
!  
slb template client-ssl Explicit_FTPS  
    ssli-logging all  
    close-notify  
    forward-proxy-ca-certificate Cert123.pem key key123  
    forward-proxy-enable  
!  
slb template client-ssl Implicit_FTPS  
    ssli-logging all  
    close-notify  
    forward-proxy-ca-certificate Cert123.pem key key123  
    forward-proxy-decrypt dscp 10 2  
    forward-proxy-enable  
    enable-ssli-ftp-alg 990  
!  
slb virtual-server Decrypt_VIP 0.0.0.0 acl 100  
    port 0 tcp  
        service-group ALL_TCP_SG  
        no-dest-nat  
    port 0 udp  
        service-group ALL_UDP_SG  
        no-dest-nat  
    port 0 ssl-proxy  
        service-group ALL_TCP_SG tcp  
        template client-ssl Implicit_FTPS
```



```
no-dest-nat
port 21 ssli
  service-group FW1_Inspect_FTP
  template client-ssl Explicit_FTPS
  template ssli ftp_insight
  no-dest-nat
port 25 ssli
  service-group FW1_Inspect_SG
  template client-ssl SSLInsight_DecryptSide
  template ssli smtp_insight
  no-dest-nat
port 110 ssli
  service-group FW1_Inspect_SG
  template client-ssl SSLInsight_DecryptSide
  template ssli pop_insight
  no-dest-nat
port 389 ssli
  service-group FW1_Inspect_LDAP
  template client-ssl SSLInsight_DecryptSide
  template ssli ldap_insight
  no-dest-nat
port 5522 ssli
  service-group FW1_Inspect_SG
  template client-ssl SSLInsight_DecryptSide
  template client-ssl xmpp_insight
  no-dest-nat
!
end
```

ACOS_encrypt# show running-config

```
!
access-list 101 permit ip any any vlan 15
!
vlan 20
  tagged ethernet 2
  router-interface ve 20
!
vlan 15
  tagged ethernet 1
  router-interface ve 15
```

```
!  
hostname ACOS_encrypt  
!  
interface management  
    ip address dhcp  
!  
interface ethernet 1  
    enable  
!  
interface ethernet 2  
    enable  
!  
interface ve 20  
    ip address 20.1.1.2 255.255.255.0  
!  
interface ve 15  
    ip address 10.15.1.12 255.255.255.0  
    ip allow-promiscuous-vip  
!  
slb template ssli xmpp_insight  
    type xmpp  
!  
slb template ssli smtp_insight  
    type smtp  
!  
slb template ssli pop_insight  
    type pop  
!  
slb template ssli ftp_insight  
    type ftp  
!  
slb template ssli ldap_insight  
    type ldap  
!  
slb server Default_Gateway 20.1.1.10  
    port 0 tcp  
        health-check-disable  
    port 0 udp  
        health-check-disable  
    port 21 tcp
```

```
    health-check-disable
port 25 tcp
    health-check-disable
port 110 tcp
    health-check-disable
port 389 tcp
    health-check-disable
port 5522 tcp
    health-check-disable
!
slb service-group DG_SSL_SG tcp
    member Default_Gateway 25
    member Default_Gateway 5522
    member Default_Gateway 110

slb service-group DG_TCP_SG tcp
    member Default_Gateway 0
!
slb service-group DG_UDP_SG udp
    member Default_Gateway 0
!
slb service-group DG_SSL_FTP tcp
    member Default_Gateway 21
!
slb service-group DG_SSL_LDAP tcp
    member Default_Gateway 389
!
slb template server-ssl SSLInsight_EncryptSide
    forward-proxy-enable
!
slb template server-ssl Implicit_FTPS
    close-notify
    forward-proxy-enable
    session-ticket-enable
    enable-ssli-ftp-alg 990
!
slb template server-ssl Explicit_FTPS
    ssli-logging all
    close-notify
    forward-proxy-enable
```

```
!  
slb virtual-server Encrypt_VIP 0.0.0.0 acl 101  
  port 0 tcp  
    no-dest-nat  
    service-group DG_TCP_SG  
  port 0 udp  
    no-dest-nat  
    service-group DG_UDP_SG  
  port 0 others  
    no-dest-nat  
    service-group DG_UDP_SG  
  port 21 ssli  
    source-nat auto  
    no-dest-nat  
    service-group DG_SSL_FTP  
    template server-ssl Explicit_FTPS  
    template ssli ftp_insight  
  port 25 ssli  
    no-dest-nat  
    service-group DG_SSL_SG  
    template server-ssl SSLInsight_EncryptSide  
    template ssli smtp_insight  
  port 110 ssli  
    no-dest-nat  
    service-group DG_SSL_SG  
    template server-ssl SSLInsight_EncryptSide  
    template ssli pop_insight  
  port 389 ssli  
    no-dest-nat  
    service-group DG_SSL_LDAP  
    template server-ssl SSLInsight_EncryptSide  
    template ssli ldap_insight  
  port 5522 ssli  
    no-dest-nat  
    service-group DG_SSL_SG  
    template server-ssl SSLInsight_EncryptSide  
    template ssli xmpp_insight  
!  
<For DSCP Implicit FTPS>  
access-list 102 permit ip any any dscp 10
```

```
!  
.  
.  
.  
.  
slb virtual-server implicit-dscp10 0.0.0.0 acl 102  
  port 0 tcp-proxy  
  no-dest-nat  
  source-nat pool V4_20_NAT  
  service-group DG_TCP_SG  
  template server-ssl Implicit_FTPS  
!  
end
```

Inbound Static-Port Type HTTPS

This section describes how to configure inbound SSLi. It refers to the intercepting and decrypting SSL/TLS traffic originating from the Internet into your internal SSL web application servers.

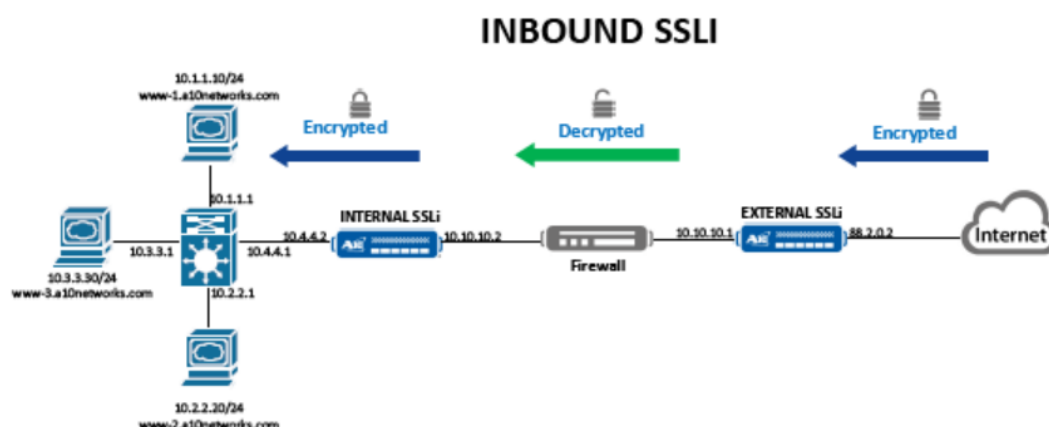
The following topics are covered:

Deployment Example	143
CLI Configuration	144

Deployment Example

This section provides detailed steps for configuring SSLi to transparently intercept HTTPS traffic from clients, decrypt the traffic so that it can be inspected at the firewall, re-encryption of the traffic and forwarding it to the SSL server that the clients are trying to reach. The example of SSLi contained in this section intercepts only HTTPS sessions. Using virtual port type HTTPS, the virtual ports are specified using the `port 443 https` command. All other SSL and non-SSL traffic is bypassed. The topology for this example is illustrated in [Figure 14](#).

Figure 14 : Inbound SSLi Network Topology



The configuration of SSLi in this section is one in which the clients are connecting to SSL servers running on a private network behind a firewall. The session connects "inbound" to the private network.

Inbound and outbound SSLi can be configured together. Traffic flowing in both directions would be decrypted and re-encrypted. However the command lines that configure the inbound virtual-servers must go before the command lines that configure the outbound virtual servers. For the configuration of outbound SSLi, refer [Static-Port Type HTTPS SSLi](#).

CLI Configuration

Configure the External Inbound ACOS device

Before beginning this configuration, you must import the certificates and private keys of the SSL/TLS servers that SSLi will be provisioned to decrypt and encrypt. In the configuration that follows, each server will be mapped by domain to a certificate, private key pair. In addition, a default certificate and corresponding private key will be configured.

See the [Importing a Certificate and Key](#) for information on importing certificates and keys.

1. Configure the access lists. Traffic coming from the Internet is filtered to permit traffic going to the following three private networks.

```
access-list 101 permit ip any 10.1.1.0 0.0.0.255
access-list 101 permit ip any 10.2.2.0 0.0.0.255
access-list 101 permit ip any 10.3.3.0 0.0.0.255
```

2. Configure the virtual Ethernet interface, 100, facing the Internet and give an IP address, 10.10.10.1. Configure a second interface, 882, facing the firewall protecting the private networks. Assign the public IP address 88.2.0.2 to this interface.

```
vlan 100
  untagged ethernet 52
  router-interface ve 100
!
vlan 882
  untagged ethernet 51
  router-interface ve 882
!
hostname Ext-Inbound-SSLi
!
timezone America/Los_Angeles
!
```



```
ntp server 10.101.4.10
!
interface management
 ip address 10.101.6.190 255.255.252.0
 ip default-gateway 10.101.4.1
!
interface ethernet 51
 enable
!
interface ethernet 52
 enable
!
interface ve 100
 ip address 10.10.10.1 255.255.255.0
!
interface ve 882
 ip address 88.2.0.2 255.255.255.0
 ip allow-promiscuous-vip
!
```

3. Configure a default route to an Internet router, and configure static routes from the virtual Ethernet interfaces to the private network.

```
ip route 0.0.0.0 /0 88.2.0.1
ip route 10.1.1.0 /24 10.10.10.2
ip route 10.2.2.0 /24 10.10.10.2
ip route 10.3.3.0 /24 10.10.10.2
ip route 10.4.4.0 /24 10.10.10.2
```

4. Configure the SSL-client template for SNI-mapped certificate-key pairs. If a client includes the SNI extension in its Hello message, the SSLi session connects to the server in the specified domain using the certificate and key that are mapped to the domain requested by the client. Use the following command:

```
certificate <cert-name> key <key-name> [pass-phrase <pass-phrase-str>]
[chain-cert <chain-cert-name>]
```

Certificate and key configuration must be put in one line because they should exist at the same time.

```
slb template client-ssl inbound-ssli
```

```

server abc.com certificate cert1 key key1 pass-phrase Pass1 chain-cert
Cert1
server xyz.com certificate cert2 key key2 pass-phrase Pass2 chain-cert
Cert2
!
```

5. Configure three protocol ports that forward traffic on real servers to the firewall. Only port 8080 tcp is configured to decrypt the SSL traffic that it receives from the Internet on port 443 https. Protocol port 0 udp and port 0 tcp forward all other traffic to the firewall.

```

slb server gw2-bp 10.10.10.2
  port 0 tcp
    health-check-disable
  port 0 udp
    health-check-disable
  port 8080 tcp
    health-check-disable
!
slb service-group gw2-bp-8080 tcp
  member gw2-bp 8080
!
slb service-group gw2-bp-tcp tcp
  member gw2-bp 0
!
slb service-group gw2-bp-udp udp
  member gw2-bp 0
!
```

6. Configure the virtual server with the ports configured in the previous. Assign service groups to forward the traffic of these ports to the firewall. In addition provision the IP datagrams to send replies to clients back through the last hop on which the request for the virtual port's service was received and to use the IP address of the virtual traffic when forwarding traffic (do not use destination NAT).

```

slb virtual-server vip1-ext 0.0.0.0 acl 101
  port 0 tcp
    service-group gw2-bp-tcp
    use-rcv-hop-for-resp
    no-dest-nat
```

```
port 0 udp
  service-group gw2-bp-udp
  use-rcv-hop-for-resp
  no-dest-nat
port 0 others
  service-group gw2-bp-udp
  use-rcv-hop-for-resp
  no-dest-nat
port 443 https
  service-group gw2-bp-8080
  use-rcv-hop-for-resp
  template client-ssl inbound-ssli
  no-dest-nat port-translation
```

7. Use the `show running-config` command to check your configuration of the external ACOS device.

```
Ext-Inbound# show running-config
!
access-list 101 permit ip any 10.1.1.0 0.0.0.255
access-list 101 permit ip any 10.2.2.0 0.0.0.255
access-list 101 permit ip any 10.3.3.0 0.0.0.255
!
vlan 100
  untagged ethernet 52
  router-interface ve 100
!
vlan 882
  untagged ethernet 51
  router-interface ve 882
!
hostname Ext-Inbound-SSLi
!
timezone America/Los_Angeles
!
ntp server 10.101.4.10
!
interface management
  ip address 10.101.6.190 255.255.252.0
  ip default-gateway 10.101.4.1
```

```
!  
interface ethernet 51  
    enable  
!  
interface ethernet 52  
    enable  
!  
interface ve 100  
    ip address 10.10.10.1 255.255.255.0  
!  
interface ve 882  
    ip address 88.2.0.2 255.255.255.0  
    ip allow-promiscuous-vip  
!  
ip route 0.0.0.0 /0 88.2.0.1  
ip route 10.1.1.0 /24 10.10.10.2  
ip route 10.2.2.0 /24 10.10.10.2  
ip route 10.3.3.0 /24 10.10.10.2  
ip route 10.4.4.0 /24 10.10.10.2  
!  
slb template client-ssl inbound-ssli  
    server abc.com certificate cert1 key key1  
!  
slb server gw2-bp 10.10.10.2  
    port 0 tcp  
        health-check-disable  
    port 0 udp  
        health-check-disable  
    port 8080 tcp  
        health-check-disable  
!  
slb service-group gw2-bp-8080 tcp  
    member gw2-bp 8080  
!  
slb service-group gw2-bp-tcp tcp  
    member gw2-bp 0  
!  
slb service-group gw2-bp-udp udp
```

```
member gw2-bp 0
!
slb virtual-server vip1-ext 0.0.0.0 acl 101
  port 0 tcp
    service-group gw2-bp-tcp
    use-rcv-hop-for-resp
    no-dest-nat
  port 0 udp
    service-group gw2-bp-udp
    use-rcv-hop-for-resp
    no-dest-nat
  port 0 others
    service-group gw2-bp-udp
    use-rcv-hop-for-resp
    no-dest-nat
  port 443 https
    service-group gw2-bp-8080
    use-rcv-hop-for-resp
    template client-ssl inbound-ssli
    no-dest-nat port-translation
!
```

Configure the Internal Inbound ACOS device

1. Configure the access lists. Traffic coming from the Internet is filtered to permit traffic going to the following three private networks.

```
access-list 101 permit ip any 10.1.1.0 0.0.0.255
access-list 101 permit ip any 10.2.2.0 0.0.0.255
access-list 101 permit ip any 10.3.3.0 0.0.0.255
```

2. Configure the virtual Ethernet interface, 100, facing the inbound traffic and give it an IP address, 10.10.10.2. Configure a second interface, 104, facing the outbound direction and the private networks. Assign the private IP address 10.4.4.2 to this interface.

```
vlan 100
  untagged ethernet 52
  router-interface ve 100
!
```

```
vlan 104
  untagged ethernet 51
  router-interface ve 104
!
hostname Int-Inbound-SSLi
!
timezone America/Los_Angeles
!
ntp server 10.101.4.10
!
interface management
  ip address 10.101.6.191 255.255.252.0
  ip default-gateway 10.101.4.1
!
interface ethernet 51
  enable
!
interface ethernet 52
  enable
!
interface ve 100
  ip address 10.10.10.2 255.255.255.0
  ip allow-promiscuous-vip
!
interface ve 104
  ip address 10.4.4.2 255.255.255.0
!
```

3. Configure a default route to the private network and specify the service groups that forward traffic to that network.

The ACOS real server, the server-ssl is configured to re-establish SSL sessions that were intercepted by the external ACOS device.

```
ip route 0.0.0.0 /0 10.10.10.1 ip route 10.1.1.0 /24 10.4.4.1 ip route
10.2.2.0 /24 10.4.4.1 ip route 10.3.3.0 /24 10.4.4.1 !slb server
internal-gw 10.4.4.1 port 0 tcp health-check-disable port 0 udp
health-check-disable port 443 tcp health-check-disable !slb service-
group internal-gw-443 tcp member internal-gw 443 !slb service-group
internal-gw-tcp tcp member internal-gw 0 !slb service-group internal-
gw-udp udp member internal-gw 0 !
slb template server-ssl inbound-ssli
    forward-proxy-enable
!
```

4. Configure the virtual server that re-encrypted traffic received on port 8080 http. The non-SSL sessions are received on the wildcard ports 0 udp, 0 tcp, and 0 others.

```
slb virtual-server vip1-int 0.0.0.0 acl 101
    port 0 tcp
        service-group internal-gw-tcp
        use-rcv-hop-for-resp
        no-dest-nat
    port 0 udp
        service-group internal-gw-udp
        use-rcv-hop-for-resp
        no-dest-nat
    port 0 others
        service-group internal-gw-udp
        use-rcv-hop-for-resp
        no-dest-nat
    port 8080 http
        service-group internal-gw-443
        use-rcv-hop-for-resp
        template server-ssl inbound-ssli
        no-dest-nat port-translation
!
```

5. Use the `show running-config` command to check your configuration of the internal ACOS device.

```
Int-Inbound# show running-config
!
access-list 101 permit ip any 10.1.1.0 0.0.0.255
```

```
access-list 101 permit ip any 10.2.2.0 0.0.0.255
access-list 101 permit ip any 10.3.3.0 0.0.0.255
!
vlan 100
    untagged ethernet 52
    router-interface ve 100
!
vlan 104
    untagged ethernet 51
    router-interface ve 104
!
hostname Int-Inbound-SSLi
!
timezone America/Los_Angeles
!
ntp server 10.101.4.10
!
interface management
    ip address 10.101.6.191 255.255.252.0
    ip default-gateway 10.101.4.1
!
interface ethernet 51
    enable
!
interface ethernet 52
    enable
!
interface ve 100
    ip address 10.10.10.2 255.255.255.0
    ip allow-promiscuous-vip
!
interface ve 104
    ip address 10.4.4.2 255.255.255.0
!
ip route 0.0.0.0 /0 10.10.10.1
ip route 10.1.1.0 /24 10.4.4.1
ip route 10.2.2.0 /24 10.4.4.1
ip route 10.3.3.0 /24 10.4.4.1
!
```



```
slb server internal-gw 10.4.4.1
  port 0 tcp
    health-check-disable
  port 0 udp
    health-check-disable
  port 443 tcp
    health-check-disable
!
slb service-group internal-gw-443 tcp
  member internal-gw 443
!
slb service-group internal-gw-tcp tcp
  member internal-gw 0
!
slb service-group internal-gw-udp udp
  member internal-gw 0
!
slb template server-ssl inbound-ssli
  forward-proxy-enable
!
slb virtual-server vip1-int 0.0.0.0 acl 101
  port 0 tcp
    service-group internal-gw-tcp
    use-rcv-hop-for-resp
    no-dest-nat
  port 0 udp
    service-group internal-gw-udp
    use-rcv-hop-for-resp
    no-dest-nat
  port 0 others
    service-group internal-gw-udp
    use-rcv-hop-for-resp
    no-dest-nat
  port 8080 http
    service-group internal-gw-443
    use-rcv-hop-for-resp
    template server-ssl inbound-ssli
    no-dest-nat port-translation
!
```

Dynamic-Port Inspection

This section describes how to configure dynamic port inspection for SSLi. It allows to decrypt SSL-wrapped traffic on any port, not just port 443 where most of it happens.

The following topics are covered:

Configuration Workflow	154
CLI Configuration	156
DSCP Inspection	168

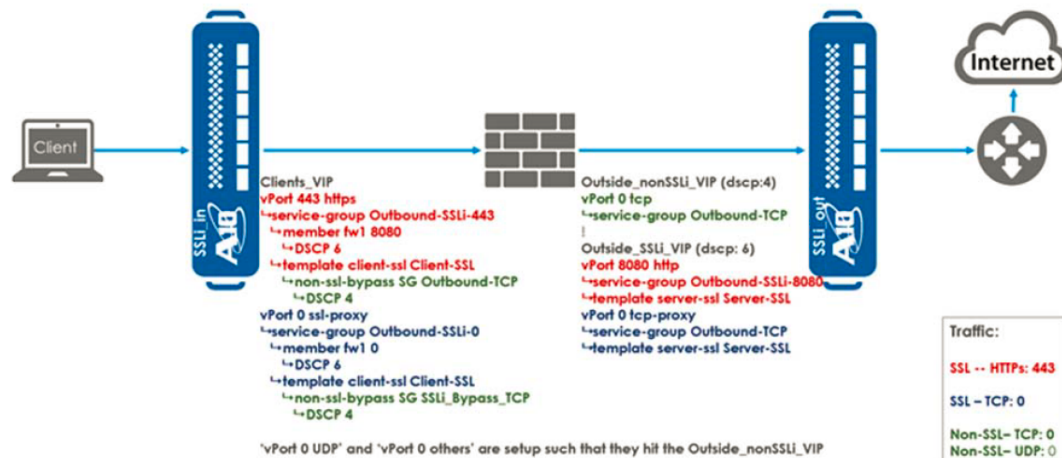
Configuration Workflow

Since Dynamic-Port SSLi is configured in parallel with SSLi over known ports, in order to configure Dynamic-Port SSLi you need to address three flows:

- SSL traffic arriving on known ports—This is addressed by standard static-port SSLi configuration, however you will need to explicitly tag this traffic as decrypted using a custom DSCP value (ex. Dscp=6).
- SSL traffic arriving on unknown ports—This is addressed by Dynamic-Port SSLi configuration, and all decrypted traffic is tagged using a custom DSCP value (ex. Dscp=6).
- All SSLi-bypassed & non-SSL (TCP, UDP, ICMP etc.) traffic arriving on unknown ports—This is addressed with wildcard vPorts and service-groups, however you will need to explicitly tag this traffic as non-SSK using a custom DSCP value (ex. Dscp=4).

[Figure 15](#) below illustrates the overall DSCP dynamic-port SSLi configuration workflow.

Figure 15 : DSCP Dynamic-Port Configuration Workflow



ACOS_decrypt Configuration

Clients_VIP SLB Virtual Server

- Provides SSL forward proxy service that enables ACOS_decrypt to proxy for the remote SSL servers and bring up SSL sessions with the clients. SSL traffic from the clients arriving on unknown ports is decrypted and forwarded to the `Outbound-SSLi-0` service group, whereas bypassed and non-SSL traffic is forwarded to either the `Outbound-TCP` service group or the `Outbound-UDP` service group. SSL traffic arriving on standard SSL vPort is decrypted and forwarded to the `Outbound-SSLi-443` service-group.
- Outbound-SSLi-0 SLB Service Group—Marks all decrypted SSL traffic arriving on unknown TCP ports with custom DSCP value (ex.6) and forwards it to the security device.
- Outbound-SSLi-443 SLB Service Group—Marks all decrypted SSL traffic arriving on known SSL ports (443 in this example) with custom DSCP value (6 in this example) and forwards it to the security device.
- Outbound-TCP and Outbound-UDP SLB Service Groups—Marks all other TCP traffic with custom DSCP value (4 in this example) and forwards it to the security device.

This traffic stream includes non-SSL traffic as well as any SSL traffic which was purposefully bypassed in SSLi configuration.

ACOS_encrypt Configuration

- **Encrypt_SSLi_VIP** wildcard SLB Virtual Server—Provides server-SSL services for decrypted traffic that enable ACOS_decrypt to establish SSL connections with the remote SSL servers through the Gateway SLB real server, completing end-to-end SSL connectivity.
- **Outside_nonSSLi_VIP** wildcard SLB Virtual Server—Forwards all bypassed TCP traffic arriving at the outside ACOS to the Gateway SLB real server.
- **Outbound-SSLi-8080** SLB Service Group—Forwards all decrypted traffic arriving on static port 8080 to the Internet default gateway.
- **Outbound-TCP** and **Outbound-UDP** SLB Service Groups—Forwards all other non-SSL as well as decrypted TCP traffic to the Internet default gateway.

CLI Configuration

The following topics are covered:

Inside ACOS Configuration	156
ACOS_encrypt Configuration	159
ACOS_decrypt Configuration	161
Consolidated Configuration	164

Inside ACOS Configuration

The ACOS_decrypt zone is configured as the client-facing device. Key configuration elements include the following:

1. Define Access-List to identify traffic of interest.
2. Import a proxied CA certificate, and the certificate's private key. This certificate must be trusted by clients.

3. Define two SLB port templates for marking dscp values. In this example, we use dscp=6 for marking decrypted traffic and dscp=4 for marking all bypassed traffic.

```
!  
slb template port decrypt-dscp-6  
    dscp 6  
!  
slb template port non-ssli-dscp-4  
    dscp 4
```

4. Create an SLB real server for a path through the security device for all TCP and UDP traffic.

```
!  
slb server FW1 10.10.2.20  
    port 0 tcp  
        health-check-disable  
    port 0 udp  
        health-check-disable  
    port 8080 tcp  
        health-check-disable
```

5. Define an SLB service group for all TCP traffic and bind the port template for dscp=4 under it. This service group will be used for all bypassed TCP traffic.
6. Define an SLB service group for all UDP traffic and bind the port template for dscp=4 under it. This service group will be used for all UDP traffic.
7. Define an SLB service group for all TCP traffic and bind the port template for dscp=6 under it. This service group will be used for all decrypted TCP traffic.
8. Define an SLB service group for all TCP traffic and bind the port template for dscp=6 under it. This service group will be used for all decrypted TCP traffic.
9. Configure the client-SSL template. You must complete the following tasks:
 - a. Define Access-List to identify traffic of interest.
 - b. Import a proxied CA certificate, and the certificate's private key. This certificate must be trusted by clients.
 - c. Define two SLB port templates for marking dscp values. In this example, we use dscp=6 for marking decrypted traffic and dscp=4 for marking all bypassed

traffic.

```
!  
slb template port decrypt-dscp-6  
    dscp 6  
!  
slb template port non-ssli-dscp-4  
    dscp 4
```

- d. Create an SLB real server for a path through the security device for all TCP and UDP traffic.

```
!  
slb server FW1 10.10.2.20  
    port 0 tcp  
        health-check-disable  
    port 0 udp  
        health-check-disable  
    port 8080 tcp  
        health-check-disable
```

- e. Define an SLB service group for all TCP traffic and bind the port template for dscp=4 under it. This service group will be used for all bypassed TCP traffic.
- f. Define an SLB service group for all UDP traffic and bind the port template for dscp=4 under it. This service group will be used for all UDP traffic.
- g. Define an SLB service group for all TCP traffic and bind the port template for dscp=6 under it. This service group will be used for all decrypted TCP traffic.
- h. Define an SLB service group for all TCP traffic and bind the port template for dscp=6 under it. This service group will be used for all decrypted TCP traffic.
- i. Configure the client-SSL template. You must complete the following tasks:
- i. Enable SSL Insight support.
 - ii. Add the proxied CA certificate.
 - iii. Add the CA certificate's private key.
 - iv. Bind the service-group for bypassed TCP traffic.
- j. Configure a wildcard VIP to capture all client traffic, and add a wildcard ssl-Proxy vPort under it, along with wildcard TCP, UDP and others vPorts.

- k. Enable promiscuous VIP mode on the Ethernet interface that is connected to the clients' network. This is required by the wild-card VIP.
10. Configure a wildcard VIP to capture all client traffic, and add a wildcard ssl-Proxy vPort under it, along with wildcard TCP, UDP and others vPorts.
11. Enable promiscuous VIP mode on the Ethernet interface that is connected to the clients' network. This is required by the wild-card VIP.

ACOS_encrypt Configuration

1. On ACOS_encrypt, configure two access lists. The first, access-list 101, filters decrypted traffic arriving with dscp=6, and the second, access-list 102, filters all other traffic arriving with dscp=4.

```
ACOS_encrypt(config)# access-list 101 permit ip any any dscp 6
ACOS_encrypt(config)# access-list 102 permit ip any any dscp 4
```

2. Create vlan 30 and specify its VE interface to be on a subnet that links to the Internet default gateway.

```
ACOS_encrypt(config)# vlan 30
ACOS_encrypt(config-vlan:30)# untagged ethernet 1
ACOS_encrypt(config-vlan:30)#router-interface ve 30
ACOS_encrypt(config)# interface ve 30
ACOS_encrypt(config-if:ve:30)# ip address 10.10.3.20 255.255.255.0
```

3. Configure a VE interface for vlan 20 and configure ip allow-promiscuous-vip under it.

```
ACOS_encrypt(config)# vlan 20
ACOS_encrypt(config-vlan:20)# untagged ethernet 2
ACOS_encrypt(config-vlan:20)#router-interface ve 20
ACOS_encrypt(config)# interface ve 20
ACOS_encrypt(config-if:ve:20)# ip address 10.10.2.20 255.255.255.0
ACOS_encrypt(config-if:ve:20)# ip allow-promiscuous-vip
```

4. The outside ACOS needs to support forward-proxy services for SSLi. The server-ssl template: Server-SSL enables this capability when bound to a virtual server.

```
ACOS_encrypt(config)# slb template server-ssl Server-SSL
ACOS_encrypt(config-server ssl)# forward-proxy-enable
```

5. Configure the SLB real server, Gateway, on the IP subnet that links to the default gateway. Configure the server with the wildcard port for TCP sessions and disable health check.

```
ACOS_encrypt(config)# slb server Gateway 10.10.3.1
ACOS_encrypt(config-real server)# port 0 tcp
ACOS_encrypt(config-real server-node port)# health-check-disable
ACOS_encrypt(config-real server)# port 443 tcp
ACOS_encrypt(config-real server-node port)# health-check-disable
ACOS_encrypt(config-real server)# port 0 udp
ACOS_encrypt(config-real server-node port)# health-check-disable
```

6. Configure TCP and UDP service groups which have Gateway as their only member.

```
ACOS_encrypt(config)# slb service-group Outbound-TCP tcp
ACOS_encrypt(config-slb svc group)# member Gateway 0
ACOS_encrypt(config)# slb service-group Outbound-UDP tcp
ACOS_encrypt(config-slb svc group)# member Gateway 0
ACOS_encrypt(config)# slb service-group Outbound-SSLi-8080 tcp
ACOS_encrypt(config-slb svc group)# member Gateway 443
```

7. Create the virtual server, Outside_nonSSLi_VIP, to handle non-SSL and bypassed TCP connections.

```
ACOS_encrypt(config)# slb virtual-server Outside_nonSSLi_VIP 0.0.0.0
acl 102
ACOS_encrypt(config-slb vserver)# port 0 tcp
ACOS_encrypt(config-slb vserver-vport)# service-group Outbound-TCP
ACOS_encrypt(config-slb vserver-vport)# no-dest-nat
ACOS_encrypt(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS_encrypt(config-slb vserver)# port 0 udp
ACOS_encrypt(config-slb vserver-vport)# service-group Outbound-UDP
ACOS_encrypt(config-slb vserver-vport)# no-dest-nat
ACOS_encrypt(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS_encrypt(config-slb vserver)# port 0 others
ACOS_encrypt(config-slb vserver-vport)# service-group Outbound-UDP
ACOS_encrypt(config-slb vserver-vport)# no-dest-nat
ACOS_encrypt(config-slb vserver-vport)# use-rcv-hop-for-resp
```

8. Create the virtual server, Encrypt_SSLi_VIP, to handle SSLi TCP connections. Bind the previously configured server-ssl template to this server to enable the

forward-proxy process.

```
ACOS_encrypt(config)# slb virtual-server Encrypt_SSli_VIP 0.0.0.0 acl
101
ACOS_encrypt(config-slb vsrver)# port 0 tcp-proxy
ACOS_encrypt(config-slb vsrver-vport)# service-group Outbound-TCP
ACOS_encrypt(config-slb vsrver-vport)# template server-ssl Server-SSL
ACOS_encrypt(config-slb vsrver-vport)# no-dest-nat
ACOS_encrypt(config-slb vsrver-vport)# use-rcv-hop-for-resp
ACOS_encrypt(config-slb vsrver-vport)# port 8080 http
ACOS_encrypt(config-slb vsrver-vport)# name PORT_8080
ACOS_encrypt(config-slb vsrver-vport)# service-group Outbound-SSli-
8080
ACOS_encrypt(config-slb vsrver-vport)# template server-ssl Server-SSL
ACOS_encrypt(config-slb vsrver-vport)# no-dest-nat port-translation
ACOS_encrypt(config-slb vsrver-vport)# use-rcv-hop-for-resp
```

ACOS_decrypt Configuration

1. On ACOS_decrypt, configure an access list to permit traffic arriving from the clients.

```
ACOS_decrypt(config)# access-list 101 permit ip 10.10.1.0 0.0.0.255 any
```

2. Create vlan 10 on Ethernet 1 port for connecting the clients' network to ACOS_decrypt and configure a VE interface 10 with an IP address on the same subnet as the clients. Lastly, configure ip allow-promiscuous-vip under the VE interface.

```
ACOS_decrypt(config)# vlan 10
ACOS_decrypt(config-vlan:10)# untagged ethernet 1
ACOS_decrypt(config-vlan:10)#router-interface ve 10
ACOS_decrypt(config)# interface ve 10
ACOS_decrypt(config-if:ve:10)# ip address 10.10.1.10 255.255.255.0
ACOS_decrypt(config-if:ve:10)# ip allow-promiscuous-vip
```

3. Create vlan 20 on Ethernet 2 port for connecting the security device to ACOS_decrypt and configure a VE interface 20.

```
ACOS_decrypt(config)# vlan 20
ACOS_decrypt(config-vlan:20)# untagged ethernet 2
ACOS_decrypt(config-vlan:20)#router-interface ve 20
```

```
ACOS_decrypt(config)# interface ve 20
ACOS_decrypt(config-if:ve:20)# ip address 10.10.2.10 255.255.255.0
```

4. Create the SLB real server, FW1 with IP address 10.10.2.20. This would match the IP address assigned to ve 20 on ACOS_encrypt. Enable wildcard ports for TCP and UDP. Disable health check.

NOTE: Since port is wildcard port 0, health check will fail if enabled.

```
ACOS_decrypt(config)# slb server FW1 10.10.2.20
ACOS_decrypt(config-real server)# port 0 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server)# port 0 udp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server)# port 8080 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
```

5. Define port templates for setting DSCP=6 and DSCP=4.

```
ACOS_decrypt(config)# slb template port decrypt-dscp-6
ACOS_decrypt(config-rport)# dscp 6
ACOS_decrypt(config)# slb template port non-ssl-dscp-4
ACOS_decrypt(config-rport)# dscp 4
```

6. Define service-groups for the security device for all bypassed traffic by binding the non-ssl-dscp-4 port template to server port memberships:

```
ACOS_decrypt(config)# slb service-group Outbound-UDP udp
ACOS_decrypt(config-slb svc group)# member FW1 0
ACOS_decrypt(config-slb svc group-member:0)# template non-ssli-dscp-4
ACOS_decrypt(config)# slb service-group Outbound-TCP tcp
ACOS_decrypt(config-slb svc group)# member FW1 0
ACOS_decrypt(config-slb svc group-member:0)# template non-ssli-dscp-4
```

7. Define service-groups for the security device for all decrypted traffic by binding the decrypt-dscp-6 port template to server port memberships:

```
ACOS_decrypt(config)# slb service-group Outbound-SSLi-0 tcp
ACOS_decrypt(config-slb svc group)# member FW1 0
ACOS_decrypt(config-slb svc group-member:0)# template decrypt-dscp-6
ACOS_decrypt(config)# slb service-group Outbound-SSLi-443 tcp
ACOS_decrypt(config-slb svc group)# member FW1 8080
```

```
ACOS_decrypt(config-slb svc group-member:8080)# template decrypt-dscp-6
```

8. Configure a client-ssl template: Client-SSL provisioned with the certificate and private key needed to proxy a certificate that would be accepted by the clients seeking an SSL session with the remote servers. Enable forward-proxy and non-SSL bypass.

When the SSL client is enabled for forward proxy, ACOS processes intercepted traffic by default as if it were an HTTPS session. It is therefore necessary to disable the default HTTPS processing for non-HTTP protocol sessions. The `non-ssl-bypass` command disables this processing for non-HTTP protocols.

```
ACOS_decrypt(config)# slb template client-ssl Client-SSL
ACOS_decrypt(config-client ssl)# forward-proxy-ca-certificate
Cert123.pem key key123
ACOS_decrypt(config-client ssl)# forward-proxy-enable
ACOS_decrypt(config-client ssl)# non-ssl-bypass service-group Outbound-
TCP
```

9. Create a virtual server `decrypt_SSli_VIP` for `ACOS_decrypt` facing the clients. Enable its wildcard port for SSL-proxy service, disable destination NAT, and bind the previously configured service groups and client-ssl template to it.

When you enable SSL-proxy service on the wildcard VIP, it will dynamically proxy for any protocol running over SSL; in other words all SSL protocols running over SSL will be intercepted.

- a. Disable destination NAT to preserve the destination IP address on load-balanced traffic.
- b. Bind the wildcard SSL proxy port to the service-group named `Outbound-SSli-0` to provide a path to the inspection device and the outside ACOS. Also bind an HTTPs vport to the service-group `Outbound-SSli-443`.
- c. Bind the wildcard SSL-proxy port to the SSL client template named `Client-SSL` to enable forward proxy services (SSLi) on that port.
- d. Bind the `Outbound-SSli-443` port to the SSL client template named `Client-SSL` to enable forward proxy services (SSLi) on that port.

```
ACOS_decrypt(config)# slb virtual-server decrypt_SSli_VIP 0.0.0.0
acl 101
ACOS_decrypt(config-slb vserver)# port 0 ssl-proxy
```

```
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat
ACOS_decrypt(config-slb vserver-vport)# service-group Outbound-SSLi-0
ACOS_decrypt(config-slb vserver-vport)# template client-ssl Client-SSL
ACOS_decrypt(config-slb vserver)# port 443 https
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat port translation
ACOS_decrypt(config-slb vserver-vport)# service-group Outbound-SSLi-443
ACOS_decrypt(config-slb vserver-vport)# template client-ssl Client-SSL
```

10. Enable wildcard udp and others ports and provide service groups for them.

```
ACOS_decrypt(config-slb vserver)# port 0 udp
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat
ACOS_decrypt(config-slb vserver-vport)# service-group Outbound-UDP
ACOS_decrypt(config-slb vserver)# port 0 others
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat
ACOS_decrypt(config-slb vserver-vport)# service-group Outbound-UDP
```

Consolidated Configuration

ACOS_decrypt

```
!
access-list 101 permit ip 10.10.1.0 0.0.0.255 any
!
interface ethernet 1
    enable
!
interface ethernet 2
    enable
!
vlan 10
    untagged ethernet 1
    router-interface ve 10
!
vlan 20
```

```
untagged ethernet 2
router-interface ve 20
!
interface ve 10
  ip address 10.10.1.10 255.255.255.0
  ip allow-promiscuous-vip
!
interface ve 20
  ip address 10.10.2.10 255.255.255.0
!
slb template port decrypt-dscp-6
  dscp 6
!
slb template port non-ssli-dscp-4
  dscp 4
!
slb server FW1 10.10.2.20
  port 0 tcp
    health-check-disable
  port 0 udp
    health-check-disable
  port 8080 tcp
    health-check-disable
!
!
slb service-group Outbound-TCP tcp
  member FW1 0
    template non-ssli-dscp-4
!
slb service-group Outbound-UDP udp
  member FW1 0
    template non-ssli-dscp-4
!
slb service-group Outbound-SSLi-0 tcp
  member FW1 0
    template decrypt-dscp-6
!
slb service-group Outbound-SSLi-443 tcp
  member FW1 8080
    template decrypt-dscp-6
```

```
!  
slb template client-ssl Client-SSL  
  forward-proxy-ca-certificate Cert123.pem key key123  
  forward-proxy-enable  
  non-ssl-bypass service-group Outbound-TCP  
!  
slb virtual-server Clients_VIP 0.0.0.0 acl 101  
  port 0 ssl-proxy  
    no-dest-nat  
    service-group Outbound-SSLi-0  
    template client-ssl Client-SSL  
  port 0 udp  
    no-dest-nat  
    service-group Outbound-UDP  
  port 0 others  
    no-dest-nat  
    service-group Outbound-UDP  
  port 443 https  
    no-dest-nat port-translation  
    service-group Outbound-SSLi-443  
    template client-ssl Client-SSL  
!  
end
```

ACOS_encrypt

```
!  
access-list 101 permit ip any any dscp 6  
!  
access-list 102 permit ip any any dscp 4  
!  
vlan 20  
  untagged ethernet 2  
  router-interface ve 20  
!  
vlan 30  
  untagged ethernet 1  
  router-interface ve 30  
!  
ip route 0.0.0.0 /0 10.10.3.1
```

```
!  
interface ethernet 1 enable  
!  
interface ve 20  
    ip address 10.10.2.20 255.255.255.0  
    ip allow-promiscuous-vip  
!  
interface ve 30  
    ip address 10.10.3.20 255.255.255.0  
!  
slb server Gateway 10.10.3.1  
    port 443 tcp  
        health-check-disable  
    port 0 tcp  
        health-check-disable  
    port 0 udp  
        health-check-disable po  
!  
slb service-group Outbound-TCP tcp  
    member Gateway 0  
!  
slb service-group Outbound-UDP udp  
    member Gateway 0  
!  
slb service-group Outbound-SSLi-8080 tcp  
    member Gateway 443  
!  
slb template server-ssl Server-SSL  
    forward-proxy-enable  
!  
slb virtual-server Outside_nonSSLi_VIP 0.0.0.0 acl 102  
    port 0 tcp  
        service-group Outbound-TCP  
        no-dest-nat  
        use-rcv-hop-for-resp  
    port 0 udp  
        service-group Outbound-UDP  
        no-dest-nat  
        use-rcv-hop-for-resp  
    port 0 others
```

```
service-group Outbound-UDP
no-dest-nat
use-rcv-hop-for-resp
!
slb virtual-server Encrypt_SSLi_VIP 0.0.0.0 acl 101
port 0 tcp-proxy
service-group Outbound-TCP
template server-ssl Server-SSL
no-dest-nat
use-rcv-hop-for-resp
port 8080 http
name PORT_8080
service-group Outbound-SSLi-8080
template server-ssl Server-SSL
no-dest-nat port-translation
use-rcv-hop-for-resp
!
end
```

DSCP Inspection

You can set the Differentiated Services Code Point (DSCP) for decrypted and bypassed traffic by using the **forward-proxy-decrypt dscp** command without changing service groups. The configured DSCP is applied to the IP header of the decrypted or bypassed traffic.

NOTE: If the service group has a template with DSCP configured, the **forward-proxy-decrypt dscp** command takes precedence.

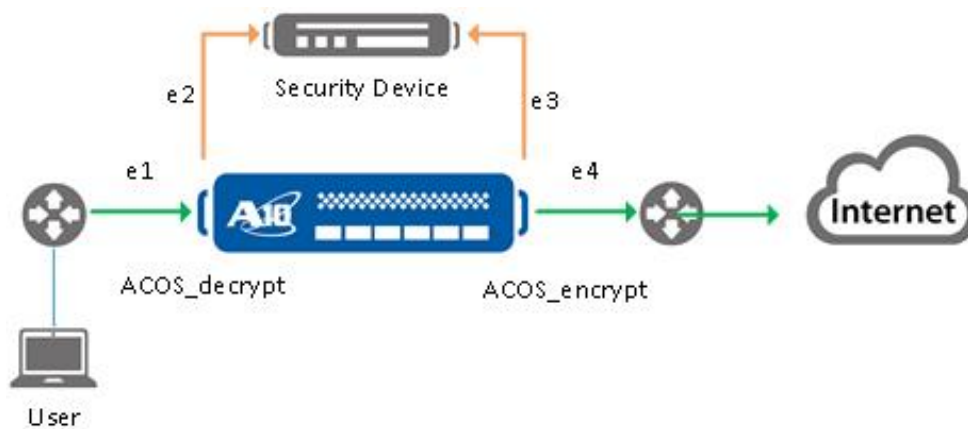
Deployment Example

The following configuration example includes a single SSLi device with two partitions. ACOS_encrypt and ACOS_decrypt are the two partitions. This L2 configuration example uses the DSCP argument in the client SSLi template to handle decrypted and bypassed traffic. The configuration uses DSCP tagging to enable **ACOS_decrypt** to communicate to **ACOS_encrypt** about which traffic was decrypted, and thus needs to be re-encrypted. The DSCP tagging is achieved with the **forward-proxy-decrypt**

dscp command and is referenced in the service groups that handle decrypted traffic. As the traffic is decrypted, it gets a DSCP 6 tag. An access-list is configured for the **ACOS_encrypt** partition that catches traffic with this tag. All other traffic (without a DSCP 6 tag) is switched by ACOS on the **ACOS_encrypt** partition. DSCP enables us to avoid rewriting the port when decrypting SSL traffic.

[Figure 16](#) is an example deployment. In the following example deployment, the client network is connected through a layer 2 switch to the ACOS device. The ACOS device, which has two partitions, is in turn connected to a security device for traffic inspection purposes. The security device is a L2 transparent device that preserves the L2 header while processing the traffic flows. The ACOS device is then connected through a layer 2 switch to the Internet. Interfaces 1 and 2 belong to the **ACOS_decrypt** partition. Interfaces 3 and 4 belong to the **ACOS_encrypt** partition.

Figure 16 : Single-Device Double-Partition SSLi Configuration with DSCP



Traffic WorkFlow

The traffic flow from the client network is sent to the **ACOS_decrypt** partition on the e1 interface. The traffic flow is decrypted by the **ACOS_decrypt** partition. The traffic from the **ACOS_decrypt** partition is directed to the security device in the forward direction. From the security device, the traffic is directed to the **ACOS_encrypt** partition on the e3 interface. The **ACOS_encrypt** partition re-encrypts the traffic and forwards the traffic to the gateway by using normal SLB operation.

The traffic flow is shown as follows:

HTTPS/443 >>Traffic Decrypted in **ACOS_decrypt** >>HTTP/443 through security devices
>>Traffic Re-encrypted in **ACOS_encrypt** >>HTTPS/443 to Internet

The following list includes information about the other kinds of traffic flow:

- **UDP/ICMP/Other traffic**—This traffic is not caught by any VIP configuration and is just switched by ACOS.
- **HTTPS on port 443**—This traffic is decrypted in the ACOS_decrypt partition, tagged with DSCP 6, and re-encrypted by the ACOS_encrypt partition by the "port 0 tcp-proxy" vPort.
- **HTTP on port 80**—Traffic is caught by the wildcard VIP on ACOS_decrypt, and is only called out in case DLP configuration needs to be added. Otherwise the "port 80 http" vPort is omitted.
- **TCP+SSL on any other port**—Traffic is caught by the wildcard VIP in ACOS_decrypt, tagged with DSCP 6, and re-encrypted by the ACOS_encrypt partition.
- **TCP on any other port**—Traffic is caught by the wildcard VIP in **ACOS_decrypt**, but since it is not SSL it is not tagged with DSCP 6. When it hits the **ACOS_encrypt** partition, there is no DSCP tag, so the wildcard VIP doesn't see it and it gets switched by ACOS. In the client-ssl template in **ACOS_decrypt**, non-ssl traffic is sent to the **SG_SSLi_TCP-bypass** service-group, via the "non-ssl-bypass service-group" command in the client-ssl template.

NOTE: The static port intercept for the HTTP protocol is required when you have configured either HTTP policy or the ICAP feature. Otherwise, you can remove the static port intercept for each virtual server.

CLI Configuration

To avoid a duplicate MAC address because of the VLAN that is shared, add the global command of **system ve-mac-scheme system-mac**.

```
ACOS(config)# system ve-mac-scheme system-mac
```

Assign an IP address and default gateway to the management interface:

```
ACOS(config)# interface management
ACOS(config-if:management)# ip address 10.10.30.15 255.255.255.0
ACOS(config-if:management)# ip control-apps-use-mgmt-port
```

```
ACOS(config-if:management)# ip default-gateway 10.10.30.1
ACOS(config-if:management)# exit
```

Create the two partitions of ACOS_decrypt and ACOS_encrypt:

```
ACOS(config)# partition ACOS_decrypt id 1 application-type adc
ACOS(config)# partition ACOS_encrypt id 2 application-type adc
```

ACOS_decrypt Configuration

The work-flow for configuring the **ACOS_decrypt** partition includes the following:

Configuring the Default VLAN

1. Configure the default VLAN. Bind ethernet ports 1 and 2 to the VLAN. Also, bind a virtual interface ve to the VLAN. A VE is required in order to configure an IP address on a VLAN. In this example, a default VLAN of 850 is configured.

```
ACOS_decrypt(config)# vlan 850
ACOS_decrypt(config-vlan:850)# untagged ethernet 1 to 2
ACOS_decrypt(config-vlan:850)# router-interface ve 850
ACOS_decrypt(config-vlan:850)# exit
```

2. Enable the ethernet interfaces 1 and 2 that are associated with the VLAN.

```
ACOS_decrypt(config)# interface ethernet 1
ACOS_decrypt(config-if:ethernet:1)# enable
ACOS_decrypt(config-if:ethernet:1)# exit
ACOS_decrypt(config)# interface ethernet 2
ACOS_decrypt(config-if:ethernet:2)# enable
ACOS_decrypt(config-if:ethernet:2)# exit
```

3. Verify the operational state of the interfaces by running the show interfaces command.

```
ACOS_decrypt(config)# show interfaces brief
```

Configuring the ACL

1. Configure the access lists. Configure the ACL to drop UDP-based traffic from any source to any destination on ports 80 and 443. If the traffic is IP-based, it is allowed to be forwarded.

```
ACOS_decrypt(config)# access-list 101 deny udp any any eq 80
```

```
ACOS_decrypt(config)# access-list 101 deny udp any any eq 443
ACOS_decrypt(config)# access-list 101 permit ip any any
```

2. Configure the ACL to permit IP traffic from any source to any destination for the VLAN 850:

```
ACOS_decrypt(config)# access-list 190 permit ip any any vlan 850
```

Configuring Network IP Addresses for Untagged VLANs

On the virtual interface 850, enable promiscuous VIP support. When you enable promiscuous VIP support on a VE, the option is automatically enabled on each ethernet data port in the VE. Provision the virtual interfaces to allow promiscuous IP in order to subject traffic to the rules enabled on each interface. In addition, assign an IP address and a default gateway to the VLAN. In this example, we assign the IP address and gateway to interface ve 850. Additionally, bind ACL 101 to the interface for all inbound traffic.

```
ACOS_decrypt(config)# interface ve 850
ACOS_decrypt(config-if:ve850)# access-list 101 in
ACOS_decrypt(config-if:ve850)# ip address 10.10.10.98 255.255.255.0
ACOS_decrypt(config-if:ve850)# ip allow-promiscuous-vip
ACOS_decrypt(config-if:ve850)# exit
```

Configuring the Security Device

1. Configure the server GW and its ports. Configure ports 0, 80, and 443 for TCP traffic. Disable health check for each port.

```
ACOS_decrypt(config)# slb server gw 10.10.10.1
ACOS_decrypt(config-real server)# health-check-disable
ACOS_decrypt(config-real server)# port 0 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit

ACOS_decrypt(config-real server)# port 80 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit

ACOS_decrypt(config-real server)# port 443 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit
```

2. Configure the server service group called **SG_SSLi_HTTP** of type TCP. Associate GW and port 80 with the service group.

```
ACOS_decrypt(config)# slb service-group SG_SSLi_HTTP tcp
ACOS_decrypt(config-slb svc group)# member gw 80
ACOS_decrypt(config-slb svc group-member:80)# exit
ACOS_decrypt(config-slb svc group)# exit
```

3. Configure the server service group called **SG_SSLi_HTTPS** of type TCP. Associate GW and port 443 with the service group.

```
ACOS_decrypt(config)# slb service-group SG_SSLi_HTTPS tcp
ACOS_decrypt(config-slb svc group)# member gw 443
ACOS_decrypt(config-slb svc group-member:443)# exit
ACOS_decrypt(config-slb svc group)# exit
```

4. Configure the server service group called **SG_SSLi_TCP** of type TCP. Associate GW and port 0 with the service group.

```
ACOS_decrypt(config)# slb service-group SG_SSLi_TCP tcp
ACOS_decrypt(config-slb svc group)# member gw 0
ACOS_decrypt(config-slb svc group-member:0)# exit
ACOS_decrypt(config-slb svc group)# exit
```

5. Configure the server service group called **SG_SSLi_TCP-bypass** of type TCP. Associate GW and port 0 with the service group.

```
ACOS_decrypt(config)# slb service-group SG_SSLi_TCP-bypass tcp
ACOS_decrypt(config-slb svc group)# member gw 0
ACOS_decrypt(config-slb svc group-member:0)# exit
ACOS_decrypt(config-slb svc group)# exit
```

Configuring the SSLi Services for ACOS_decrypt Partition

1. Configure the client SSL template by specifying the SSLi self-signed certificate and private key. For all encrypted traffic, add a DSCP tag of 6. For all bypassed traffic, add a DSCP tag of 1.

```
ACOS_decrypt(config)# slb template client-ssl SSLi
ACOS_decrypt(config-client ssl)# forward-proxy-ca-certificate
Cert123.pem key key123 chain-cert abc.home
ACOS_decrypt(config-client ssl)# forward-proxy-decrypt dscp 6 1
ACOS_decrypt(config-client ssl)# forward-proxy-enable
```

```
ACOS_decrypt(config-client ssl)# forward-proxy-failsafe-disable
```

2. When the SSL client is enabled for forward proxy, ACOS processes intercept traffic by default as if it were an HTTPS session. It is therefore necessary to disable the default HTTPS processing for non-HTTP protocol sessions. The **non-ssl-bypass** command disables this processing for non-HTTP protocols.

```
ACOS_decrypt(config-client ssl)# non-ssl-bypass service-group SG_SSLi_TCP-bypass
```

Configuring Handling of Incoming Traffic

1. Create a virtual server called **ACOS_decrypt** for the **ACOS_decrypt** partition facing the clients.
2. Enable its wildcard port for SSL-proxy service, disable destination NAT, and bind the previously configured service groups and client-ssl template to it. The ACL 190 is bound to the wildcard VIP. When you enable SSL-proxy service on the wildcard VIP, it will dynamically proxy for any protocol running over SSL; in other words all SSL protocols running over SSL will be intercepted.

```
ACOS_decrypt(config)# slb virtual-server ACOS_decrypt 0.0.0.0 acl 190
```

3. Bind the wildcard SSL proxy port to the service-group named **SG_SSLi_TCP** to provide a path to the inspection device and the **ACOS_encrypt** partition. Bind the wildcard SSL-proxy port to the SSL client template named **SSLi** to enable forward proxy services (SSLi) on that port.

```
ACOS_decrypt(config-slb vsrver)# port 0 ssl-proxy
ACOS_decrypt(config-slb vsrver-vport)# service-group SG_SSLi_TCP
ACOS_decrypt(config-slb vsrver-vport)# template client-ssl SSLi
ACOS_decrypt(config-slb vsrver-vport)# no-dest-nat
ACOS_decrypt(config-slb vsrver-vport)# exit
```

4. Bind an HTTPs vport to the service-group **SG_SSLi_HTTPS**. Bind the Outbound-SSLi-443 port to the SSL client template named **SSLi** to enable forward proxy services (SSLi) on that port.

```
ACOS_decrypt(config-slb vsrver)# port 443 https
ACOS_decrypt(config-slb vsrver-vport)# service-group SG_SSLi_HTTPS
ACOS_decrypt(config-slb vsrver-vport)# template client-ssl SSLi
ACOS_decrypt(config-slb vsrver-vport)# no-dest-nat
ACOS_decrypt(config-slb vsrver-vport)# exit
```

5. Associate port 80 of type HTTP with service group **SG_SSLi_HTTP**. Disable destination NAT.

```
ACOS_decrypt(config-slb vserver)# port 80 http
ACOS_decrypt(config-slb vserver-vport)# service-group SG_SSLi_HTTP
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat
ACOS_decrypt(config-slb vserver-vport)# exit
ACOS_decrypt(config-slb vserver)# exit
```

ACOS_encrypt Configuration

The work-flow for configuring the **ACOS_encrypt** partition includes the following:

Configuring the ACL

Configure two access lists. The first, access-list 191, filters decrypted traffic arriving with dscp=6, and the second, access-list 192, filters all other traffic arriving with dscp=1.

```
ACOS[ACOS_encrypt](config)# access-list 191 permit ip any any dscp 6
ACOS[ACOS_encrypt](config)# access-list 192 permit ip any any dscp 1
```

Configuring the Default VLAN

Configure the default VLAN. Bind ethernet ports 3 and 4 to the VLAN. Also, bind a virtual interface ve to the VLAN. In this example, a default VLAN of 860 is configured.

```
ACOS[ACOS_encrypt](config)# vlan 860
ACOS[ACOS_encrypt](config-vlan:860)# untagged ethernet 3 to 4
ACOS[ACOS_encrypt](config-vlan:860)# router-interface ve 860
ACOS[ACOS_encrypt](config-vlan:860)# exit

ACOS[ACOS_encrypt](config)# interface ethernet 3
ACOS[ACOS_encrypt](config-if:ethernet:3)# enable
ACOS[ACOS_encrypt](config-if:ethernet:3)# exit
ACOS[ACOS_encrypt](config)# interface ethernet 4
ACOS[ACOS_encrypt](config-if:ethernet:4)# enable
ACOS[ACOS_encrypt](config-if:ethernet:4)# exit
```

Configuring Network IP Addresses for the VLAN

1. On the virtual interface 860, enable promiscuous VIP support. When you enable

promiscuous VIP support on a VE, the option is automatically enabled on each ethernet data port in the VE. Provision the virtual interfaces to allow promiscuous IP in order to subject traffic to the rules enabled on each interface. In addition, assign an IP address and a default gateway to the VLAN. In this example, we assign the IP address and gateway to interface ve 860.

```
ACOS[ACOS_encrypt](config)# interface ve 860
ACOS[ACOS_encrypt](config-if:ve860)# ip address 10.10.10.99
255.255.255.0
ACOS[ACOS_encrypt](config-if:ve860)# ip allow-promiscuous-vip
ACOS[ACOS_encrypt](config-if:ve860)# exit
```

2. Enable the ethernet interfaces 3 and 4 that are associated with the VLAN.

```
ACOS[ACOS_encrypt](config)# interface ethernet 3
ACOS[ACOS_encrypt](config-if:ethernet:3)# enable
ACOS[ACOS_encrypt](config-if:ethernet:3)# exit
ACOS[ACOS_encrypt](config)# interface ethernet 4
ACOS[ACOS_encrypt](config-if:ethernet:4)# enable
ACOS[ACOS_encrypt](config-if:ethernet:4)# exit
```

3. Verify the operational state of the interfaces by running the show interfaces command.

```
ACOS[ACOS_encrypt](config)# show interfaces brief
```

Configuring the Security Device

1. Configure the server GW and its ports.

```
ACOS[ACOS_encrypt](config)# slb server gw 10.10.10.1
ACOS[ACOS_encrypt](config-real server)# health-check-disable

ACOS[ACOS_encrypt](config-real server)# port 0 tcp
ACOS[ACOS_encrypt](config-real server-node port)# health-check-
disable
ACOS[ACOS_encrypt](config-real server-node port)# exit

ACOS[ACOS_encrypt](config-real server)# port 443 tcp
ACOS[ACOS_encrypt](config-real server-node port)# health-check-
disable
ACOS[ACOS_encrypt](config-real server-node port)# exit
```


2. Configure the server service group called **SG_SSLi_HTTP** of type TCP. Associate **GW** and port 443 with the service group.

```
ACOS[ACOS_encrypt](config)# slb service-group SG_SSLi_HTTP tcp
ACOS[ACOS_encrypt](config-slb svc group)# member gw 443
ACOS[ACOS_encrypt](config-slb svc group-member:443)# exit
ACOS[ACOS_encrypt](config-slb svc group)# exit
```

3. Configure the server service group called **SG_SSLi_TCP** of type TCP. Associate **GW** and port 0 with the service group.

```
ACOS[ACOS_encrypt](config)# slb service-group SG_SSLi_TCP
ACOS[ACOS_encrypt](config-slb svc group)# member gw 0
ACOS[ACOS_encrypt](config-slb svc group-member:0)# exit
ACOS[ACOS_encrypt](config-slb svc group)# exit
```

Configuring the SSLi Services for ACOS_encrypt Partition

Create an SSL server template on the **ACOS_encrypt** partition so that the VIP can operate as an SSL client and handshake with the enterprise server. Enable forward proxy services on the template to enable SSLi operation on the VIP.

```
ACOS[ACOS_encrypt](config)# slb template server-ssl SSLi
ACOS[ACOS_encrypt](config-server ssl)# forward-proxy-enable
ACOS[ACOS_encrypt](config-server ssl)# exit
```

Configuring Handling of Outgoing Traffic

1. Create the virtual server **ACOS_encrypt** filter incoming traffic with a tag of **dscp=6**.

```
ACOS[ACOS_encrypt](config)# slb virtual-server ACOS_encrypt 0.0.0.0 acl 191
```
2. Bind the virtual port **port 0 tcp-proxy** to the service group **SG_SSLi_TCP** and the SSLi server template. Bind the virtual port **port 443 http** to the service group **SG_SSLi_HTTP** and the SSLi server template. Disable destination NAT to preserve the destination IP address on load-balanced traffic. The HTTPS traffic tagged with DSCP=6 arriving at the vport port 0 tcp-proxy is re-encrypted.

```
ACOS[ACOS_encrypt](config-slb vsrver)# port 0 tcp-proxy
ACOS[ACOS_encrypt](config-slb vsrver-vport)# service-group SG_SSLi_TCP
ACOS[ACOS_encrypt](config-slb vsrver-vport)# template server-ssl SSLi
```

```

ACOS[ACOS_encrypt](config-slb vservers-vport)# no-dest-nat
ACOS[ACOS_encrypt](config-slb vservers-vport)# exit

ACOS[ACOS_encrypt](config-slb vservers)# port 443 http
ACOS[ACOS_encrypt](config-slb vservers-vport)# service-group SG_SSLi_
HTTP
ACOS[ACOS_encrypt](config-slb vservers-vport)# template server-ssl
SSLi
ACOS[ACOS_encrypt](config-slb vservers-vport)# no-dest-nat
ACOS[ACOS_encrypt](config-slb vservers-vport)# exit

```

3. Create the virtual server, **ACOS_encrypt_bypass**, to handle non-SSL and bypassed TCP connections with a tag of dscp=4.

```

ACOS[ACOS_encrypt](config)# slb virtual-server ACOS_encrypt_bypass
0.0.0.0 acl 192

```

Consolidated Configuration

```

!
system ve-mac-scheme system-mac
!
partition ACOS_decrypt id 1 application-type adc
!
partition ACOS_encrypt id 2 application-type adc

interface management
 ip address 10.10.30.15 255.255.255.0
 ip control-apps-use-mgmt-port
 ip default-gateway 10.10.30.1
!
interface ethernet 1
!
interface ethernet 2
!
interface ethernet 3
!
interface ethernet 4
!
end
active-partition ACOS_decrypt

```

```
!  
!  
access-list 101 deny udp any any eq 80  
!  
access-list 101 deny udp any any eq 443  
!  
access-list 101 permit ip any any  
!  
access-list 190 permit ip any any vlan 850  
!  
vlan 850  
    untagged ethernet 1 to 2  
    router-interface ve 850  
    name ACOS_decrypt_ingress_egress  
    user-tag ACOS_decrypt_ingress_egress  
!  
interface ethernet 1  
    name ACOS_decrypt_ingress  
    enable  
!  
interface ethernet 2  
    name ACOS_decrypt_egress  
    enable  
!  
interface ve 850  
    name ACOS_decrypt_ingress_egress  
    access-list 101 in  
    ip address 10.10.10.98 255.255.255.0  
    ip allow-promiscuous-vip  
!  
!  
slb server gw 10.10.10.1  
    health-check-disable  
    user-tag ACOS_decrypt  
    port 0 tcp  
        health-check-disable  
    port 80 tcp  
        health-check-disable  
    port 443 tcp  
        health-check-disable
```

```
!  
slb service-group SG_SSLi_HTTP tcp  
  member gw 80  
!  
slb service-group SG_SSLi_HTTPS tcp  
  member gw 443  
!  
slb service-group SG_SSLi_TCP tcp  
  member gw 0  
!  
slb service-group SG_SSLi_TCP-bypass tcp  
  member gw 0  
!  
slb template client-ssl SSLi  
  forward-proxy-ca-certificate Cert123.pem key key123 chain-cert abc.home  
  forward-proxy-decrypted dscp 6 1  
  forward-proxy-enable  
  forward-proxy-failsafe-disable  
  non-ssl-bypass service-group SG_SSLi_TCP-bypass  
!  
slb virtual-server ACOS_decrypt 0.0.0.0 acl 190  
  port 0 ssl-proxy  
    service-group SG_SSLi_TCP  
    template client-ssl SSLi  
    no-dest-nat  
  port 80 http  
    service-group SG_SSLi_HTTP  
    no-dest-nat  
  port 443 https  
    service-group SG_SSLi_HTTPS  
    template client-ssl SSLi  
    no-dest-nat  
!  
end  
active-partition ACOS_encrypt  
!  
!  
access-list 191 permit ip any any dscp 6  
!  
access-list 192 permit ip any any dscp 1
```

```
!  
vlan 860  
    untagged ethernet 3 to 4  
    router-interface ve 860  
!  
interface ethernet 3  
    enable  
!  
interface ethernet 4  
    enable  
!  
interface ve 860  
    ip address 10.10.10.99 255.255.255.0  
    ip allow-promiscuous-vip  
!  
!  
slb template server-ssl SSLi  
    forward-proxy-enable  
!  
slb server gw 10.10.10.1  
    health-check-disable  
    port 0 tcp  
        health-check-disable  
    port 443 tcp  
        health-check-disable  
!  
slb service-group SG_SSLi_HTTP tcp  
    member gw 443  
!  
slb service-group SG_SSLi_TCP tcp  
    member gw 0  
!  
slb virtual-server ACOS_encrypt 0.0.0.0 acl 191  
    port 0 tcp-proxy  
        service-group SG_SSLi_TCP  
        template server-ssl SSLi  
        no-dest-nat  
    port 443 http  
        service-group SG_SSLi_HTTP  
        template server-ssl SSLi
```

```
no-dest-nat
!
slb virtual-server ACOS_encrypt_bypass 0.0.0.0 acl 192
!
end
!Current config commit point for partition 2 is 0 & config mode is
classical-mode
TH3030S#
```

Static Port SSH Insight

This section describes how to configure static port SSH Insight. ACOS supports intercepting, decrypting, and re-encrypting Secure Shell (SSH) sessions. Only static port SSH Insight (SSH*i*) with RSA keys. This feature transparently intercept and decrypt SSH traffic so that it can be inspected for security reasons, and then re-encrypt the traffic before forwarding it to the SSH server.

For detailed information on RSA keys, see *System Administration and Configuration Guide*.

The following topics are covered:

Overview	184
Deployment	185
CLI Configuration	187

Overview

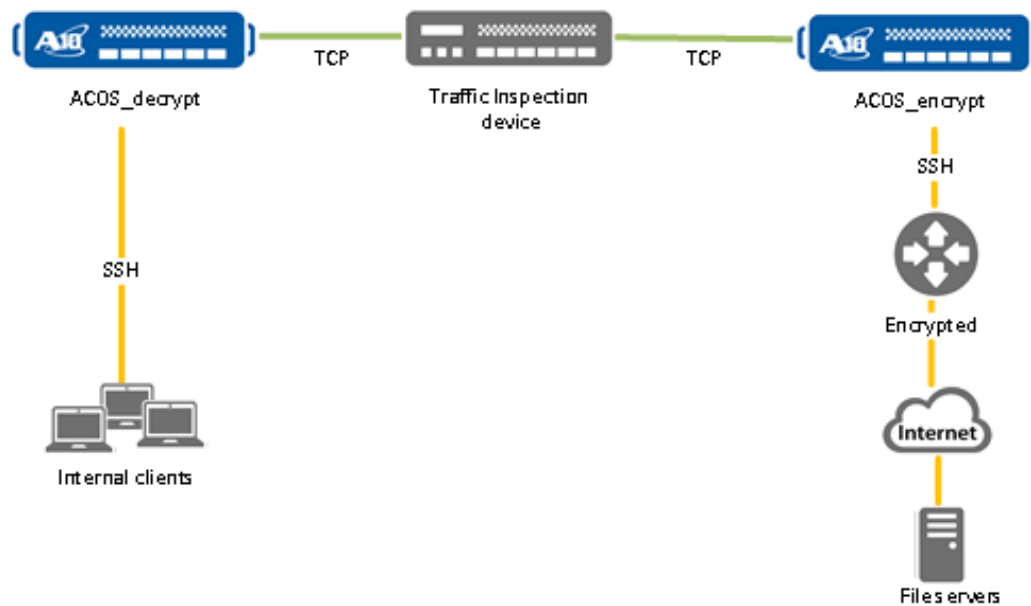
In the sample deployment as shown in [Figure 17](#), the client device is connected to the SSHi solution, which is then connected to the external gateway. The SSHi solution consists of two ACOS devices and a single security device. The ACOS device connected to the client has a partition called ACOS_decrypt. The ACOS device connected to the external gateway has a partition called ACOS_encrypt.

The following steps provide an overview of the SSHi process:

1. The client sets up an SSH connection with ACOS_decrypt and sends an encrypted request.
2. ACOS_decrypt selects a traffic inspection device, decrypts the request, and sends the request over a TCP connection to the traffic inspection device.
3. The traffic inspection device inspects the request data.
4. ACOS_encrypt encrypts the request and sends it to the outside server.
5. The server sends the encrypted reply.
6. ACOS_encrypt decrypts the reply and sends it back to the same traffic inspection device.
7. If the reply traffic is allowed by the traffic inspection device, the reply is forwarded to ACOS_decrypt.
8. ACOS_decrypt encrypts the reply and sends it to the client.

[Figure 17](#) shows the SSHi process when applied to SFTP sessions.

Figure 17 : SSHi Overview



Limitation

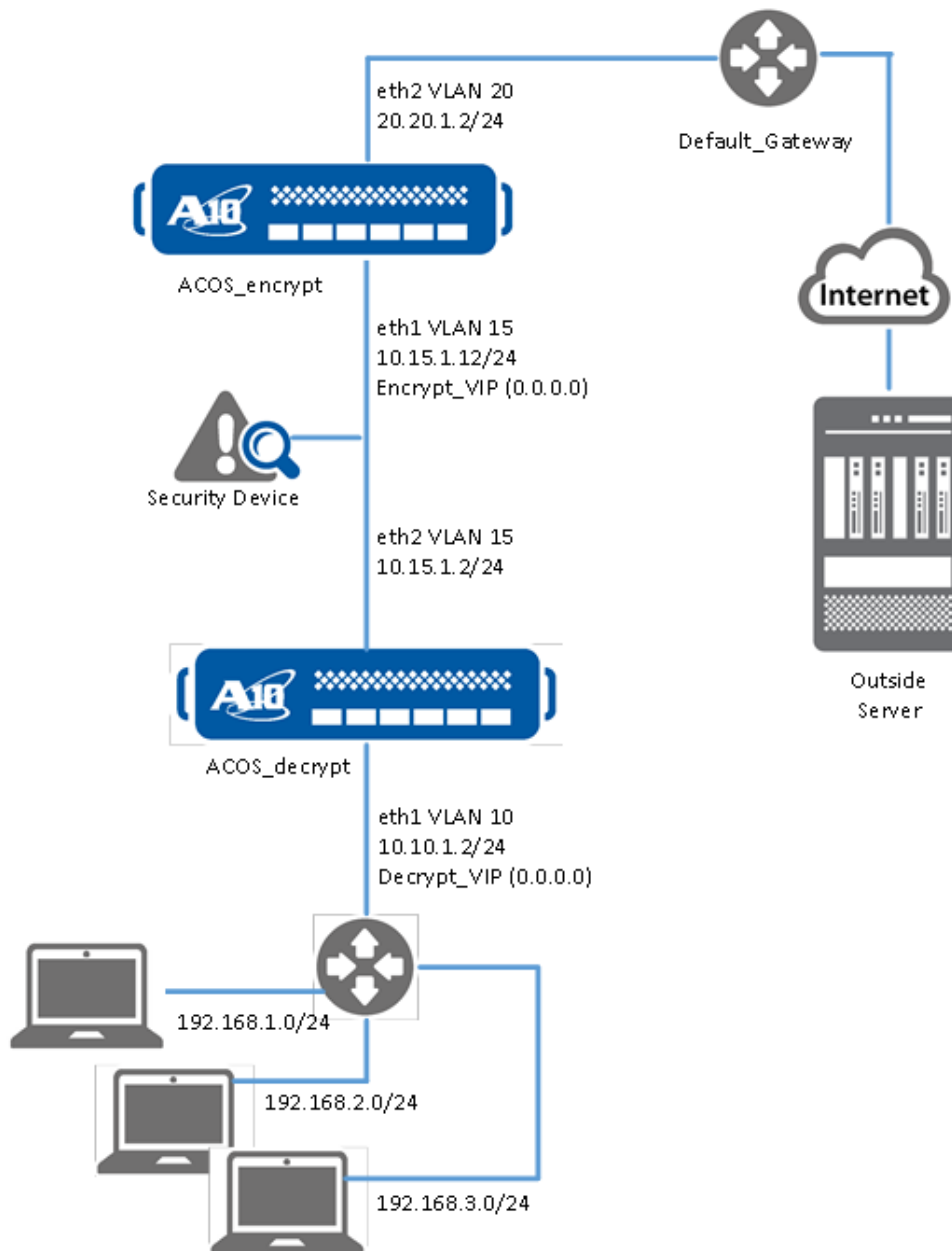
SSHi does not support the Intel QuickAssist Technology (QAT) module.

Deployment

In this example, the SSHi solution consists of two ACOS devices, each with a partition with the inspection device in between. The Decrypt_VIP SLB virtual server provides SSH Forward Proxy service that enables ACOS_decrypt to proxy for remote SSH servers and bring up SSH sessions with the clients. SSH traffic from the clients is decrypted and forwarded to the FW1_Inspect SLB real server. The FW1_Inspect SLB real server forwards decrypted SSH traffic and all other traffic to the Traffic Inspection device. In this example, the Traffic Inspection device is operating in layer-2 mode. The Encrypt_VIP wildcard VIP provides server-SSH services for decrypted traffic that enable the ACOS_encrypt to establish SSH connections with remote SSH servers through the Default_Gateway SLB real server, completing end-to-end SSH connectivity. The Default_Gateway SLB real server forwards all traffic to the Internet default gateway.

Alternately, instead of using two ACOS devices, you can use one device by creating two separate partitions, one for ACOS_decrypt and the other for ACOS_encrypt. In this case, to avoid a duplicate MAC address, add the global command of `system ve-mac-scheme system-mac` in the shared partition. See *Configuring Application Delivery Partitions* for further information. The key components of the example SSHi deployment are illustrated in [Figure 18](#).

Figure 18 : Example SSHi Static Port Network Topology



The following table provides the VLAN IDs, Virtual Ethernet (VE) Addresses, and interface configurations for the SSHi network topology illustrated in [Figure 18](#).

Table 14 :

Table 13 : SSHi Deployment Details

Partition	Tagged VLAN	VE IP Address	Ethernet Port Number
ACOS_decrypt	10	10.10.1.2 /24	eth 1
	15	10.15.1.2 /24	eth 2
ACOS_encrypt	20	20.1.1.2 /24	eth 2
	15	10.15.1.12 /24	eth 1

CLI Configuration

In order to configure SSHi for a two ACOS device single partition deployment, you must first configure the two partitions, ACOS_decrypt and ACOS_encrypt. Also, for a list of prerequisites, see [Prerequisites](#).

ACOS_decrypt Configuration

Perform the following steps for the ACOS_decrypt partition:

Step 1. Configuring the Network VLANs

For an explanation of the procedure, refer to a similar procedure discussed in [Step 1. Configuring the Network VLANs](#).

```
ACOS(config)# interface ethernet 1
ACOS(config-if:ethernet:1)# enable
ACOS(config-if:ethernet:1)# exit
!
ACOS(config)# interface ethernet 2
ACOS(config-if:ethernet:2)# enable
ACOS(config-if:ethernet:2)# exit
!
ACOS(config)# hostname ACOS_decrypt
ACOS_decrypt(config)# vlan 10
```

```
ACOS_decrypt(config-vlan:10)# tagged ethernet 1
ACOS_decrypt(config-vlan:10)#router-interface ve 10
ACOS_decrypt(config-vlan:10)# exit

ACOS_decrypt(config)# vlan 15
ACOS_decrypt(config-vlan:15)# tagged ethernet 2
ACOS_decrypt(config-vlan:15)#router-interface ve 15
ACOS_decrypt(config-vlan:15)# exit
```

Step 2. Configuring the Network IP Addresses

For an explanation of the procedure, refer to a similar procedure discussed in [Step 2. Configuring the Network IP Addresses](#).

```
ACOS_decrypt(config)# interface ve 10
ACOS_decrypt(config-if:ve10)# ip address 10.10.1.2 /24
ACOS_decrypt(config-if:ve10)# ip allow-promiscuous-vip
ACOS_decrypt(config-if:ve10)# exit

ACOS_decrypt(config)# interface ve 15
ACOS_decrypt(config-if:ve15)# ip address 10.15.1.2 /24
ACOS_decrypt(config-if:ve15)# exit
```

Step 3. Configuring the SSHi Services

1. Configure an SSHi client template, by running the following commands.

```
ACOS_decrypt(config)# slb template client-ssh SSHInsight_DecryptSide
ACOS_decrypt(config-client ssh)# forward-proxy-hostkey RSA_key_1234
ACOS_decrypt(config-client ssh)# forward-proxy-enable
ACOS_decrypt(config-client ssh)# exit
```

2. Configure a real server called **FW1_Inspect** with the IP address 10.15.1.12. This IP address matches the virtual IP address of ACOS_decrypt so that the real server connects to ACOS_decrypt over VLAN 15. Bind **FW1_Inspect** interface to TCP port 2323 so that ACOS_decrypt forwards decrypted SSH over VLAN 15 to the security device. All other UDP and TCP traffic is forwarded on VLAN 15 by using the wildcard ports **port 0 tcp** and **port 0 udp**.

```
ACOS_decrypt(config)# slb server FW1_Inspect 10.15.1.12

ACOS_decrypt(config-real server)# port 2323 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
```

```
ACOS_decrypt(config-real server-node port)# exit

ACOS_decrypt(config)# slb server FW1_Inspect 10.15.1.12
ACOS_decrypt(config-real server)# port 0 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit

ACOS_decrypt(config-real server)# port 0 udp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit
```

Step 4. Configuring the SSHi Service Groups

For an explanation of the procedure, refer to a similar procedure discussed in [Step 4. Configuring the SSLi Service Groups](#).

```
ACOS_decrypt(config)# slb service-group FW1_Inspect_SG tcp
ACOS_decrypt(config-slb svc group)# member FW1_Inspect 2323
ACOS_decrypt(config-slb svc group)# exit

ACOS_decrypt(config)# slb service-group ALL_TCP_SG tcp
ACOS_decrypt(config-slb svc group)# member FW1_Inspect 0
ACOS_decrypt(config-slb svc group)# exit

ACOS_decrypt(config)# slb service-group ALL_UDP_SG udp
ACOS_decrypt(config-slb svc group)# member FW1_Inspect 0
ACOS_decrypt(config-slb svc group)# exit
```

Step 5. Configuring the Virtual Server

For an explanation of the procedure, refer to a similar procedure discussed in [Step 5. Configuring the Virtual Server](#).

```
ACOS_decrypt(config)# access-list 100 permit ip any any vlan 10

ACOS_decrypt(config)# slb virtual-server Decrypt_VIP 0.0.0.0 acl 100

ACOS_decrypt(config-slb vsrver)# port 22 ssh
ACOS_decrypt(config-slb vsrver-vport)# no-dest-nat port-translation
ACOS_decrypt(config-slb vsrver-vport)# service-group FW1_Inspect_SG
ACOS_decrypt(config-slb vsrver-vport)# template client-ssh SSHInsight_
DecryptSide
```

```
ACOS_decrypt(config-slb vserver-vport)# exit

ACOS_decrypt(config-slb vserver)# port 0 tcp
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat
ACOS_decrypt(config-slb vserver-vport)# service-group ALL_TCP_SG
ACOS_decrypt(config-slb vserver-vport)# exit

ACOS_decrypt(config-slb vserver)# port 0 udp
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat
ACOS_decrypt(config-slb vserver-vport)# service-group ALL_UDP_SG
ACOS_decrypt(config-slb vserver-vport)# exit

ACOS_decrypt(config-slb vserver)# port 0 others
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat
ACOS_decrypt(config-slb vserver-vport)# service-group ALL_UDP_SG
ACOS_decrypt(config-slb vserver-vport)# exit
ACOS_decrypt(config-slb vserver)# exit
```

ACOS_encrypt Configuration

Perform the following steps for the ACOS_encrypt partition:

Step 1. Configuring the Network VLANs

```
ACOS(config)# hostname ACOS_encrypt
ACOS_encrypt(config)# vlan 20
ACOS_encrypt(config-vlan:20)# tagged ethernet 2
ACOS_encrypt(config-vlan:20)#router-interface ve 20
ACOS_encrypt(config-vlan:20)# exit

ACOS_encrypt(config)# vlan 15
ACOS_encrypt(config-vlan:15)# tagged ethernet 1
ACOS_encrypt(config-vlan:15)#router-interface ve 15
ACOS_encrypt(config-vlan:15)# exit
```

Step 2. Configuring the Network IP Addresses

```
ACOS_encrypt(config)# interface ve 20
ACOS_encrypt(config-if:ve20)# ip address 20.1.1.2 /24
ACOS_encrypt(config-if:ve20)# exit
```

```
ACOS_encrypt(config)# interface ve 15
ACOS_encrypt(config-if:ve15)# ip address 10.15.1.12 /24
ACOS_encrypt(config-if:ve15)# ip allow-promiscuous-vip
ACOS_encrypt(config-if:ve15)# exit
```

Step 3. Configuring the SSH Services

1. Create an SSH server template on ACOS_encrypt so that the VIP on ACOS_encrypt can operate as an SSL client and handshake with the EnterpriseABC server.

```
ACOS(config)# slb template server-ssh SSHInsight_DecryptSide
ACOS(config-server ssl)# forward-proxy-enable
ACOS(config-server ssl)# exit
```

2. Create the real server **Default_Gateway**. Bind the SLB ports of the intercepted SSH protocol (port 22) to **Default_Gateway**. ACOS_encrypt forwards the traffic on these ports over VLAN 20 to the default gateway at IP address 20.1.1.10. The default gateway has a route to the EnterpriseABC server.

```
ACOS_encrypt(config)# slb server Default_Gateway 20.1.1.10
ACOS_encrypt(config-real server)# port 22 tcp
ACOS_encrypt(config-real server-node port)# health-check-disable
ACOS_encrypt(config-real server-node port)# exit
```

3. All other UDP and TCP traffic is forwarded on VLAN 20 to the default gateway using the wildcard ports: **port 0 tcp** and **port 0 udp**.

```
ACOS_encrypt(config)# slb server Default_Gateway 20.1.1.10
ACOS_encrypt(config-real server)# port 0 tcp
ACOS_encrypt(config-real server-node port)# health-check-disable
ACOS_encrypt(config-real server-node port)# exit

ACOS_encrypt(config-real server)# port 0 udp
ACOS_encrypt(config-real server-node port)# health-check-disable
ACOS_encrypt(config-real server-node port)# exit
```

4. Create an SSH template for the SSH service protocol to be intercepted.

```
ACOS_encrypt(config)# slb template server-ssh SSHInsight_EncryptSide
ACOS_encrypt(config-ssh)# forward-proxy-enable
ACOS_encrypt(config-ssh)# exit
```

Step 4. Configuring the SSH Service Groups

1. Provide a path for intercepted SSH traffic by creating a service group called **DG_SSH_SG** and binding it to port 22 of the SLB real server.

```
ACOS_encrypt(config)# slb service-group DG_SSH_SG tcp
ACOS_encrypt(config-slb svc group)# member Default_Gateway 22
ACOS_encrypt(config-slb svc group)# exit

ACOS_encrypt(config)# slb service-group DG_TCP_SG tcp
ACOS_encrypt(config-slb svc group)# member Default_Gateway 0
ACOS_encrypt(config-slb svc group)# exit

ACOS_encrypt(config)# slb service-group DG_UDP_SG udp
ACOS_encrypt(config-slb svc group)# member Default_Gateway 0
ACOS_encrypt(config-slb svc group)# exit
```

2. Provide a path to the default gateway for all other traffic by creating two service groups called **DG_TCP_SG** and **DG_UDP_SG**.

```
ACOS_encrypt(config)# slb service-group DG_TCP_SG tcp
ACOS_encrypt(config-slb svc group)# member Default_Gateway 0
ACOS_encrypt(config-slb svc group)# exit

ACOS_encrypt(config)# slb service-group DG_UDP_SG udp
ACOS_encrypt(config-slb svc group)# member Default_Gateway 0
ACOS_encrypt(config-slb svc group)# exit
```

Step 5. Configuring the Virtual Server

For an explanation of the procedure, refer to a similar procedure discussed in [Step 5. Configuring the Virtual Server](#).

```
ACOS_encrypt(config)# access-list 101 permit ip any any vlan 15

ACOS_encrypt(config)# slb virtual-server Encrypt_VIP 0.0.0.0 acl 101

ACOS_encrypt(config-slb vsrver)# port 2323 tcp-proxy
ACOS_encrypt(config-slb vsrver-vport)# no-dest-nat port-translation
ACOS_encrypt(config-slb vsrver-vport)# service-group DG_SSH_SG
ACOS_encrypt(config-slb vsrver-vport)# template server-ssh SSHInsight_
DecryptSide
```



```
ACOS_encrypt(config-slb vservice-vport)# exit

ACOS_encrypt(config-slb vservice)# port 0 tcp
ACOS_encrypt(config-slb vservice-vport)# no-dest-nat
ACOS_encrypt(config-slb vservice-vport)# service-group DG_TCP_SG
ACOS_encrypt(config-slb vservice-vport)# exit

ACOS_encrypt(config-slb vservice)# port 0 udp
ACOS_encrypt(config-slb vservice-vport)# no-dest-nat
ACOS_encrypt(config-slb vservice-vport)# service-group DG_UDP_SG
ACOS_encrypt(config-slb vservice-vport)# exit

ACOS_encrypt(config-slb vservice)# port 0 others
ACOS_encrypt(config-slb vservice-vport)# no-dest-nat
ACOS_encrypt(config-slb vservice-vport)# service-group DG_UDP_SG
ACOS_encrypt(config-slb vservice-vport)# exit
ACOS_encrypt(config-slb vservice)# exit
```

Consolidated Configuration

Show Running Config ACOS_decrypt

```
!
access-list 100 permit ip any any vlan 10
!
vlan 10
    tagged ethernet 1
    router-interface ve 10
!
vlan 15
    tagged ethernet 1
    router-interface ve 15
!
interface ethernet 1
    enable
!
interface ve 10
    ip address 10.10.1.2 255.255.255.0
    ip allow-promiscuous-vip
!
```

```
interface ve 15
  ip address 10.15.1.2 255.255.255.0
!
slb server FW1_Inspect 10.15.1.12
  port 2323 tcp
    health-check-disable
  port 0 tcp
    health-check-disable
  port 0 udp
    health-check-disable
!
slb service-group ALL_TCP_SG tcp
  member FW1_Inspect 0
!
slb service-group ALL_UDP_SG udp
  member FW1_Inspect 0
!
slb service-group FW1_Inspect_SG tcp
  member FW1_Inspect 2323
!
slb template client-ssh SSHInsight_DecryptSide
  forward-proxy-hostkey RSA_key_1234
  forward-proxy-enable
!
slb virtual-server Decrypt_VIP 0.0.0.0 acl 100
  port 22 ssh
    service-group FW1_Inspect_SG
    template client-ssh SSHInsight_DecryptSide
    no-dest-nat port-translation
  port 0 tcp
    service-group ALL_TCP_SG
    no-dest-nat
  port 0 udp
    service-group ALL_UDP_SG
    no-dest-nat
  port 0 others
    service-group ALL_UDP_SG
    no-dest-nat
!
```

```
end
```

Show Running Config ACOS_encrypt

```
!  
access-list 101 permit ip any any vlan 15  
!  
vlan 20  
    tagged ethernet 1  
    router-interface ve 20  
!  
vlan 15  
    tagged ethernet 1  
    router-interface ve 15  
!  
interface ethernet 1  
    enable  
!  
interface ve 20  
    ip address 20.1.1.2 255.255.255.0  
!  
interface ve 15  
    ip address 10.15.1.12 255.255.255.0  
    ip allow-promiscuous-vip  
!  
slb server Default_Gateway 20.1.1.10  
    port 22 tcp  
        health-check-disable  
    port 0 tcp  
        health-check-disable  
    port 0 udp  
        health-check-disable  
!  
slb service-group DG_SSH_SG tcp  
    member Default_Gateway 22  
  
slb service-group DG_TCP_SG tcp  
    member Default_Gateway 0
```

```
slb service-group DG_UDP_SG udp
  member Default_Gateway 0
!
slb template server-ssh SSHInsight_EncryptSide
  forward-proxy-enable
!
slb virtual-server Outside_VIP 0.0.0.0 acl 101
  port 2323 tcp-proxy
    no-dest-nat port-translation
    service-group DG_SSH_SG
    template server-ssh SSHInsight_EncryptSide
  port 0 tcp
    no-dest-nat
    service-group DG_TCP_SG
  port 0 udp
    no-dest-nat
    service-group DG_UDP_SG
  port 0 others
    no-dest-nat
    service-group DG_UDP_SG
!
end
```

Certificate and Keys Management

This section describes managing SSL certificates, private keys, and Certificate Revocation Lists (CRLs). An ACOS device can offload SSL processing from servers or, for some types of traffic, can be used as an SSL proxy.

The following topics are covered:

SSL Certificate Management	198
CAs and CSRs Management	240
OCSP Certificate Management	281

SSL Certificate Management

Some types of client-server traffic need to be encrypted for security. For example, traffic for online shopping must be encrypted to secure sensitive account information from being stolen.

Commonly, clients and servers use SSL or TLS to secure traffic. For example, a client that is using a shopping application on a server will encrypt data before sending it to the server. The server will decrypt the client's data, then send an encrypted reply to the client. The client will decrypt the server reply, and so on. SSL is an older version of TLS.

ACOS device supports the following SSL and TLS versions:

- SSL v3.0
- SSL v3.0
- TLS v1.1
- TLS v1.2
- TLS v1.3

By default, the following RFC standards are supported in ACOS:

- [RFC 3268](#) - AES Cipher suites for TLS. For simplicity, elsewhere this document and other ACOS user documents use the term “SSL” to mean both SSL and TLS.
- [RFC 5746](#) - Renegotiation Indication Extension along with renegotiation_info TLS extension. It allows ACOS to securely renegotiate TLS connections with clients, using existing secure connections.
- [RFC 1421](#) - Privacy Enhanced Mail (PEM) format for certificate files and CRLs. It allows ACOS to process PEM format and RSA encryption.
- [RFC 7627](#) - Extended Master Secret (EMS) (and a TLS extension). It allows ACOS to define a TLS extension that contextually binds the master secret to a log of the full handshake that computes and prevents man-in-the-middle (MITM) attacks.

NOTE: When an SSL certificate expires or is near expiration, the ACOS device will automatically send a system log warning, rather than a system log notice. For information on enabling SNMP traps for SSL certificate events, refer to the *System Configuration and Administration Guide*.

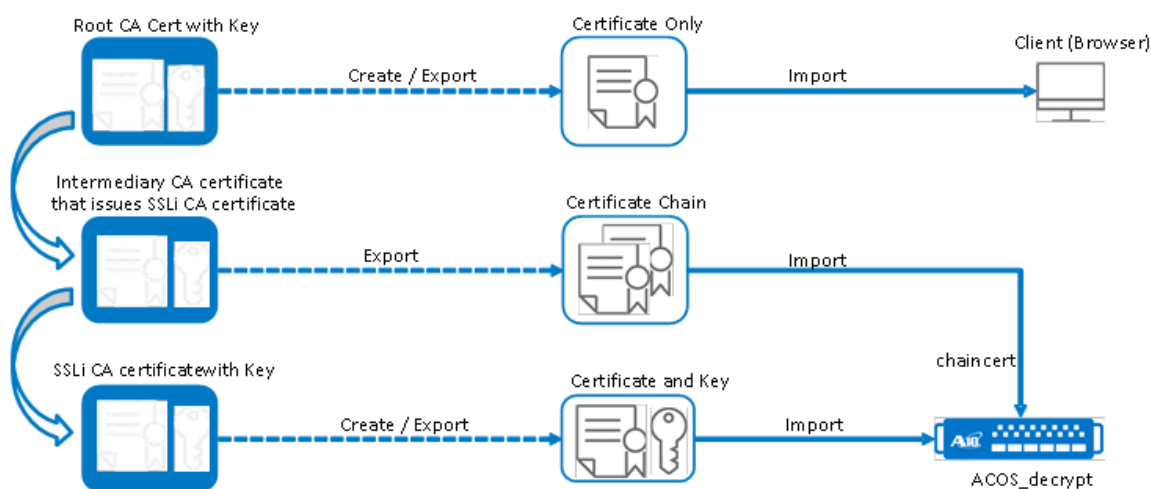
CA Certificate Chaining

SSLi requires a CA certificate and key pair to decrypt traffic between clients and any external SSL servers that are not controlled by the same organization. When an internal user from the client network initiates any SSL communication with an external server, the SSLi solution intercepts the server certificate from the original server, modifies the certificate and then re-signs it using the CA certificate. This proxy certificate is then sent to the internal user as a server certificate of the original server.

This CA certificate must be signed by the root CA. Otherwise, internal users see an `SSL untrusted root` error whenever they try to connect to an SSL-enabled website. Import the CA certificate and key pair to the `ACOS_decrypt`. This CA certificate must be trusted by the client web browsers. There are a number of third-party certificate distribution solutions available for this function. Microsoft Group Policy Manager is a recommended tool for Windows-based clients.

In the following example, the CA certificate for SSLi is signed by another trusted intermediate CA instead of a root CA. A CA certificate chain is required to complete the chain of trust. The CA certificate chain is created by concatenating the intermediate CA certificates from the one for SSLi up to the one signed by the root CA. In this example, the intermediate CA certificate is signed by the root CA. The certificate chain include two certificates and the root CA (ca.cert.pem).

Figure 19 : SSLi CA Certificate Chain



After the intermediate CA and certificate chain are ready, you can import both as a certificate type into the SSLi device. Since CSR is used, the private key (ssli-ca.key) is already on the SSLi device.

From the client's perspective, the SSL session is directly between the client and the outside SSL server. However, the SSL session is actually between the ACOS_decrypt device and the client.

The following is the workflow for the exchange of security certificates during the SSLi operation:

1. The client sends a request to set up an SSL session with the outside server.
2. Assuming that ACOS_decrypt has cached a proxied certificate for the outside server, it presents the certificate to the client.
3. If the client browser contains a copy of the proxied certificate, the client trusts ACOS_decrypt and allows the SSL session to be set up.
4. If ACOS_decrypt has not cached a proxied certificate for the outside server, then:
 - a. It opens an SSL session with the server and retrieves the server's public certificate.
 - b. It modifies and resigns with its imported private key to create the needed proxied certificate. The header information is specifically extracted from the server certificate.

- c. The issuer and the public key are changed as specified in the client-SSLi template.
- d. The modified certificate is then re-signed with the CA private key specified in the client-SSLi template.

The default CA bundle is used for remote certificate validation. The trusted CA certificates imported from browsers such as Mozilla do not require importing of any private keys.

Ensure that you have the latest root certificate bundle for remote certificate validation. The default_ca_bundle may not contain the latest certificates. It is highly recommended to update the default_ca_bundle periodically using either an automated or manual process. For the most current root certificates, refer [Mozilla Certs](#).

Additionally, the device administrators can automatically update the default CA bundle (a10_autoupdate_ca) from the GLM server using 'automatic-update' option. The CA bundle file is from [CA Extracts](#).

NOTE:	Before using the automatic-update feature, ensure that the device is registered with the Global License Manager (GLM).

For more information on:

- How to register the device with GLM, refer 'Activating your Appliance' section in *Global License Manager User Guide*.
- How to update the CA bundle, refer 'automatic-update ca-bundle' section in *Command Line Reference Guide*.

CA-Signed and Self-Signed Certificates

Typically, clients have a certificate store that includes certificates signed by the various root CAs. The certificate store may also have some non-CA certificates that can be validated by a root CA certificate, either directly or through a chain of certificates that end with a root certificate.

Each certificate is digitally “signed” to validate its authenticity. Certificates can be CA-signed or self-signed:

- **CA-signed** – A CA-signed certificate is a certificate that is created and signed by a recognized Certificate Authority (CA). To obtain a CA-signed certificate, an admin creates a key and a Certificate Signing Request (CSR), and sends the CSR to the CA. The CSR includes the key.

The CA then creates and signs a certificate. The admin installs the certificate on the ACOS device. When a client sends an HTTPS request, the ACOS device sends a copy of the certificate to the client, to verify the identity of the server (ACOS device).

To ensure that clients receive the required chain of certificates, you also can send clients a certificate chain in addition to the server certificate. (See [Certificate Chain](#).)

The example in [Typical SSL Handshake \(simplified\)](#) uses a CA-signed certificate.

- **Self-signed** – A self-signed certificate is a certificate that is created and signed by the ACOS device. A CA is not used to create or sign the certificate.

CA-signed certificates are considered to be more secure than self-signed certificates. Likewise, clients are more likely to be able to validate a CA-signed certificate than a self-signed certificate. If you configure the ACOS device to present a self-signed certificate to clients, the client's browser may display a certificate warning. This can be alarming or confusing to end users. Users can select the option to trust a self-signed certificate, in which case the warning will not re-appear.

CA Certificate Versus SSL Certificate

Although both terms, CA certificate and SSL certificate, refer to a certificates used in the SSL protocol, ACOS reserves the term SSL certificate for self-signed certificates that are used to create proxied certificates for SSL handshaking with clients in the SSLi, SSL Proxy or SSL offload applications. SSL certificates require a private key to be proxied

CA certificates are issued by publicly recognized certificate authorities. These certificates are used for other purposes.

Configuring Certificate Key Pair

- For client SSL template, use the following command:

```
certificate <cert-name> key <key-name> [pass-phrase <pass-phrase-str>]  
[chain-cert <chain-cert-name>]
```

- For server SSL template, use the following command:

```
certificate <cert-name> key <key-name> [pass-phrase <pass-phrase-str>]
```

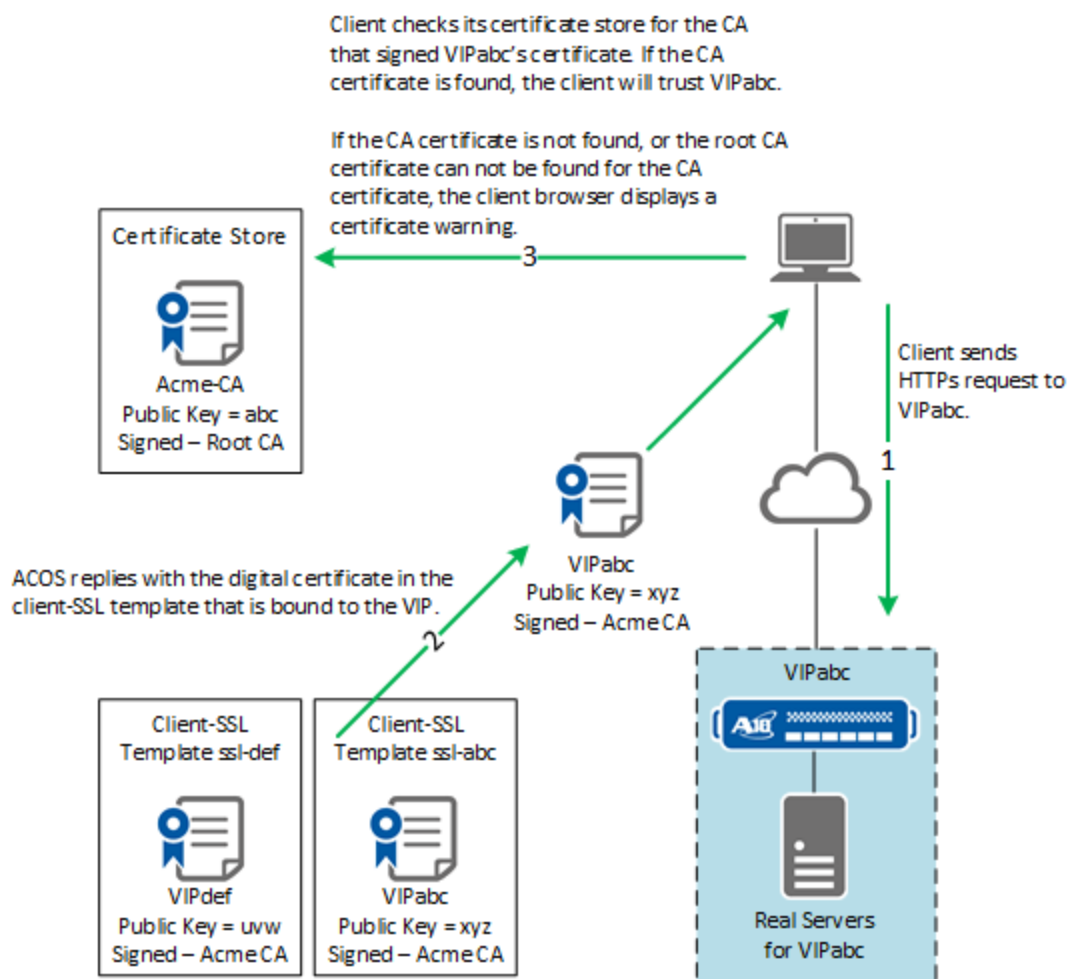
NOTE: When ACOS is upgraded, old certificate and key command configurations are not converted to new certificate command. New certificate commands will be lost if ACOS is downgraded. One client-SSL template can have two certificate and key pairs configured. Once one certificate /key pair configured, to update certificate or key, remove the old one and configure the new pair.

SSL Handshake WorkFlow

SSL works using certificates and keys. Typically, a client will begin a secure session by sending an HTTPS request to a VIP. The request begins an SSL handshake. The ACOS device will respond with a digital certificate, to provide verification of the content server's identity. From the client's perspective, this certificate comes from the server. Once the SSL handshake is complete, the client begins an encrypted client-server session with the ACOS device.

[Figure 20](#) shows a simplified example of an SSL handshake. In this example, the ACOS device is acting as an SSL proxy for backend servers.

Figure 20 : Typical SSL Handshake (simplified)



To begin, the client sends an HTTPS request. The request includes some encryption details such as the cipher suites supported by the client.

The ACOS device, on behalf of the server, checks for a client-SSL template bound to the VIP. If a client-SSL template is bound to the VIP, the ACOS device sends all the digital certificates contained in the template to the client.

The client browser checks its certificate store (sometimes called the certificate list) for a copy of the server certificate. If the client does not have a copy of the server certificate, the client will check for a certificate from the Certificate Authority (CA) that signed the server certificate.

Certificate Chain

Ultimately, a certificate must be validated by a root CA. Certificates from root CAs are the most trusted. They do not need to be signed by a higher (more trusted) CA.

If the CA that signed the certificate is a root CA, the client browser needs a copy of the root CA's certificate. If the CA that signed the server certificate is not a root CA, the client browser should have another certificate or a certificate chain that includes the CA that signed the CA's certificate.

A certificate chain contains the “chain” of signed certificates that leads from the CA to the signature authority that signed the certificate for the server. Typically, the certificate authority that signs the server certificate also will provide the certificate chain. [Figure 21](#) shows an example of a certificate chain containing three certificates:

Figure 21 : SSL Certificate Chain Example

```
-----BEGIN CERTIFICATE-----
ZS9naWYwITAfMacGBSsOAwIaBBRLa7kolgYMu9BSOJsprEsHiyEFGDAmFiRodHRw
Oi8vbG9nby52ZXJpc2lnbi5jb20vdnNsb2dvMS5naWYwDQYJKoZIhvcNAQEFBQAD
gYEAheIVEe8vArUOZxKkUIGjaYymzJA8Ty0uUPrikLpQ0IGezByVdbDUJ+HQLGp
2eruTPZpBNADaEfymstIPixrsuCRhyr3Ymsa2rgzwy9kSxEG83H7E7HxRnpDNZ8
l+uzpU/rk4j3bO/JVxPZMnwzMWriPSYgL1EKYcOSKyReACOSQ=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
ZS9naWYwITAfMacGBSsOAwIaBBRLa7kolgYMu9BSOJsprEsHiyEFGDAmFiRodHRw
Oi8vbG9nby52ZXJpc2lnbi5jb20vdnNsb2dvMS5naWYwDQYJKoZIhvcNAQEFBQAD
gYEAheIVEe8vArUOZxKkUIGjaYymzJA8Ty0uUPrikLpQ0IGezByVdbDUJ+HQLGp
2eruTPZpBNADaEfymstIPixrsuCRhyr3Ymsa2rgzwy9kSxEG83H7E7HxRnpDNZ8
l+uzpU/rk4j3bO/JVxPZMnwzMWriPSYgL1EKYcOSKyReACOSQ=
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
ZS9naWYwITAfMacGBSsOAwIaBBRLa7kolgYMu9BSOJsprEsHiyEFGDAmFiRodHRw
Oi8vbG9nby52ZXJpc2lnbi5jb20vdnNsb2dvMS5naWYwDQYJKoZIhvcNAQEFBQAD
gYEAheIVEe8vArUOZxKkUIGjaYymzJA8Ty0uUPrikLpQ0IGezByVdbDUJ+HQLGp
2eruTPZpBNADaEfymstIPixrsuCRhyr3Ymsa2rgzwy9kSxEG83H7E7HxRnpDNZ8
l+uzpU/rk4j3bO/JVxPZMnwzMWriPSYgL1EKYcOSKyReACOSQ=
-----END CERTIFICATE-----
```

The certificate chain file and the server certificate files are text files. Each certificate must begin with the “-----BEGIN CERTIFICATE-----” line and end with the “-----END CERTIFICATE-----” line.

The certificate at the top of the certificate chain file is the root CA's certificate. The next certificate is an intermediary certificate signed by the root CA. The next certificate is signed by the intermediate signature authority that was signed the root CA.

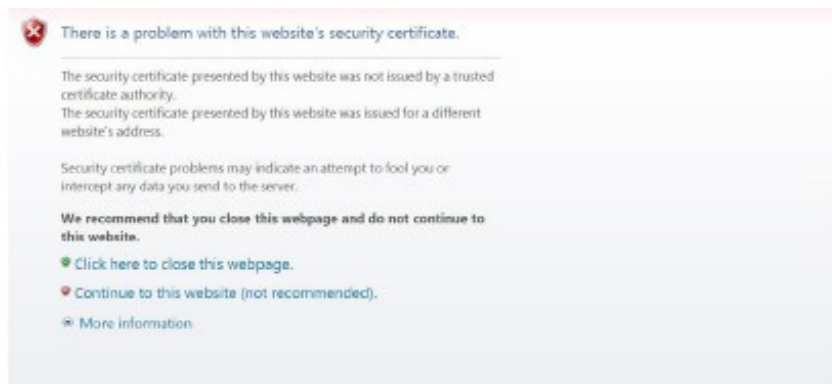
A certificate chain in an SSL template must begin at the top with the root CA's certificate, followed in order by the intermediary certificates. If the certificate authority that signs the server certificate does not provide the certificate chain in a single file, you can use a text editor to chain the certificates together in a single file as shown in [Figure 21](#).

Certificate Warning from Client Browser

After the client browser validates the server certificate, the client accepts the certificate and begins an encrypted session with the ACOS device.

If the client can not validate the server certificate or the certificate is out of date, the client's browser may display a certificate warning. [Figure 22](#) shows an example of a certificate warning displayed by Internet Explorer.

Figure 22 : Example of Certificate Warning

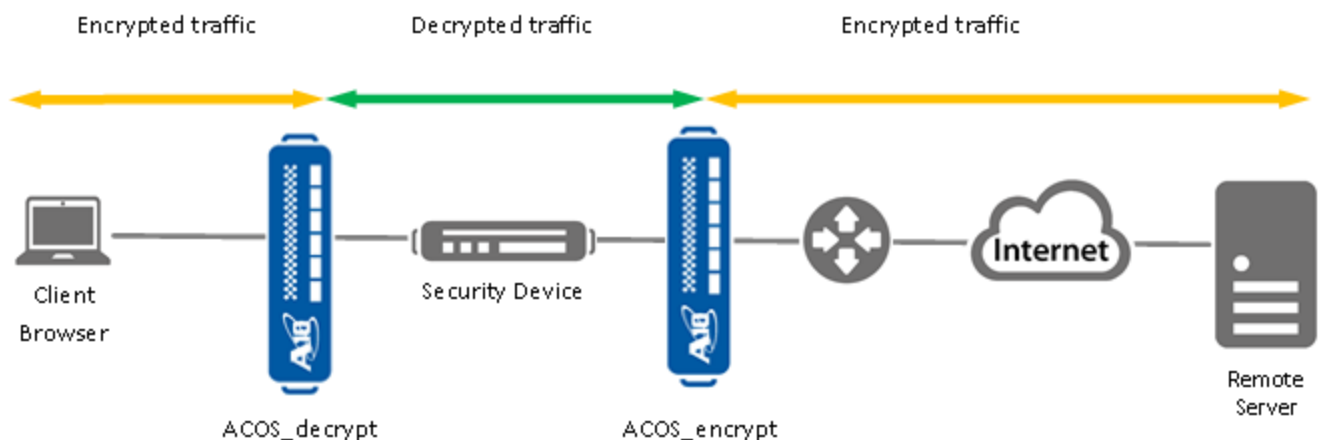


NOTE: It is normal for the ACOS device to display a certificate warning when an admin accesses the ACOS management GUI. Certificates used for SLB are not used by the management GUI.

Websites Workflow

The flow of traffic from the client to the gateway by using an SSLi solution requires a security certificate to be configured for the SSLi solution. In this section, the sequence of events, including the security certificate exchange process, is explained for processing the SSL traffic in a typical deployment. The process is explained for both new and revisited websites.

Figure 23 : SSLi Flow of Traffic



In any typical SSLi deployment such as the one displayed in this section, the flow of traffic from the client network to the outside network or server network is processed by the SSLi solution as follows for new websites:

1. The client establishes an SSL connection with the remote server and receives a security certificate from the remote server.
2. In ACOS_decrypt, the header information is extracted from the server certificate.
3. In the client SSLi template defined for ACOS_decrypt, a new security certificate is generated by using the CA certificate specified in the client SSLi template. This reconstructed server-hello message is sent to the client instead of the original encrypted hello message.
4. ACOS_decrypt is now able to intercept traffic, decrypt it and send the clear-text to the security device.
5. A new SSL session is initiated with the remote server by ACOS_encrypt.
6. Clear text data is passed from the security device to ACOS_encrypt. ACOS_encrypt re-encrypts the data and sends it to the remote server.

7. The server response is intercepted by ACOS_encrypt which decrypts it and passes it to the security device.
8. The security device processes the clear text data and passes it to ACOS_decrypt. ACOS_decrypt re-encrypts the data and sends it to the client.

Now that ACOS_decrypt has a cached certificate and if the client were to make another request for connection to the remote server, the flow of traffic from the client network to the outside network or server network is processed by the SSLi solution as follows:

1. The client establishes an SSL connection with the remote server and receives the security certificate from the remote server.
2. ACOS_decrypt sends the client the cached certificate of the website.
3. ACOS_decrypt is now able to intercept traffic, decrypt it and send the clear-text to the security device.
4. A new SSL session is initiated with the remote server by ACOS_encrypt.
5. Clear text data is passed from the security device to ACOS_encrypt. ACOS_encrypt re-encrypts the data and sends it to the remote server.
6. The server response is intercepted by ACOS_encrypt which decrypts it and passes it to the security device.
7. The security device processes the clear text data and passes it to ACOS_decrypt. ACOS_decrypt re-encrypts the data and sends it to the client.

Certificates in SSL Templates

You can install more than one key-certificate pair on the ACOS device. The ACOS device selects the certificate(s) to send a client or server based on the SSL template bound to the VIP. You can bind the following types of SSL templates to VIPs:

- Client-SSL template – Contains keys and certificates for SSL-encrypted traffic between clients and the ACOS device. A client-SSL template can also contain a certificate chain.
- Server-SSL template – Contains CA certificates for SSL-encrypted traffic between servers and ACOS device.

NOTE: If you replace a certificate and key in a client-SSL or server-SSL template, you must unbind the template from the virtual ports that use it, then rebind the template to the virtual ports, to place the change into effect.

One Client SSL template can have two certificate-key pairs configured. Thus, once one certificate-key pair is configured, certificate or key. To update certificate or key, remove the old one and then add new one.

Client-SSL Template Configuration and Usage Guidelines

Use client-SSL templates for deployments in which traffic between clients and the ACOS device will be SSL-encrypted. Client-SSL templates have the following options.

For the simple deployment example in [Typical SSL Handshake \(simplified\)](#), only the first option (Certificate) needs to be configured. You may also need to configure the Certificate chain option. A client-SSL template can contain up to 128 certificates or certificate chains.

For more information on all the available options, see *Command Line Interface Reference*.

- **Certificate** – Specifies the server certificate that the VIP will send to a client when configured for SSL proxy, SSL offload, or SSLi operation. The client uses this certificate to validate the server's identity. The certificate can be generated on the ACOS device (self-signed) or can be signed by another entity and imported onto the ACOS device.

Multiple certificates can be associated to one client-SSL template such as RSA or ECDSA or SM2 certificates. Each type can have one certificate associated to one client-SSL template.

Use the `show pki cert` command to show the list of certificates and private keys stored on the ACOS device. Additionally, you can also update the CA bundle (a10_ autoupdate_ca) installed on the GLM server.

- **CA-Certificate** – Specifies a CA certificate that the ACOS device can use to authenticate the identity of a client the requesting to connect to the ACOS device. If CA certificates are required, they must be imported onto the ACOS device.

Multiple CA-certificate can be associated with the client-SSL template.

The ACOS device is configured at the factory with the CA certificate bundle file "default_ca_bundle," including common trusted certificate authorities. Use the `show pki ca-cert` command to view the list of ca-certificates.

- Certificate Revocation List (CRL) – Specifies a list of client certificates that have been revoked by the CAs that signed them. This option is applicable only if the ACOS device will be required to validate the identities of clients.

The CRL should be signed by the same issuer as the CA certificate. Otherwise, the client and ACOS device will not be able to establish a connection.

- Cipher Template – Specifies the cipher suites supported by the ACOS device. When the client sends its connection request, it also sends a list of the cipher suites it can support. The ACOS device selects the strongest cipher suite supported by the client that is also enabled in the template, and uses that cipher suite for traffic with the client. For a list of supported ciphers, refer to the slb template cipher command in the Command Line Interface Reference
- Client connection-request response – Specifies the ACOS response to connection requests from clients. This option is applicable only if the ACOS device will be required to validate the identities of clients. The response can be one of the following:
 - ignore (default) – The ACOS device does not request the client to send its certificate.
 - request – The ACOS device requests the client to send its certificate. With this action, the SSL handshake proceeds even if either of the following occurs:
 - The client sends a NULL certificate (one with zero length).
 - The certificate is invalid, causing client verification to fail.

Use this option if you want to the request to trigger an aFlex policy for further processing.

 - require – The ACOS device requires the client certificate. This action requests the client to send its certificate. However, the SSL handshake does not proceed (it fails) if the client sends a NULL certificate or the certificate is invalid.

- Close-notify – Specifies whether the ACOS device sends a close_notify message when an SSL transaction ends, before sending a FIN. This behavior is required by certain types of applications, including PHP cgi.
- Early Data - Specifies the early data (0-RTT) for SSL version TLSv1.3. This allows the TLS client to send encrypted data in the same packet as the Client Hello during the handshake for resumed sessions. Optionally, enable or disable anti-replay for early data. Additionally, you must configure `session-cache-size` to do PSK resumption.
- EC-name - Specifies the Elliptic curve cryptographic (ECC) algorithm name for key exchange during SSL handshake.
- DH parameters - Specifies the Diffie-Hellman parameters.
- SSL version – Highest (most secure) version of SSL/TLS to use. The ACOS device supports the following SSL/TLS versions:
 - SSL v3.0
 - TLS v1.0 (the default)
 - TLS v1.1
 - TLS v1.2
 - TLS v1.3
- SSLv2 bypass – Redirects clients who request SSLv2 sessions to the specified service group.
- Session cache size – Specifies the maximum number of cached sessions for SSL session ID reuse.
- Session cache timeout – Sets the maximum number of seconds a cache entry can remain unused before being removed from the cache. Cache entries age according to the ticket age time. The age time is not reset when a cache entry is used.
- Session ticket - Specifies whether the stateless SSL session ticketing feature is enabled or disabled.
- Session ticket lifetime – Sets the lifetime for stateless SSL session ticketing. After a client's SSL ticket expires, they must complete an SSL handshake in order to set up the next secure session with ACOS.
- SSL False Start – Specifies whether SSL False Start is enabled. SSL False Start is an

SSL modification used by the Google Chrome browser for web optimization.

NOTE: The following ciphers are not supported with SSL False Start: SSL3_RSA_DES_64_CBC_SHA, SSL3_RSA_RC4_40_MD5, TLS1_RSA_EXPORT1024_RC4_56_MD5. If no other ciphers but these are enabled in the client-SSL template, SSL False Start handshakes will fail.

- Cipher Template – Configure the cipher template containing a set of ciphers to use with clients. By default, the client-SSL template's own set of ciphers is used. (See [Cipher Template Configuration and Usage Guidelines](#).)
- Forward proxy options – Options that are used for SSL Insight.
- Authentication username attribute – Specifies the field to check in SSL certificates from clients, to find the client name.

Server-SSL Template Configuration and Usage Guidelines

A server-SSL template is needed only if traffic between the ACOS device and real servers will be encrypted using SSL. In this case, the ACOS device will be required to validate the identities of the servers.

- CA-Certificate – Specifies a CA certificate that the ACOS device can use to authenticate the identity of a server the ACOS device is connecting to. If CA certificates are required, they must be imported onto the ACOS device. Multiple CA-certificate can be associated with the server-SSL template. See [Multiple CA Certificate Support in Server-SSL Templates](#).

The ACOS device is configured at the factory with the CA certificate bundle file "default_ca_bundle," including common trusted certificate authorities. Use the `show pki ca-cert` command to view the list of ca-certificates.

- Certificate – Specifies a client certificate that the ACOS device will send to a server when requested for client authentication. In SSL proxy and SSL Insight, when a server requests a client's digital certificate, the ACOS device responds on behalf of the client. Following successful authentication, the server and ACOS device communicates over an SSL-encrypted session.
 - In SSL Proxy, the client and ACOS device communicate over a non-encrypted session. From the server's perspective, the server has an encrypted session with the client.

- In SSL Insight, the client and ACOS device communicate over an encrypted session. From the client's and the server's perspective, the SSL session is fully encrypted.
- Cipher list – Specifies the cipher suites supported by the ACOS device. When the server sends its connection request, it also sends a list of the cipher suites it can support. The ACOS device selects the strongest cipher suite supported by the server that is also enabled in the template and uses that cipher suite for traffic with the server. The same cipher suites supported in client-SSL templates are supported in server-SSL templates, for CA certificates. Support for all of them is enabled by default.

NOTE: For client certificates, the key length for SSL3_RSA_DES_40_CBC_SHA and SSL3_RSA_RC4_40_MD5 must be 512 bits or less. The TLS1_RSA_EXPORT1024_RC4_56_MD5 and TLS1_RSA_EXPORT1024_RC4_56_SHA ciphers are not supported.

- Close notification – Specifies whether the ACOS device sends a close_notify message when an SSL transaction ends, before sending a FIN. This behavior is required by certain types of applications, including PHP cgi.

The close notification option may not work if connection reuse is also configured on the same virtual port. In this case, when the server sends a FIN to the ACOS device, the ACOS device will not send a FIN followed by a close notification. Instead, the ACOS device will send a RST.

- Early Data - Specifies the early data (0-RTT) is enabled for SSL version TLSv1.3. This allows the server to respond immediately by including the requested data in the Server Hello/Finished message. Additionally, you must configure either Session cache size or Session Ticket Enable to do PSK resumption.
- EC-name - Specifies the Elliptic curve cryptographic (ECC) algorithm name for key exchange during SSL handshake.
- DH parameters - Specifies the Diffie-Hellman parameters.
- SSL version – Highest (most secure) version of SSL/TLS to use. The ACOS device supports the following SSL/TLS versions:
 - SSL v3.0
 - TLS v1.0 (the default)

- TLS v1.1
- TLS v1.2
- TLS v1.3
- Session cache size – Specifies the maximum number of cached sessions for SSL session ID reuse.
- Session cache timeout – Sets the maximum number of seconds a cache entry can remain unused before being removed from the cache. Cache entries age according to the ticket age time. The age time is not reset when a cache entry is used.
- Session ticket enable – Sets the lifetime for stateless SSL session ticketing. After an SSL ticket expires, the SSL handshake must be performed again in order to set up the next secure session with ACOS.
- Cipher template – Name of a cipher template containing a set of ciphers to use with servers. By default, the server-SSL template's own set of ciphers is used. (See [Cipher Template Configuration and Usage Guidelines](#).)
- Forward proxy – Enables support for capabilities required for SSL Intercept.

Cipher Template Configuration and Usage Guidelines

A cipher template contains a [list of ciphers](#). A client or server who connects to a virtual port that uses the cipher template can use only the ciphers that are listed in the template.

Optionally, you can assign a priority value to each cipher in the template. In this case, the ACOS device tries to use the ciphers based on priority. If the client supports the cipher that has the highest priority, that cipher is used. If the client does not support the highest-priority cipher, the ACOS device attempts to use the cipher that has the second-highest priority, and so on.

Cipher priority can be 1-100. The highest priority (most favored) is 100. By default, each cipher has priority 1. More than one cipher can have the same priority. In this case, the strongest (most secure) cipher is used.

NOTE:

- An SSL cipher template takes effect only when applied to a client-SSL template or server-SSL template.
- When you apply (bind) a cipher template to a client-SSL or server-SSL template, the settings in the cipher template override any cipher settings in that client-SSL or server-SSL template.
- Priority values are supported only for client-SSL templates. If a cipher template is used by a server-SSL template, the priority values in the cipher template are ignored.

Certificate Fetching and Forging

In earlier SSLi deployments for new connections, when a server certificate fetch request was sent to a server, the incoming new SSLi connection requests to the same server were either bypassed or reset (based on configuration) till the time the server certificate was forged and ready.

However, this behavior may cause a security breach especially during initial connections when a cache certificate expired and all subsequent connections were either reset or bypassed till a new forged certificate was ready.

As a solution to this issue, there is a new configuration option available in the client-SSL template where you are able to buffer all new connections to a server till the time the forged certificate is ready. In case of an SSLi deployment with OSCP and CRL implemented, the new connections are buffered till a verification result response is received from the server.

NOTE:

The default option for this SSLi configuration is to bypass all new connections. Hence, in order to buffer the new connections from a server, the SSLi connection buffer option must be enabled either through the CLI or GUI, refer to the next sections.

For the certificate not ready option, the following is the output of the help command.

```
ACOS_decrypt(config-client ssl)#forward-proxy-cert-not-ready-action ?
bypass          bypass the connection(default)
reset           reset the connection
```

```
intercept      wait for cert and then inspect the connection.
```

NOTE: "intercept" is not supported under proxy-chaining configuration with both TP and EP.

CLI Configuration

To enable SSLi connection buffering in CLI, perform the following steps:

1. Configure the client SSL template called `SSLInsight_DecryptSide` by running the following commands:

```
ACOS_decrypt(config)# slb template client-ssl SSLInsight_DecryptSide
ACOS_decrypt(config-client ssl)# forward-proxy-ca-certificate
Cert123.pem key key123
ACOS_decrypt(config-client ssl)# forward-proxy-enable
```

2. Enable the option for intercept for the certificate not ready stage.

```
ACOS_decrypt(config-client ssl)# forward-proxy-cert-not-ready-action
intercept
```

3. Save the configuration.

GUI Configuration

For SSLi, perform the following steps:

1. Navigate to **Security > SSLi > Templates > +Create**. The **Create Client SSL** template page is displayed.
2. Enter the name of the template.
3. Select **Forward Proxy Enable**.
4. Under **SSLi Forward Proxy**, select the CA cert and Key.
5. Under **Advanced**, select **Intercept for Forward Proxy Cert Not Ready Action**.
6. Click **Create** to create the template.

For ADC, perform the following steps:

1. Navigate to **ADC >> Templates >> SSL >> Create >> Client SSL**.

The **Create Client SSL** template page is displayed.

2. Enter the name of the template.
3. Select **Intercept** for **Forward Proxy Cert Not Ready Action**.
4. Continue with the other fields to create the template.

Certificate Pinning Candidate List

Certificate Pinning is a method used by certain apps to secure traffic and defend against man-in-the-middle (MITM) attacks. In this case, an app stores or pins a copy of the original certificate within its code, and if it sees a modified version of that certificate, the app rejects it and issues RST.

As a solution to support such apps, SSLi provides a certificate pinning candidate list feature. This feature maintains a list of known domain names that use certificate pinning on their client apps.

To view the certificate pinning candidate list, use the following show or clear command.

```
ACOS(config)# show slb ssl-cert-pinning-candidate-list
ACOS(config)# clear slb ssl-cert-pinning-candidate-list {server-name} <1-255 characters>
```

Show Output:

SNI	Counter	TTL

youtube.com	10	1440
gmail.com	6	1440
google.com	5	1440
yahoo.com	3	1440
api.snapcraft.io	1	1430

Configure Bypass Rule for Local Cert Pinning List

This topic describes the configuration of SSLi bypass decision-making for local cert pinning list.

The SSLi bypass decision-making process is described below:

1. The bypass filter is invoked when Thunder receives a request, and the client SSL bypass/exception-bypass rule is configured.
2. Once the local cert pinning list bypass rule is set, the bypass filter will first look at this list.
3. If the SNI string is in the local cert pinning list, and if the connection failure count exceeds the threshold defined in the rule, the decision process will be broken. However, the processes will continue to bypass.
4. If not, it will check the other bypass and exception rules in the correct order.
5. If none of the rules are met, the intercept procedure will be initiated.

CLI Configuration

- Set the TTL value for the local cert pinning list, if required.

The value can be between 1 to 1008 (multiple of 10 minutes). By default, the value is 1440 minutes (a day), and the maximum value is 10080 minutes (a week).

The TTL will be reduced every 10 minutes, and the SNI entry will be deleted from the local cert pinning list when it reaches 0. The related TTL is reset to the default or set value if the same SNI is received again.

```
ACOS(config)# slb common
ACOS(config-common)# cert-pinning
ACOS(config-common-cert-pinning)# ttl 360
```

- Configure the forward-proxy bypass method under SLB client-SSL template configuration mode.

```
ACOS(config)# slb template client-ssl SSL_Client
ACOS(config-client ssl)# forward-proxy-bypass local-cert-pin-list conn-
fail-count 100
```

Show Command

- To view the local cert pinning list, use the following show command.

```
ACOS(config)# show slb ssl-cert-pinning-candidate-list local-list
bypassed alphabet-order
Local Generated Cert-Pinning Candidate List:
```

SNI	Conn failures	Bypass count	TTL

-			
api.snapcraft.io	501	60	60
cdn.fwupd.org	100	50	1100
dashboard.snapcraft.io	1200	56	930
extensions.gnome.org	100	73	1400
odrs.gnome.org	765	12	1010

- To clear the statistics, use the following clear command:

```
ACOS(config)# clear slb ssl-cert-pinning-candidate-list {SNI {name} |
all | central-list | local-list | stats-only}
```

GUI Configuration - TBD

- To configure cert pinning with TTL options, navigate to **ADC > SSL Management > Certificate Pinning List**.
- To configure forward-proxy-bypass options, navigate to **ADC > SSL Templates > Client-SSL Template**.
- To view local cert pinning list counters, navigate to **ADC > SSL Management > Certificate Pinning List**.

Central Cert Pinning List

The central cert pinning list is a global domain names list received and compiled from all the Thunder devices. A10 utilizes 'Pinned Certificates List Service' (also known as Central List Service) independent service to compile this list, upload it to the Global License Manager (GLM), and then redeploy to Thunder devices.

You can choose to provide your certificate pinning list to A10 by configuring the 'candidate-list-feedback-opt-in' option under SLB common configuration mode.

The central list service performs the following process:

1. Gathers all the cert pinning data from all the Thunder devices who enables this feature. The data contains pairs of SNI and value, calculated by the sum of connection failure count and bypassed count.

2. Sorts the SNIs in descending order and adds the top 100 entries in the central cert pinning list.
3. Compiles the central cert pinning list in the central list service and daily uploads to the GLM server if there is any difference.
4. Applies the central cert pinning list to every Thunder partition once the 'automatic-update' command is executed/configured.

NOTE: Before using the 'automatic-update' option, ensure that the Thunder device is registered with the GLM server. For more information on how to register the device, refer 'Activating your Appliance' section in the *Global License Manager User Guide*.

Download or Update Central Cert Pinning List

This topic describes how to download or update the central cert pinning list.

When updating the central cert pinning list, consider the following points:

- If the SNI entry in the central cert pinning list is already available in the local cert pinning list, it would be moved to the central list directly with the accumulated bypassed count.
- If the SNI entry of the central cert pinning list is removed in the latest version, the SNI entry will be moved to the local cert pinning list with the accumulated bypassed count if the maximum number of local lists is not reached.

CLI Configuration

- To immediately update the central cert pinning list to the latest version from the GLM server, use the following command:

```
ACOS(config)#automatic-update check-now central-cert-pin-list
```

- To immediately revert to the previous version of the central cert pinning list, use the following command:

```
ACOS(config)#automatic-update revert central-cert-pin-list
```

- To configure the schedule for updating the central cert pinning list periodically, use the following command:

```
ACOS(config)# automatic-update central-cert-pin-list schedule daily
12:30
```

Show Command

To view the central cert pinning list version/status, use the following show command:

```
ACOS(config)# show automatic-update central-cert-pin-list
```

Feature name	Version	Schedule	Time	Last Updated	Next
Check					
app-fw	1.360.0-23	N/A	00:00	2020-05-22	N/A
central-cert-pin-list	20201216-1	Daily	00:00	2020-12-15	
2020-12-16					

Configure Bypass Rule for Central Cert Pinning List

This topic describes the configuration of SSLi bypass decision-making for central cert pinning list.

The SSLi bypass decision-making process for the central cert pinning list is the same as [local cert pinning list](#) . However, the bypass filter only compares the SNI.

NOTE: If both the bypass rules of central and local cert pinning lists are set in the client-SSL template, then the central cert pinning list configuration takes precedence.

CLI Configuration

- To configure the forward-proxy bypass method under SLB client-SSL template configuration mode.

```
ACOS(config)#slb template client-ssl SSL_Client
ACOS(config-client ssl)#forward-proxy-bypass central-cert-pin-list
```

Show Command

- To view the central cert pinning list statistics, use the following show command:

```
ACOS(config)# show slb ssl-cert-pinning-candidate-list central-list
bypassed alphabet-order
```

SNI	Bypass count
api.snapcraft.io	113
cdn.fwupd.org	100
dashboard.snapcraft.io	501
extensions.gnome.org	96
odrs.gnome.org	300

- To clear the statistics, use the following clear command:

```
ACOS(config)# clear slb ssl-cert-pinning-candidate-list {SNI {name} |
all | central-list | local-list | stats-only}
```

GUI Configuration

- To configure forward-proxy-bypass options, go to **ADC > SSL Templates > Client-SSL Template**.
- To view certificate pinning candidate list counters, go to **ADC > SSL Management > Certificate Pinning List**.

Upload Local Cert Pinning list to Central List Service

This topic describes how to enable and upload the local cert pinning list to the central list service.

1. Enable the 'candidate-list-feedback-opt-in' option under SLB common configuration mode.
2. Schedule the time to upload the local cert pinning list data to the central list service.
3. Thunder will collect the local cert pinning list entries from every partition on the scheduled time and upload them to the central list service.
4. Central list service will sort the entries collected from devices by the sum of connection failure count and non-zero bypassed count in descending order and add the top 100 of them to the central cert pinning list.

CLI Configuration

To configure this feature, use the following command:

```

ACOS(config)# slb common
ACOS(config-common)# cert-pinning
ACOS(config-common-cert-pinning)# candidate-list-feedback-opt-in
ACOS(config-common-cert-pinning-candidate...)# schedule weekly monday
12:00
ACOS(config-common-cert-pinning-candidate...)# use-mgmt-port
ACOS(config-common-cert-pinning-candidate...)# exit

```

TLS Server Name Indication (SNI) Support

The ACOS device supports the Server Name Indication (SNI) extension for Transport Layer Security (TLS). The SNI extension enables the servers that manage content for multiple domains at the same IP address to use a separate server certificate for each domain. For example, web hosting services. In addition, the device supports both the Static and Dynamic SNI extensions.

NOTE: This feature is supported in both the shared partitions and L3V private partitions.

The following topics are covered:

Prerequisite	223
SNI Extension Support	224
SNI Support on vThunder	230
Configuring TLS SNI	231
Configuring Dynamic SNI Transmission for Inbound SSLi	232

Prerequisite

The client-SSL template must contain one certificate and private key pair that is not mapped to a domain. The unmapped certificate and key are the default certificate and key for the template. The ACOS device uses the default template for negotiating the SSL session with the client.

If the client includes the SNI extension in its hello message, the ACOS device uses the certificate that is mapped to the domain requested by the client. Otherwise, the ACOS device uses the default certificate.

SNI Extension Support

This section describes available SNI extension support methods: Static and Dynamic SNI Extension. When an SNI extension matches multiple entities, the selection is based on the following precedence:

- SNI extension matches static mapping configured with `server-name` and `server-name-regex` command.
- SNI extension matches dynamic mapping configured with `server-name-auto-map` command.

When an SNI extension does not match any of these entities or the client-hello does not contain an SNI extension, the default cert-key pair is used.

Static SNI Extension Support

You can configure up to 1024 certificate-to-domain mappings in a client-SSL template. Each mapping is configured using the `server-name` or the `server-name-regex` command at the configuration level for the client-SSL template.

Dynamic SNI Extension Support

When dynamic SNI extension support is enabled, a certificate-to-domain mapping is created when a certificate and key whose file names include the domain name specified by the client “hello” field of an inbound packet. The number of extensions that can be dynamically support on each virtual port is limited only by hardware restrictions.

SNI extensions use the default certificate and key when a “hello” field contains a domain name for which the device does not contains certificate and key with matching file name.

Dynamic SNI extension support is enabled by using the `server-name-auto-map` command.

SNI Bypass

When the `server-name`, `server-name-regex`, and `server-name-auto-map` is configured under client-SSL template then you can bypass the SSL traffic in the following scenarios:

- Missing cert/key, i.e., client SNI does not match - `server-name-bypass missing-cert`
- Client SNI matches one of the configured server-name, but the certificate is expired - `server-name-bypass expired-cert`
- Client SNI matches the configured SNI bypass AC type class-list - `server-name-bypass class-list sni_bypass`

CLI Configuration for Shared Partition (Inside)

```
class-list sni_bypass ac
contains 135_1
contains 132_1
!
ip access-list sni_bypass
permit ip 172.16.48.131 0.0.0.0 192.168.90.132 0.0.0.0
!
ip dns primary 192.168.90.136
!
partition adc id 1 application-type adc
!
timezone UTC
!
visibility
monitor traffic service
!
glm use-mgmt-port
glm enable-requests
!
interface management
ip address 192.168.90.48 255.255.255.0
ip default-gateway 192.168.90.254
!
interface ethernet 1
enable
!
interface ethernet 2
enable
ip address 192.168.91.48 255.255.255.0
```

```
!  
interface ethernet 4  
enable  
ip address 10.10.48.133 255.255.255.0  
!  
interface ethernet 7  
enable  
ip address 172.16.48.133 255.255.255.0  
ip allow-promiscuous-vip  
!  
ip nat pool inside_2 10.10.48.140 10.10.48.140 netmask /32  
!  
ip route 0.0.0.0 /0 10.10.48.143  
!  
ip route 168.95.1.1 /32 192.168.91.254  
!  
ip route 192.168.90.0 /24 192.168.90.254  
!  
ip route 192.168.98.0 /24 192.168.91.254  
!  
ip route 192.168.99.0 /24 192.168.91.254  
!  
slb template cipher all  
TLS1_ECDHE_ECDSA_AES_128_GCM_SHA256  
TLS1_ECDHE_ECDSA_AES_128_SHA  
TLS1_ECDHE_ECDSA_AES_128_SHA256  
TLS1_ECDHE_ECDSA_AES_256_GCM_SHA384  
TLS1_ECDHE_ECDSA_AES_256_SHA  
TLS1_ECDHE_RSA_AES_128_GCM_SHA256  
TLS1_ECDHE_RSA_AES_128_SHA  
TLS1_ECDHE_RSA_AES_128_SHA256  
TLS1_ECDHE_RSA_AES_256_GCM_SHA384  
TLS1_ECDHE_RSA_AES_256_SHA  
TLS1_ECDHE_RSA_AES_256_SHA384  
TLS1_ECDHE_ECDSA_AES_256_SHA384  
TLS1_ECDHE_RSA_CHACHA20_POLY1305_SHA256 priority 100  
TLS1_ECDHE_ECDSA_CHACHA20_POLY1305_SHA256  
!  
slb server ad 192.168.90.136
```

```
port 443 tcp
!
slb service-group adsg tcp
member ad 443
!
slb service-group sgtcp tcp
member apache1 0
!
slb template client-ssl cliss1
certificate acos_a10-tplab_com.pfx key acos_a10-tplab_com.pfx
server-name 172-16-48-135.a10-tplab.com cert jinling key jinling
server-name 192-168-90-132.a10-tplab.com cert jinling key jinling
server-name-regex 135_2 cert expiring key expiring
server-name-regex 132_2 cert expiring key expiring
server-name-bypass missing-cert
server-name-bypass expired-cert
server-name-bypass explicit-class-list sni_bypass
server-name-bypass enable-log
!
slb virtual-server ep_in_vs 172.16.48.135
port 443 https
source-nat pool inside_2
service-group adsg
template client-ssl cliss1
!
slb virtual-server in_https 0.0.0.0 acl name sni_bypass
port 0 https
source-nat pool inside_2
service-group sgtcp
template client-ssl cliss1
no-dest-nat
!
logging monitor debugging
!
logging syslog debugging
!
logging console debugging
!
logging host 192.168.90.132 port 24680
```

```
!  
harmony-controller profile  
host 192.168.96.250 use-mgmt-port port 443  
provider root  
user-name _a10_hc_device  
cluster-name ck  
cluster-id 5359a844-e8eb-11ea-8882-424130b0cd9c  
password encrypted  
QUzWauG4s05NueHYPZASxzwQjLjV2wDnPBCMuNXbAOc8EIy41dsA5zwQjLjV2wDn  
register  
thunder-mgmt-ip 192.168.90.48  
!
```

CLI Configuration for Shared Partition (Outside)

```
active-partition adc  
!  
access-list 11 permit 10.10.48.140 0.0.0.0  
!  
interface ethernet 1  
enable  
ip address 192.168.92.48 255.255.255.0  
ip nat outside  
!  
interface ethernet 6  
enable  
ip address 10.10.48.143 255.255.255.0  
ip allow-promiscuous-vip  
ip nat inside  
!  
ip nat pool outside 192.168.92.136 192.168.92.139 netmask /30  
!  
ip route 0.0.0.0 /0 192.168.92.254  
!  
ip route 172.16.48.0 /24 10.10.48.133  
!  
slb template server-ssl srvrssl  
!  
slb server gw 192.168.92.254
```

```
health-check-disable
port 0 tcp
health-check-disable
port 0 udp
health-check-disable
!
slb service-group gw_TCP tcp
health-check-disable
member gw 0
!
slb service-group gw_UDP udp
health-check-disable
member gw 0
!
slb template http bypass
non-http-bypass service-group gw_TCP
!
slb virtual-server out_http 0.0.0.0 acl 11
port 0 http
aflex sni_bypass
source-nat pool outside
service-group gw_TCP
template http bypass
template server-ssl srvrssl
no-dest-nat
!
logging monitor debugging
!
logging syslog debugging
!
logging console debugging
!
logging host partition shared
!
```

aFlex script used to separate bypassed and decrypted traffic

```
when CLIENT_ACCEPTED {
SSL::disable serverside
}
```

```
when HTTP_REQUEST {  
  SSL::enable serverside  
}
```

SNI Support on vThunder

ACOS provides support for the SNI extension to vThunder models. The SNI is an extension to TLS that allows a single IP address to host multiple domain names, with a separate certificate for each domain.

The client-SSL template bound to the virtual port can contain multiple certificates. When you add a certificate and key to a client-SSL template, you can specify the domain name (“server name”) that the certificate and key belong to. When a client sends an SSL session setup request to the VIP, ACOS sends the server certificate for the requested domain name, based on the configuration in the client-SSL template.

In addition to certificates and keys for individual domain names, a client-SSL template also can contain one “default” certificate and key. If the template does not have a certificate for the domain name requested by the client, ACOS sends the default certificate instead.

- ACOS 2.7.2 adds SNI support to vThunder models. Previous releases support the feature on hardware models but not on vThunder models.
- The ACOS configuration does not contain any SLB server certificates by default. The “default” certificate and key in a client-SSL template must be imported or generated in ACOS, then added to the template. If you add them to the template without associating them with a domain name, then they become the default certificate and key for the template.
- SSL Intercept, a feature on certain hardware models that uses SNI support, is not supported on vThunder devices. This enhancement does not provide SSL Intercept support on vThunder models.

Configuring an SSL VIP TLS SNI (CLI Procedure)

The commands in this section configure an SSL VIP that serves the following domains:

- www.example.com
- www.example2.com

- mail.example.com

This configuration allows the ACOS device to set up secure SSL sessions with a client who sends requests to 192.168.2.69:443. ACOS selects a server certificate to send to the client based on the domain name requested by the client.

This example assumes the certificates and keys were already imported into or generated in ACOS.

The `slb template client-ssl cssl` command configures the client-SSL template and places the CLI in template configuration mode where the following commands are available:

- The certificate commands add the default certificate and key.
- The server-name commands add the certificates and keys for specific domain names.
- The “cert2” and “cert3” certificates are used for SSL session setup requests to domains `www.example2.com` and `mail.example.com`, respectively.
- The “def_cert” certificate is used for requests to any other domain name, such as `www.example.com`.

Configuring an SSL VIP TLS SNI (CLI Example)

These commands bind the client-SSL template to the SSL virtual port:

```
ACOS(config)# slb virtual-server example 192.168.2.69
ACOS(config-slb vserver)# port 443 ssl
ACOS(config-slb vserver-vport)# template client-ssl cssl
ACOS(config-slb vserver-vport)# exit
```

Configuring TLS SNI

Before creating the certificate-domain mappings, import the server certificates onto the ACOS device. See [Importing a Certificate and Key](#).

Configuring Static TLS SNI (CLI Procedure)

To map a certificate to a domain, use the `server-name` command at the configuration level for the client-SSL template:

```
ACOS(config)# slb template client-ssl clientssl
```

```
ACOS(config-client ssl)# server-name SNIserver cert SNICert key SNIKey  
pass-phrase SNIHelp
```

Configuring Dynamic SNI Extension Support (CLI Procedure)

To enable dynamic SNI extension support, use the `server-name-auto-map` command at the configuration level for the client-SSL template:

```
ACOS(config)# slb template client-ssl clientssl  
ACOS(config-client ssl)# server-name-auto-map
```

Configuring TLS SNI (GUI Procedure)

The configuration page for client-SSL templates has a Server Name Indication section. In this section, to create a certificate-domain mapping:

1. Enter the domain name in the Server Name field.
2. Select the certificate from the Server Certificate drop-down list.
3. Select the certificate's private key from the Server Private Key drop-down list.
4. Click Add.
5. Repeat for each mapping.

Configuring Dynamic SNI Transmission for Inbound SSLi

SSLi is deployed in ACOS as a dual proxy, where client-side (decryption) and server-side (re-encryption) TLS negotiations are handled independently. Therefore, ACOS preserves the SNI (Server Name Indication) information from the client's TLS request while decrypting and forwards it to the server side while re-encrypting. This allows SSLi to inspect TLS sessions on multi-hosted servers.

For outbound (or forward-proxy) SSLi over HTTPS, ACOS forwards the SNI information when forward-proxy-enable is configured under the client-SSL and server-SSL templates. For non-HTTP-based TLS sessions, an additional command, `ssli-sni-hash-enable` under `slb common`, is also configured. The `ssli-sni-hash-enable` command allows ACOS to maintain an internal hashing mechanism for SNI forwarding as long as SSLi is deployed in a single appliance. For a dual-appliance SSLi setup, the SNI is conveyed through a special header in the decrypted payload.

For inbound (or reverse-proxy) SSLi, ACOS can leverage the HTTP HOST header from the decrypted HTTPS request in deriving the SNI for the re-encryption process. This does not require any additional CLI commands. However, it fails to derive the SNI if the decrypted HTTPS request does not contain any HOST header or is dealing with non-HTTP-based applications.

This release overcomes all the above-mentioned limitations with SNI forwarding by extending the SNI hashing mechanism to inbound SSLi and dual-appliance SSLi deployments.

This functionality applies to:

- Non-HTTP-based applications in outbound and inbound SSLi for dynamic port inspection.
- HTTP-based applications without a "HOST" header for the inbound SSLi.
- All SSLi [Deployments](#) on Thunder SSLi or Thunder CFW device platforms.
- SSL proxy and generic proxy configurations.
- All SSL modules except the new Nitrox V (N5), which only support inbound SSLi.

Key Considerations

- The `ssli-inbound-enable` option under the client SSL template enables the basic functions of inbound SSLi with dynamic SNI forwarding. However, as the server-SSL template does the same task as the general SSLi (outbound SSLi), you can continue to use the `forward-proxy-enable` option under the server-SSL template.

NOTE: The `forward-proxy-enable` and `ssli-inbound-enable` options cannot be enabled simultaneously under the client SSL template.

- If one SNI requires this feature and the client SSL template is already associated, enable the `ssli-inbound-enable` option in the associated client SSL template.
- The source NAT configuration on the client SSL template (in inbound SSLi) will not work with this feature.

CLI Configuration

For a single-device deployment:

1. Enable the **ssli-sni-hash-enable** command under SLB common.

```
ACOS(config)# slb common
ACOS(config-common)# ssli-sni-hash-enable
```

2. Enable the **ssli-inbound-enable** command under the client SSL template.

```
ACOS(config)# slb template client-ssl client-tmpl
ACOS(config-client ssl)# ssli-inbound-enable
```

3. Bind the client SSL template with the virtual port.

```
ACOS(config)# slb virtual-server inbound 0.0.0.0 acl 101
ACOS(config-slb vserver)# port 0 ssl-proxy
ACOS(config-slb vserver-vport)# template client-ssl client-tmpl
```

For a multiple-device deployment:

If you use multiple devices, use the management IP Address of both the devices to set up a TCP connection to relay the SNI information. Therefore, one client-side device can send the SNI information only to one server-side device and vice versa.

- If the connection is not established between the client-side and server-side for SNI forwarding, the server-side application connection will be terminated.
- If the server-side does not receive the SNI information in time, the server-side application connection will be terminated.

NOTE: The single device deployment (inbound SSLi) cannot be used in conjunction with multiple device deployment.

1. Configure the client SSL template and enable the **ssli-inbound-enable** command.

```
ACOS(config)# slb template client-ssl client-tmpl
ACOS(config-client ssl)# ssli-inbound-enable
```

2. Configure the client-side management IP and server-side port number.

```
ACOS(config)# slb common
ACOS(config-common)# ssli-sni-hash-enable clientside-ip 10.65.68.199
port 5555
```

OR

```
ACOS(config)# slb common
ACOS(config-common)# ssli-sni-hash-enable clientside-ipv6
2001:1:1:1::18 port 45253
```

3. Configure the server-side management IP and server-side port number on the client-side.

```
ACOS(config)# slb common
ACOS(config-common)# ssli-sni-hash-enable serverside-ip 10.65.68.200
port 5555
```

OR

```
ACOS(config)# slb common
ACOS(config-common)# ssli-sni-hash-enable serverside-ipv6
2001:1:1:1::19 port 45253
```

NOTE: It is recommended to configure either the IPv4 or IPv6 management IP addresses on both the devices. Additionally, avoid using the common port numbers like 22, 80, and 443.

Consolidated Configuration Example

ACOS Client Side

```
ACOS (config)#show run
!Current configuration: 783 bytes
!Configuration last updated at 04:24:06 IST Thu Oct 6 2022
!Configuration last saved at 04:44:30 IST Thu Oct 6 2022
!64-bit Advanced Core OS (ACOS) version 6.0.0-d, build 323 (Oct-05-
2022,09:10)
!
access-list 101 4 permit ip any any
!
terminal idle-timeout 0
!
vlan 33
    tagged ethernet 11
    router-interface ve 33
!
hostname client-side-ax
```

```
!  
!  
interface management  
    ip address 10.65.19.199 255.255.255.0  
    ip control-apps-use-mgmt-port  
    ip default-gateway 10.65.19.1  
!  
interface ethernet 1  
    enable  
    ip address 4.4.4.1 255.255.255.0  
!  
interface ethernet 2  
!  
...  
interface ethernet 11  
    enable  
    ip allow-promiscuous-vip  
!  
...  
interface ethernet 20  
!  
interface ve 33  
    ip address 6.6.33.245 255.255.255.0  
    ip allow-promiscuous-vip  
!  
!  
ip route 8.8.133.0 /24 4.4.4.2  
!  
slb common  
    ssli-sni-hash-enable serverside-ip 10.65.18.199 port 5555  
!  
slb server outside-ax 4.4.4.2  
    port 8080 tcp  
    health-check-disable  
!  
slb service-group outside-ax-grp tcp
```

```
member outside-ax 8080
!
slb template client-ssl client-tmpl
  ssli-inbound-enable
  certificate 4k key 4k
!
slb virtual-server inbound 0.0.0.0 acl 101
  port 0 ssl-proxy
  service-group outside-ax-grp
  template client-ssl client-tmpl
!
!
end
```

ACOS Server Side

```
ACOS(config)#show run
!Current configuration: 858 bytes
!Configuration last updated at 04:58:14 IST Thu Oct 6 2022
!Configuration last saved at 04:58:20 IST Thu Oct 6 2022
!64-bit Advanced Core OS (ACOS) version 6.0.0-d, build 323 (Oct-05-
2022,09:10)
!
terminal idle-timeout 0
!
vlan 133
  tagged ethernet 11
  router-interface ve 133
!
hostname serverside-ax
!
!
interface management
  ip address 10.65.18.199 255.255.255.0
  ip control-apps-use-mgmt-port
  ip default-gateway 10.65.18.1
!
interface ethernet 1
  enable
  ip address 4.4.4.2 255.255.255.0
```

```
ip allow-promiscuous-vip
!
interface ethernet 2
!
...
interface ethernet 11
    enable
!
...
interface ethernet 20
!
interface ve 133
    ip address 8.8.133.245 255.255.255.0
!
!
ip route 6.6.33.0 /24 4.4.4.1
!
slb common
    ssli-sni-hash-enable clientside-ip 10.65.19.199 port 5555
!
slb template server-ssl server-tmpl
    forward-proxy-enable
!
slb server backend-srv 8.8.133.105
    port 443 tcp
        health-check-disable
    port 9443 tcp
        health-check-disable
!
slb service-group backend-srv-grp tcp
    member backend-srv 443
!
slb service-group backend-srv-grp-9443 tcp
    member backend-srv 9443
!
slb virtual-server serverside 0.0.0.0
    port 9443 tcp-proxy
        service-group backend-srv-grp-9443
        template server-ssl server-tmpl
```

```
no-dest-nat
!  
!  
end
```

TLS 1.3 Support for Software SSL

ACOS supports RFC 8446, TLS 1.3 protocol to provide faster and more secured channel for communication. To view the SSL software mode, use `show hardware` command.

NOTE: TLS 1.3 is currently supported for Software SSL, while ACOS SSLi only supports TLS 1.2 by default.

To configure cipher template with TLS 1.3 and set the cipher:

```
ACOS(config)# slb template cipher c1  
ACOS(config-cipher)# tls1_3 TLS_AES_256_GCM_SHA384 priority 10
```

NOTE: If you are working on hardware SSL, TLS 1.3 cipher and version command displays warning to notify that TLS 1.3 is only supported in software SSL mode.

CAs and CSRs Management

Installing SSL resources on the ACOS device enables the device to provide SSL services on behalf of real servers.

The following topics are covered:

Importing a Certificate and Key	240
Generating a Certificate Signing Request (CSR)	243
Generating a Self-Signed Certificate and Key	244
Generating an SSL Certificate – Private Key File with a CSR	246
Installing Certificates	250
Implementing Certificates to SSL Templates	253
Creating Multiple CA Certificate in Server-SSL Templates	254
Binding Server-SSL Templates to Individual Real Ports	257
Configuring Email Notification for SSL Certificate Expiration	259
Converting Certificates and CRLs to PEM Format	260
Importing a Certificate Revocation List (CRL)	262
Exporting Certificates, Keys, and CRLs	263
Importing a CA Cert and Private Key	264
Configuring Forward Proxy Alternate Signing Cert	265
Deleting Certificate Files	266
Configuring Simple Certificate Enrollment Protocol (SCEP) Certificates	266
Configuring Automatic Certificate Management Environment (ACME) Certificates	272

Importing a Certificate and Key

To import certificate and key files, place them on the PC that is running the ACOS GUI or CLI session, or onto a PC or file server that ACOS can reach and fetch the files.

The following topics are covered:

Importing Individual Files	241
SSL Certificate and Key Files Bulk Import	242

Importing Individual Files

To import an SSL certificate CA certificate, certificate chain, or private key follow these instructions.

Importing Certificates (GUI Procedure)

1. Navigate to **ADC >> SSL Management >> SSL Certificates**.
2. Click **Import** to import a certificate or certificate chain.
3. In the **File Name** field, enter a name for the certificate.
4. In the **Import** field, select the item you want to import
5. In the **Import Certificate from** field, select **Local** to import from a local drive on your management PC, **Remote** to import from a remote location, or **Text** to import from the text box that appears
6. In the **SSL or CA Certificate** field, select either **SSL Certificate** or **CA Certificate**.
7. If you are importing a CA-signed certificate for which you used ACOS to generate the CSR, you do not need to import the key. The key is automatically generated by ACOS when you generate the CSR.
8. In the **Certificate Format** field, select the file format of the certificate you are importing. Certificate and private keys in a single file use the PFX format which is automatically chosen.
9. The **Certificate Source** field provides the location and other fields you need to import the selected item.
10. Decide whether to enable or disable the **Overwrite Existing File** option.
11. Click **Import**.

Importing Certificates (CLI Procedure)

- Use the `import cert` command to import a certificate or certificate chain that you will be using with its private key to create proxied certificates for SSL handshaking with clients in the SSLi, SSL Proxy or SSL offload applications. If you import the cert

and its key in a single file use the PFX format.

An example of importing a certificate for SSLi is found in [Importing a CA Cert and Private Key](#).

- Use the `import ca-cert` command to import a certificate or a certificate chain for certificates for verifying SSL servers and authenticating clients and other purposes. However the CA certificate cannot be used for creating proxied signed certificates for handshaking with clients.

NOTE: If you are importing a CA-signed certificate for which you used ACOS to generate the CSR, you do not need to import the key. The key is automatically generated by ACOS when you generate the CSR.

- Use the `import cert-key` command to import a private key.

SSL Certificate and Key Files Bulk Import

You can import or export SSL files in bulk, as .tgz archives.

Bulk Import and Export of Certificate and Key Files (GUI Procedure)

The steps for importing or exporting SSL files are the same for individual files and for bulk archives. (For information, see [To import an SSL certificate CA certificate, certificate chain, or private key follow these instructions](#), the GUI online help.)

Bulk Import and Export of Certificate and Key Files (CLI Procedure)

To import a .tgz archive of SSL certificate files, key files, or CRL files, use the following commands:

- `import cert` – The archive contains only certificate files.
- `import cert-key bulk` – The archive contains both certificate and key files
- `import crl` – The archive contains only CRL files.
- `import key` – The archive contains only Key files.

Generating a Certificate Signing Request (CSR)

The following procedures generates a CSR that you can send to a server, so that the server can send the CSR to a CA to request a new CA-signed certificate or renew an existing one.

This process also creates a public key - private key pair. The public key is sent in the CSR. The private key used to encrypt the CSR.

Generating a CSR (GUI Procedure)

1. Navigate to **ADC >> SSL Management >> SSL Certificates**.
2. Click **+Create**. The **Create SSL Certificates** dialog window appears.
3. In the **Create As** field, select **CSR**.
4. In the **File Name** field, type the name you certificate that will be provided by the CA.
5. In the **Digest** field, select the hashing algorithm used. The default is **sha1**.
6. In the **Cert Type** field, select **RSA** or **ECDSA** depending on which cryptography standard you want.
7. The **Common Name** field is required.

To create a wild card certificate request, use an asterisk for the first part of the common name. For example, to request a wild card certificate for domain example.com and it sub-domains, enter *.example.com as the common name.

8. The **Division, Organization, Locality, State or Province**, and **Email** fields are optional.
9. Enter a number the **Valid Days** (how many days the key will remain valid) and **Key Size**, or accept the defaults 730 days and 1024 bytes.
10. Click **OK**.
11. Verify the newly created SSL certificate appears in the **ADC >> SSL Management >> SSL Certificates** page. Check the matching **Name** and **Common Name** fields. The **Type** should be **key**, and the expiration should match the number of days the certificate remains valid. See RFC 6125 for help in reading the **Issuer** field.

Generating a CSR (CLI Example)

- Use `pki create csr` command in global configuration mode to generate an RSA type of certificate signing request (CSR). In this example, the CSR name is CSR1.

```
ACOS(config)# pki create csr CSR1 generate certtype rsa
input key bits(1024,2048,4096) default 1024:
input Common Name, 1~64:CSR1
input Division, 0~31:
input Organization, 0~63:
input Locality, 0~31:
input State or Province, 0~31:
input Country, 2 characters:US
input email address, 0~64:admin@a10networks.com
ACOS(config)#
```

- To create wildcard certificates, use an asterisk as the first part of the common name. For example, to create a wildcard certificate for domain example.com and its sub-domains, enter the following common name: *.example.com.
- Use `show pki certificate csr1 detail` to show the CSR created.

Generating a Self-Signed Certificate and Key

In the following procedure the certificate file also includes the corresponding private key.

See RFC 6125 for help in filling out some of the following fields.

Generating a Self-Signed Certificate and Key (GUI Procedure)

1. Navigate to **ADC >> SSL Management >> SSL Certificates**.
2. Click **+Create**. The **Create SSL Certificates** dialog window appears.
3. In the **Create As** field, select **Certificate**.
4. In the **File Name** field, type the name you certificate that will be generated.
5. Do not enable **CSR Generate**. This checkbox enable the creation of a CSR.
6. In the **Cert Type** field, select **RSA** or **ECDSA** depending on which cryptography standard you want.
7. The **Common Name** field is required.

NOTE: If you need to create a request for a wildcard certificate, use an asterisk as the first part of the common name. For example, to request a wildcard certificate for domain example.com and its sub-domains, enter the following common name: *.example.com.

8. The **Division, Organization, Locality, State or Province**, and **Email** fields are optional.
9. Enter a number for the **Valid Days** (how many days the key will remain valid) and **Key Size**, or accept the defaults 730 days and 1024 bytes.
10. Click **OK**.
11. Verify the newly created SSL certificate appears in the **ADC >> SSL Management >> SSL Certificates** page. Check matching **Name** and **Common Name** fields. The **Type** should be **certificate/key**, and the expiration should match the number of days the certificate remains valid. See RFC 6125 for help in reading the **Issuer** field.

Generating a Self-Signed Certificate and Key (CLI Example)

To generate a self-signed certificate, use the following command at the global configuration level of the CLI:

The `pki create certificate` command generates and initializes a self-signed certificate and key. When creating a self-signed certificate it must be pushed out to inside clients (clients on the internal network). If the certificate is not pushed, the internal hosts get an SSL “untrusted root” error whenever they try to connect.

The key length, common name, and number of days the certificate is valid are required. The other information is optional. The default key length is 1024 bits. The default number of days the certificate is valid is 730.

```
ACOS(config)# pki create certificate enterpriseABC-selfsigned certtype
rsa
input key bits(1024,2048,4096) default 1024:
input Common Name, 1~64: enterpriseABC-selfsigned
input Division, 0~31:
input Organization, 0~63:
input Locality, 0~31:
input State or Province, 0~31:US
input Country, 2 characters:US
```

```
input email address, 0~64:
input valid days, 30~3650, default 730:
ACOS(config)#
```

To create a wildcard certificate, use an asterisk as the first part of the common name. For example, to create a wildcard certificate for domain example.com and its sub-domains, enter the following common name: *.example.com

Generating an SSL Certificate – Private Key File with a CSR

The following procedure generates an SSL self-signed certificate with private key and also generates a CSR that you can send to a publicly recognized CA to register your self-signed SSL certificate.

This process also creates a public key - private key pair. The public key is sent in the CSR. The private key is used to encrypt the CSR and also to create the SSL proxied certificate used in the ACOS SSLi, SSL-Offload, and SSL-Proxy applications.

GUI Configuration

1. Navigate to **ADC >> SSL Management >> SSL Certificates**.
2. Click **+Create**. The **Create SSL Certificates** dialog window appears.
3. In the **Create As** field, select **Certificate**.
4. In the **File Name** field, type the name of the certificate that will be generated.
5. Click the **CSR Generate** box to enable the creation of a CSR.
6. In the **Cert Type** field, select **RSA** or **ECDSA** depending on which cryptography standard you want.
7. The **Common Name** field is required.

NOTE:	If you need to create a request for a wildcard certificate, use an asterisk as the first part of the common name. For example, to request a wildcard certificate for domain example.com and its sub-domains, enter the following common name: *.example.com.

8. The **Division, Organization, Locality, State or Province**, and **Email** fields are optional.

9. Enter a number the **Valid Days** (how many days the key will remain valid) and **Key Size**, or accept the defaults 730 days and 1024 bytes.
10. Click **OK**.
11. Verify the newly created SSL certificate appears in the **ADC >> SSL Management >> SSL Certificates** page. Check the matching **Name** and **Common Name** fields. The **Type** should be **certificate/key**, and the expiration should match the number of days the certificate remains valid. See RFC 6125 for help in reading the **Issuer** field. The GUI does not display the CSR separately.

CLI Configuration

- Use the `pki create cert` command in global configuration mode to generate a self-signed SSL certificate and corresponding CSR. In this example, CSR file name is `csr`, CSR renewal file name is `Cert-CSR-both`, the file transport protocol is FTP, and the URL specifying where the CSR is sent is 192.168.1.10.

```
ACOS(config)# pki create cert Cert-CSR-both certtype rsa csr-generate
input key bits(1024,2048,4096) default 1024:
input Common Name, 1~64:Cert-CSR-both
input Division, 0~31:
input Organization, 0~63:
input Locality, 0~31:
input State or Province, 0~31:
input Country, 2 characters: US
input email address, 0~64: admin@a10networks.com
```

- In the above example, the CSR is generated without the root CA extensions. The syntax for the command that creates a CSR with root CA extensions follows:

```
ACOS(config)# pki create cert Cert-CSR-both certtype rsa rootca
```

- If you need to create a wildcard certificate, use an asterisk as the first part of the common name. For example, to create a wildcard certificate for domain `example.com` and its sub-domains, enter the following common name:
`*.example.com`
- Use `show pki csr Cert-CSR-both detail` to show the cert created.
- Use `show pki certificate Cert-CSR-both detail` to show the CSR created.

```
ACOS(config)# show pki cert Cert-CSR-both detail
Certificate:
```

```

Data:
  Version: 3 (0x2)
  Serial Number: 13866059162969540330 (0xc06e2357db5986ea)
  Signature Algorithm: sha1WithRSAEncryption
  Issuer: C=AF, CN= Cert-CSR-both
  Validity
    Not Before: Jan 31 05:20:36 2017 GMT
    Not After : Jan 31 05:20:36 2019 GMT
  Subject: C=AF, CN=Cert-CSR-both
  Subject Public Key Info:
    Public Key Algorithm: rsaEncryption
    Public-Key: (1024 bit)
    Modulus:
      00:96:fc:1d:cc:63:ea:c1:a9:c7:1d:dd:c5:9c:72:
      08:61:27:b7:67:1a:27:c7:f7:39:ca:9c:81:ac:f0:
      f8:05:89:1a:66:25:cf:0b:1e:55:cc:cf:8b:89:91:
      58:c5:e9:8c:b8:44:f1:d5:42:94:b1:e9:5a:a6:10:
      05:28:0d:a2:84:a6:73:a8:64:66:e4:72:cc:c8:1b:
      39:c9:4a:9c:a6:b3:67:e1:4a:d8:9d:a3:fa:bd:7c:
      0e:ad:c1:35:6c:6f:54:68:0a:5f:54:67:61:fd:6a:
      e2:55:2f:85:11:76:f3:96:c0:5c:55:11:63:a6:21:
      41:65:6f:da:67:d5:e8:7e:ff
    Exponent: 65537 (0x10001)
  Signature Algorithm: sha1WithRSAEncryption
    7d:ac:29:e8:a9:b5:2f:69:43:d2:a1:8b:7c:6d:8e:b5:21:f8:
    30:cc:7a:4f:61:71:23:87:51:2c:da:ce:89:14:29:55:f3:81:
    97:c0:2f:a7:e3:8a:4b:7d:d2:f7:cb:00:14:ce:91:db:1f:3a:
    db:a0:a0:a9:90:b8:a1:b0:7a:16:e3:54:23:94:e2:48:fb:92:
    36:0c:6d:c4:be:fd:79:77:41:6c:3a:19:3f:72:29:c6:95:f1:
    c5:41:d8:a8:ed:18:2e:ca:66:1a:af:39:16:79:10:03:d6:f0:
    95:10:93:1f:13:c8:96:70:c5:3f:97:8b:96:e1:d5:78:8d:b7:
    c7:0c
  SHA1
  Fingerprint=D5:9A:B6:96:66:5D:B9:77:FE:1F:28:B4:BC:A9:3A:43:5D:2D:C7:9
  8
  -----BEGIN CERTIFICATE-----
  MIIBxjCCAS+gAwIBAgIJAMBuI1fbWYbqMA0GCSqGSIb3DQEBBQUAMCUxCzAJBgNV
  BAYTAkFGMRYwFAYDVQQDEw1DZXJ0LUNTU11ib3R0MB4XDTE3MDEzMTA1MjAzNloX

```



```
DTE5MDEzMTA1MjAzNlowJTElMAkGA1UEBhMCQUYxFjAUBgNVBAMTDUN1cnQtQ1NS
LWJvdGgwZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAJb8Hcxj6sGp3dxZxy
CGEnt2caJ8f3Ocqcgazw+AWJGmYlzwseVczPi4mRWMXpjLhE8dVC1LHpWqYQBSgN
ooSmc6hkZuRyzMgbOclKnKazZ+FK2J2j+r18Dq3BNWxvVGgKX1RnYf1q4lUvhRF2
85bAXFURY6YhQWVv2mfV6H7/AgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAfawp6Km1
L2lD0qGLfG2OtSH4MMx6T2FxI4dRLNrOiRQpVfOB18Avp+OKS33S98sAFM6R2x86
26CgqZC4obB6FuNUI5TiSPuSngxtxL79eXdBbDoZP3IpxpXxxUHYq00YLspmGq85
FnkQA9bwlRcTHxPILnDFP5eLluHVeI23xww=
-----END CERTIFICATE-----
```

key size: 1024

ACOS(config)# **show pki csr Cert-CSR-both detail**

Certificate Request:

Data:

Version: 0 (0x0)

Subject: C=AF, CN=Cert-CSR-both

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

Public-Key: (1024 bit)

Modulus:

```
00:96:fc:1d:cc:63:ea:c1:a9:c7:1d:dd:c5:9c:72:
08:61:27:b7:67:1a:27:c7:f7:39:ca:9c:81:ac:f0:
f8:05:89:1a:66:25:cf:0b:1e:55:cc:cf:8b:89:91:
58:c5:e9:8c:b8:44:f1:d5:42:94:b1:e9:5a:a6:10:
05:28:0d:a2:84:a6:73:a8:64:66:e4:72:cc:c8:1b:
39:c9:4a:9c:a6:b3:67:e1:4a:d8:9d:a3:fa:bd:7c:
0e:ad:c1:35:6c:6f:54:68:0a:5f:54:67:61:fd:6a:
e2:55:2f:85:11:76:f3:96:c0:5c:55:11:63:a6:21:
41:65:6f:da:67:d5:e8:7e:ff
```

Exponent: 65537 (0x10001)

Attributes:

a0:00

Signature Algorithm: sha1WithRSAEncryption

```
7f:2e:82:ef:b8:ed:5d:bc:78:4a:8c:25:5e:df:46:69:11:21:
74:7e:1e:fa:29:08:d0:ea:27:1a:25:fa:4b:ae:e2:78:08:2a:
63:ed:c9:0b:8d:0b:f6:d7:1e:07:10:dc:12:2b:ff:b0:0f:4a:
d6:68:a0:e1:ac:80:8b:d7:bb:f2:a3:6e:e2:74:c6:31:6c:44:
cc:45:c3:f8:2c:85:58:cb:a9:dc:28:bb:3b:72:0f:38:95:68:
```

```

1d:f4:09:9b:08:0f:f4:49:a5:9d:4d:91:d1:df:82:6c:63:60:
b8:74:d6:13:67:dd:81:c1:a6:af:ee:fa:22:7b:b2:a4:1e:e3:
b6:3d
-----BEGIN CERTIFICATE REQUEST-----
MIIBZDCBzgIBADA1MQswCQYDVQQGEwJBRjEWMBQGA1UEAxMNQ2VydC1DU1ItYm90
aDCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA1vwdzGPqwanHHd3FnHIIYSe3
Zxonx/c5ypyBrPD4BYkaZiXPCx5VzM+LiZFYxemMuETx1UKUselaphAFKA2ihKZz
qGRm5HLMYBs5yUqcprNn4UrYnaP6vXwOrcE1bG9UaApfVGdh/WriVS+FEXbzlsBc
VRfjpiFBZW/aZ9Xofv8CAwEAAaAAMA0GCSqGSIB3DQEBBQUAA4GBAH8ugu+47V28
eEqMJV7fRmkRIXR+HvopCNDqJxol+kuu4ngIKmPtyQuNC/bXHgcQ3BIr/7APStZo
oOGsgIvXu/KjbuJ0xjFsRMxFw/gshVjLqdwouzyDziVaB30CZsID/RJpZ1NkdHf
gmxjYLh01hNn3YHBpq/u+iJ7sqQe47Y9
-----END CERTIFICATE REQUEST-----

```

Installing Certificates

To configure an ACOS device to perform SSL processing on behalf of real servers, you must install a certificate on the ACOS device. This certificate is the one that the ACOS device will present to clients during the SSL handshake. You also must configure a client-SSL template, add the key and certificate to the template, and bind the template to the VIP that will be requested by clients.

You can install a CA-signed certificate or a self-signed certificate (described in [CA-Signed and Self-Signed Certificates](#)).

This section gives an overview of the process for each type of certificate. Detailed procedures are provided later in this section.

Requesting and Installing a CA-Signed Certificate

To request and install a CA-signed certificate, use the following process. For detailed steps, see [CAs and CSRs Management](#) and [Importing a Certificate and Key](#).

1. Create an encryption key.
2. Create a Certificate Signing Request (CSR).

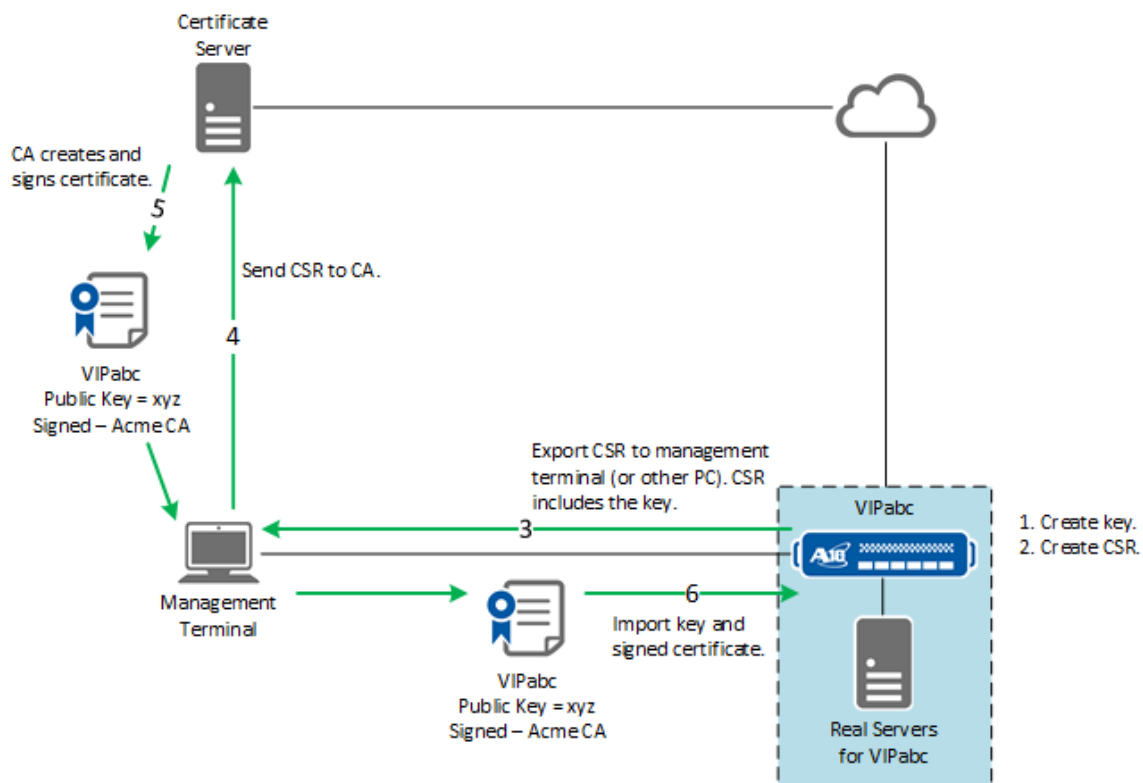
The CSR includes the public portion of the key, as well as information you enter when creating the CSR.

You can create the key and CSR on an ACOS device or a server running openssl or a similar application.

3. Submit the CSR to the CA.
4. If the CSR was created on the ACOS device, do one of the following:
 - Copy and paste the CSR from the ACOS CLI or GUI onto the CSR submission page of the CA server.
 - Export the CSR to another device, such as the PC from which you access the ACOS CLI or GUI. Email the CSR to the CA, or copy-and-paste it onto the CSR submission page of the CA server.
5. If the CSR was created on another device, email the CSR to the CA, or copy-and-paste it onto the CSR submission page of the CA server.
6. After receiving a signed certificate and the CA's public key from the CA, import them to the ACOS device.
7. If the key and certificate are provided by the CA in separate files (PKCS #7 format), import the certificate. The key does not need be imported if the CSR was created on the ACOS device because the key is already on the ACOS device. If the certificate is not in PEM format, specify the certificate format (type) when importing it.
8. If the CSR was not created on the ACOS device, you do need to import the key also.
9. If the key and certificate are provided by the CA in a single file (PKCS #12 format), specify the certificate format (type) when you import it. If the CSR was not created on the ACOS device, you need to import the key also. See [Converting SSL Certificates to PEM Format \(Windows PC Procedure\)](#).
10. If applicable, import the certificate chain onto the ACOS device. The certificate chain must be a single text file, beginning with a root CA's certificate at the top, followed in order by each intermediate signing authority's certificate. (See [Certificate Chain](#).)

[Figure 24](#) shows the most common way to obtain and install a CA-signed certificate onto the ACOS device. You also may need to install a certificate chain file.

Figure 24 : Obtaining and Installing Signed Certificate from CA

**NOTE:**

As an alternative to using a CA, you can use an application such as openssl to create a certificate, then use that certificate as a CA-signed certificate to sign another certificate. However, in this case, a client's browser is still likely to display a certificate warning to the end user.

Installing a Self-Signed Certificate

To install a self-signed certificate instead of a CA-signed certificate:

- Create an encryption key.
- Create the certificate.

See [Generating a Self-Signed Certificate and Key](#).

Implementing Certificates to SSL Templates

After creating or importing certificates and keys on the ACOS device, you must add them to an SSL template, then bind the template to a VIP, in order for them to take effect.

Creating an SSL Template (GUI Procedure)

1. Navigate to **ADC >> Templates >> SSL**.
2. Click **Create**, and:
3. Select **Client SSL** to create a template for SSL traffic between the ACOS device (VIP) and clients.
4. Select **Server SSL** to create a template for SSL traffic between the ACOS device and servers.
5. Enter or select the configuration options; refer to the online help for information about the fields on this GUI page.
6. When finished, click **OK**.

Creating an SSL Template (CLI Example)

Use one of the following commands at the global configuration level of the CLI:

- `slb template client-ssl` – creates template for SSL traffic between ACOS device (VIP) and clients.

```
ACOS(config)# slb template client-ssl TMPLT-C
ACOS(config-client ssl)# exit
```

- `slb template server-ssl` – creates template for SSL traffic between ACOS device and servers.

```
ACOS(config)# slb template server-ssl TMPLT-S
ACOS(config-server ssl)# exit
```

The command creates the template and changes the CLI to the configuration level for it. Use the commands at the template configuration level to configure template parameters. (For information, see [Certificates in SSL Templates](#) or the CLI Reference.)

Binding an SSL Template to a VIP (GUI Procedure)

1. Navigate to **ADC >> SLB > Virtual Servers**.
2. Click Create to create a new virtual server.
3. Enter the VIP name and IP address.
4. In the Port section, click Create. The Virtual Server Port page appears.
5. Click on “Templates” to expand the Templates section.
6. Select the template from the Client-SSL Template or Server-SSL Template drop-down list.

Binding an SSL Template to a VIP (CLI Example)

- Use one of the following commands at the configuration level for the virtual port on the VIP:

template client-ssl – binds client SSL template to the VIP.

```
ACOS(config)# slb virtual-server VIP-1 10.10.1.1
ACOS(config-slbg vserver)# port 80 ssl-proxy
ACOS(config-slbg vserver-vport)# template client-ssl TMPLT-C
ACOS(config-slbg vserver-vport)# exit
ACOS(config-slbg vserver)# exit
```

template server-ssl – binds server SSL template to the VIP.

```
ACOS(config)# slb virtual-server VIP-2 10.10.2.1
ACOS(config-slbg vserver)# port 80 ssl-proxy
ACOS(config-slbg vserver-vport)# template server-ssl TMPLT-S
ACOS(config-slbg vserver-vport)# exit
ACOS(config-slbg vserver)# exit
```

- Use the same command on each port for which SSL will be used.

Creating Multiple CA Certificate in Server-SSL Templates

If you need to add multiple certificates to a server-SSL template, this section describes how to configure it. A server-SSL template can have multiple CA-signed certificates.

You can add the CA certificates to the server-SSL template in either of the following ways:

- As separate files (one for each certificate)
- As a single file containing multiple certificates

Adding multiple certificates in a single file can simplify configuration. For example, you can export the CA certificates from a web browser into a single file, then import that file onto the ACOS device and add it to a server-SSL template.

Previous releases allow a server-SSL template to have only a single CA-signed certificate.

NOTE: A CA-signed certificate is a certificate signed by a Certificate Authority (CA).

Multiple Certificates in Single File – Preparing the File

You can create the multiple certificate file by exporting the certificates from a browser or you can assemble the file by hand.

To export the certificates from Internet Explorer (IE) version 9:

1. Select **Tools > Internet Options**.
2. Click on the **Content** tab.
3. Click **Certificates**.
4. Click on the **Trusted Root Certification Authorities** tab.
5. Select all the certificates.
6. Click **Export**.
7. Click **Next**.
8. Select PKCS #12 format (PFX), if not already selected.
9. Click **Next**.
10. When prompted for a file password, enter a password to secure the certificate file, and click **Next**.
11. When prompted for a filename:

- a. Click **Browse** to navigate to the save location for the file.
 - b. Enter the filename and click **Save**.
12. Click **Next**.
13. Click **Finish**.
14. On the ACOS device:
 - a. Import the certificate file as a PFX file.
 - b. Use the GUI or CLI to add the certificate file to a server-SSL certificate.
 - c. Bind the server-SSL certificate to the virtual port.

To create the file manually

1. Copy and paste each certificate to a text file. Make sure to include the "-----BEGIN CERTIFICATE-----" and "-----END CERTIFICATE-----" lines for each certificate. For example:

```
-----BEGIN CERTIFICATE-----  
MIIE0zCCA7ugAwIBAgIQGNr  
RniZ96LtKIVjNzGs7SjANBg  
kqhkiG9w0BAQUFADCB  
-----END CERTIFICATE-----  
-----BEGIN CERTIFICATE-----  
U2lnbiwgSW5jLiAtIEZvciBhd  
XRob3JpemVkJHVzZSBvbmx  
5MUUwQwYDVQQDEzxW  
-----END CERTIFICATE-----
```

2. Save the text file.
3. On the ACOS device:
 - a. Import the certificate file as a PEM file.
 - b. Use the GUI or CLI to add the certificate file to a server-SSL certificate.
 - c. Bind the server-SSL certificate to the virtual port.

Binding Server-SSL Templates to Individual Real Ports

For additional flexibility, the ACOS device supports binding of server-SSL templates to individual real ports. This configuration option is useful in cases where the real servers load balanced by a VIP have different SSL settings.

If a server-SSL template is be bound to the virtual port instead, all the real servers load balanced by the VIP must use the same SSL settings.

You can bind a server-SSL template to a real port and also to a virtual port that uses that real port. In this case, the server-SSL template bound to the real port is used for traffic sent to that real port. If you remove the server-SSL template from the real port, the template bound to the virtual port is used instead.

Binding Server SSL Templates to Real Ports (GUI Procedure)

On the configuration page for the real server, in the Port section, select the template from the Server-SSL Template drop-down list.

Binding Server SSL Templates to Real Ports (CLI Procedure)

To bind a server-SSL template to a real port, use the `template server-ssl` command at the configuration level for the real port:

Binding Server SSL Templates to Real Ports (CLI Example)

The following commands import a CA-signed certificate and key:

```
ACOS(config)# import ca-cert CACert88.pem tftp:
Address or name of remote host []?192.168.52.254
File name [/]?CACert88.pem
.0 minutes 1 seconds
ACOS(config)# import key CAkey tftp:
Address or name of remote host []?192.168.52.254
File name [/]?CAkey88
.0 minutes 1 seconds
```

The following commands create a server-SSL template and add the certificate and key to the template:

```
ACOS(config)# slb template server-ssl server-ssl1
ACOS(config-server ssl)# ca-cert CACert88.pem key CAkey88
```

```
ACOS(config-server ssl)# certificate Cert123.pem key key123 pass-phrase  
Pass123  
ACOS(config-server ssl)# exit
```

The following commands bind the server-SSL template directly to a port on a real server:

```
ACOS(config)# slb server rs88 10.8.8.8  
ACOS(config-real server)# port 443 tcp  
ACOS(config-real server-node port)# template server-ssl server-ssl1
```

NOTE: New certificate commands will be lost if downgrade one box to old image that not support new commands.

Server SSL Support for Data Plane Health Monitor

ACOS supports SSL-based health monitoring on the data plane. This is implemented by binding the server SSL to the health monitor. The server SSL can consist of an SSL cipher and other SSL settings for successful SSL negotiation.

The health monitor could utilize SSL acceleration hardware and allow ACOS data plane processes to manage HTTPS connections directly. This improves the scalability and performance of health monitoring systems.

The following configurations are supported:

- Server SSL template configuration can include `version`, `session-ticket-enable`, and `cipher` settings.
- Health monitor configuration can include HTTPS method `expect-cert-name`, `sni host`, `port`, `url`, `maintenance-code`, `http-version2`, `cert` and `key`, and `http-version3` settings and can be bound to a real port or service group.

CLI Configuration

1. Configure the cipher template with strong cipher suites.

In the SSL context, the ciphers will be selected by default. If you select the ciphers for SSL handshake, then you must bind the server SSL template with the cipher template to health monitor. See [Cipher Template Configuration and Usage Guidelines](#).

```
ACOS(config)# slb template cipher cipher_template
ACOS(config-cipher)# TLS1_ECDHE_RSA_AES_256_SHA384
ACOS(config-cipher)# TLS1_ECDHE_ECDSA_AES_256_SHA384
ACOS(config-cipher)# TLS1_ECDHE_RSA_CHACHA20_POLY1305_SHA256
ACOS(config-cipher)# TLS1_ECDHE_ECDSA_CHACHA20_POLY1305_SHA256
ACOS(config-cipher)# TLS1_DHE_RSA_CHACHA20_POLY1305_SHA256
ACOS(config-cipher)# exit
```

2. Configure the server SSL template and bind the cipher template.

```
ACOS(config)# slb template server-ssl hm-server-ssl
ACOS(config-server ssl)# template cipher cipher_template
ACOS(config-server ssl)# exit
```

3. Configure the health monitor and method HTTPS. Bind the server SSL template.

```
ACOS(config)# health monitor https
ACOS(config-health:monitor)# server-ssl hm-server-ssl
ACOS(config-health:monitor)# method https http-version2
```

4. Bind the health monitor to the real port or service-group.

```
ACOS(config)# slb server rs1 10.10.101.65
ACOS(config-real server)# port 443 tcp
ACOS(config-real server-node port)# health-check https
ACOS(config-real server-node port)# exit
ACOS(config-real server)# exit
```

Configuring Email Notification for SSL Certificate Expiration

The ACOS device can send email notification when an SSL certificate is about to expire. This feature sends a daily email listing the certificates that are about to expire or that have recently expired.

By default, this feature is not configured. To configure email notification for certificate expiration, use either of the following methods.

Configuring Email Notification for SSL Certificate Expiration (GUI Procedure)

1. Navigate to **ADC >> SSL Management >> Expiration Mail**.
2. In the SSL Expire Email Address, enter the email address (twice; both address must match) where you want the notifications to be sent.

3. Configure the other fields on this screen as desired; refer to the GUI online help for more information about the fields on this page.
4. Click **Update**.

Configuring Email Notification for SSL Certificate Expiration (CLI Procedure and Example)

To configure email notification for certificate expiration, use the `slb ssl-expire-check` command.

The following example enables certificate notifications to be sent to email address “admin1@example.com”. Expiration notifications are sent beginning 4 days before expiration and continue for 3 days after expiration.

```
ACOS(config)# slb ssl-expire-check email-address admin1@example.com  
before 4 interval 3
```

Converting Certificates and CRLs to PEM Format

The ACOS device supports Privacy Enhanced Mail (PEM) format for certificate files and CRLs.

If a certificate or CRL you plan to import onto the ACOS device is not in PEM format, it must be converted to PEM format.

You do not need to convert the certificate into PEM format before importing it. You can specify the format when you import the certificate. The ACOS device automatically converts the imported certificate into PEM format. (See [Importing a Certificate and Key](#).)

If you prefer to convert a certificate before importing it, see the following sections.

If you have certificates that are in Windows format, use the procedure in this section to convert them to PEM format. For example, you can use this procedure to export SSL certificates that were created under a Windows IIS environment, for use on servers that are running Apache.

This procedure requires a Windows PC and a Unix/Linux workstation.

Converting SSL Certificates to PEM Format (Windows PC Procedure)

1. Start the Microsoft Management Console (mmc.exe).

2. Add the Certificates snap-in:
3. Select File Add/Remove Snap-In. The Add/Remove Snap-In dialog appears.
4. Click **Add**. A list of available snap-ins appears.
5. Select Certificates.
6. Click **Add**.
7. A dialog appears with the following choices: My user account, Service account, and Computer account.
8. Select Computer Account and click Next. The Select Computer dialog appears.
9. Select Local Computer and click Finish.
10. Click **Close**.
11. Click **OK**. The Certificates snap-in appears in the Console Root list.
12. Expand the Certificate folders and navigate to the certificate you want to convert.
13. Select **Action > All Tasks > Export**.

The Export wizard guides you with instructions. Make sure to export the private key too. The wizard will ask you to enter a passphrase to use to encrypt the key.

Converting SSL Certificates to PEM Format (Unix / Linux Workstation Procedure)

1. Copy the PFX-format file that was created by the Export wizard to a UNIX machine.
2. Use OpenSSL to convert the PFX file into a PKCS12 format:

```
$ openssl pkcs12 -in filename.pfx -out pfxoutput.txt
```
3. This command creates a PKCS12 output file, which contains a concatenation of the private key and the certificate.
4. Use the vi editor to divide the PKCS12 file into two files, one for the certificate (.crt) and the other for the private key.
5. To remove the passphrase from the key, use the following command:

```
$ openssl rsa -in encrypted.key -out unencrypted.key
```

Although removing the passphrase is optional, A10 Networks recommends that you remove the passphrase for production environments where Apache must start unattended.

Converting CRLs from DER to PEM Format (Unix / Linux Workstation Procedure)

If you plan to use a Certificate Revocation List (CRL), the CRL must be in PEM format.

To convert Distinguished Encoding Rules (DER) format to PEM format, use the following command on a Unix/Linux machine where the file is located:

```
openssl crl -in filename.der -inform der -outform pem -out filename.pem
```

Importing a Certificate Revocation List (CRL)

To import a CRL, place it on the PC that is running the GUI or CLI session, or onto a PC or file server that can be locally reached over the network.

Importing a CRL (GUI Procedure)

1. Navigate to **ADC >> SSL Management >> Cert Revocation List**.
2. Click **Import**.
3. Complete the fields on this page to navigate to the location of the CRL.
4. Click **Import**.

Importing a CRL (CLI Procedure)

To import a CRL, use the `import crl` command at the Privileged EXEC or global Config level of the CLI:

Refer to the *Command Line Interface Reference* for detailed information about this command.

NOTE: When traffic is flowing, the process of loading a large CRL file will consume a significant amount of time. It is recommended to load large CRL files before the initiation of traffic or during maintenance periods. Additionally, frequent updates of large CRL files should be avoided.

Exporting Certificates, Keys, and CRLs

This section describes how to export SSL resources from the ACOS device to other devices.

Due to a limitation in Windows, it is recommended to use names shorter than 255 characters. Windows allows a maximum of 256 characters for both the file name and the directory path. If the combination of directory path and file name is too long, Windows will not recognize the file. This limitation is not present on machines running Linux/Unix.

Exporting a Certificate and Key (GUI Procedure)

1. Navigate to **ADC >> SSL Management >> SSL Certificates**.
2. To export a certificate:
3. Select the certificate. (Click the checkbox next to the certificate name.)
4. Click **Export**.
5. If the browser security settings normally block downloads, you may need to override the setting. For example, in Internet Explorer, hold the Ctrl key while clicking Export.
6. Click **Save**.
7. Navigate to the save location.
8. Click **Save** again.
9. To export a key:
10. Select the key.
11. Click **Export**.
12. Click **Save**.
13. Navigate to the save location.
14. Click **Save** again.

Exporting a Certificate and Key (CLI Procedure)

To export a certificate and its key, use the following commands at the Privileged EXEC or global Config level of the CLI:

```
export cert
```

```
export cert-key
```

Refer to the *Command Line Interface Reference* for detailed information about these commands.

Exporting a CRL (CLI Procedure)

To export a CRL, use the following command at the Privileged EXEC or global Config level of the CLI:

```
export crl
```

Refer to the *Command Line Interface Reference* for detailed information about these commands.

Exporting a CRL (GUI Procedure)

- Navigate to **ADC >> SSL Management >> Cert Revocation List**.
- Select the CRL. (Click the checkbox next to the CRL name.)
- Click **Export**.
- If the browser security settings normally block downloads, you may need to override the setting. For example, in IE, hold the Ctrl key while clicking Export.
- Click **Save**.
- Navigate to the save location.
- Click **Save** again.

Importing a CA Cert and Private Key

Import a self-signed CA certificate and the certificate's private key (CLI Example)

The following commands import a self-signed CA certificate trusted by the clients, and the certificate's private key:

```
ACOS-Inside(config)# import cert enterpriseABC-selfsignd scp:  
Address or name of remote host []?192.168.1.111  
User name []?admin  
Password []?*****  
File name [/]?enterpriseABC-selfsignd.pem
```



```
ACOS-Inside(config)# import key enterpriseABC-key scp:  
Address or name of remote host []?192.168.1.111  
User name []?admin  
Password []?*****  
File name [/]?enterpriseABC-key.pem
```

Configuring the client-SSL template to enable SSLi (CLI Example)

The following commands configure the client-SSL template to enable SSLi (forward-proxy). It also specifies the certificate and private key that the inside ACOS device uses to dynamically create (and cache) forged server certificates as clients request SSL sessions with external servers.

```
ACOS-Inside(config)# slb template client-ssl SSLInsight_ClientSide  
ACOS-Inside(config-client ssl)# forward-proxy-ca-certificate  
enterpriseABC-selfsignd key enterpriseABC-key  
ACOS-Inside(config-client ssl)# forward-proxy-enable
```

Configuring Forward Proxy Alternate Signing Cert

In the following example, the inside ACOS device is configured with a trusted CA list and an alternate signing key. When a client requests connection to an external SSL server, the inside ACOS device determines whether the certificate of SSL site is signed by a trusted CA. If it is not in the trusted list, the inside ACOS device signs the certificate with the alternate signing key. Because the alternate signing key is not trusted, the client will be warned that the site is insecure.

Forward Proxy Alternate Signing Cert (CLI Example)

1. Import the list of trusted list of CAs:

```
ACOS-Inside(config)# import cert ca-cert enterpriseABC-trusted-CAs scp:  
...
```

2. Import the list of alternate certificate and signing key:

```
ACOS-Inside(config)# import cert alt-cert scp:  
...  
ACOS-Inside(config)# import key alt-key scp:  
...
```

3. Bind the list of trusted CAs and the alternate signing key to the Client SSL template (which in turn is bound to the SSLi virtual port of the inside ACOS device.)

```
ACOS-Inside(config)# slb template client-ssl SSLInsight_ClientSide
ACOS-Inside(config-client ssl)# forward-proxy-ca-certificate
enterpriseABC-selfsignd key enterpriseABC-key
ACOS-Inside(config-client ssl)# forward-proxy-enable
ACOS-Inside(config-client ssl)# forward-proxy-trusted-ca-list
enterpriseABC-trusted-CAs
ACOS-Inside(config-client ssl)# forward-proxy-alt-sign cert
```

Deleting Certificate Files

To delete SSL files, use either of the following methods.

SSL File Delete (GUI Procedure)

1. Navigate to one of the following:
 - **ADC >> SSL Management > SSL Certificates**
 - **ADC >> SSL Management > Cert Revocation List**
2. Select the files to delete.
3. Click **Delete**.

SSL File Delete (CLI Procedure)

Using the CLI, you can delete specific SSL files by name.

Use the `pki delete` command at the global configuration level of the CLI to delete SSL files.

Configuring Simple Certificate Enrollment Protocol (SCEP) Certificates

SCEP is a part of the Public key infrastructure (PKI); it simplifies management of security certificates by providing simplified installation and automated renewal of x.509 certificates. You can use SCEP certificates with the same ACOS features that

support manually imported certificates. For example, SCEP certificates are supported with SSLi.

NOTE: This feature is not supported for HSM platforms, including Thunder 5630.

The following topics are covered:

Key Considerations	267
Enrollment and Renewal Process	267
Configuring SCEP Certificate	268
Copying SCEP Certificate	269
Viewing SCEP Certificate	269
Configuration Example	269

Key Considerations

- SCEP certificate from the CA server goes into the data interface.
- SCEP supports shared and L3V partitions. SCEP traffic follows the routing table rule and DNS setting on the shared partition. Hence, the network interfaces, routing rules, and DNS must be configured correctly in the shared partition, as the SCEP certificate is created in the L3V partition.
- To configure a SCEP certificate, you need to specify the certificate name, a password, and the location (URL) of the ES. ACOS handles the rest.
- To use the certificate, add it to an SSL template and bind the template to the virtual port in your application.
- There is no GUI support for configuring this feature.

Enrollment and Renewal Process

After you configure a SCEP certificate for enrollment, ACOS performs the following steps:

1. Generate a private key. In this step, an RSA key with the specified key length is generated for the certificate.

2. Fetch CA certificates. ACOS queries the ES for its certificates. In this step, three certificates are returned: 1 CA certificate and 2 ES certificates, and ES-encryption certificate and an ES-signature certificate.
3. Generate Certificate Signing Request (CSR). The CSR includes the SCEP password you assign to the SCEP certificate, and other parameters needed for the certificate.
4. Fetch the certificate. The CSR is encrypted using the public key of the ES-encryption certificate, and forwarded to the ES.
5. The ES validates the CSR and forwards the request to the CA. The CA then returns the signed certificate. The certificate is signed using the ES-signature certificate.
6. Store the certificate. After successful verification of the response from the CA, ACOS accepts the certificate and stores it. SCEP certificates are stored in DER format. SCEP keys are stored in PEM format.
7. Schedule renewal. ACOS handles automatic renewal of the certificate when its about to expire. ACOS checks the expiration dates of both the enrolled certificate and the issuing CA's certificate. ACOS then schedules renewal of the certificate, to occur at a specific time or periodically, depending on configuration. ACOS bases the new expiration date on the later of the expiration dates of the enrolled certificate and the CA certificate.
8. Rotate and store files. After certificate renewal, the old certificate and key files are still stored for any future reference. Old files are rotated and the new file replace the existing files.
9. This step ensures that there is no need to change the configuration for applications that use the SCEP certificates, because a valid certificate with the correct name is always stored in the same location. The same applies for private keys as well. ACOS stores up to 4 old certificate and key files for each SCEP certificate.

Configuring SCEP Certificate

To configure SCEP using the CLI:

1. Use the `pki scep-cert` command to create the certificate and change the CLI to edit it.
2. Use the `url` command to specify the location of the ES. The user is the admin

name required by the ES to accept the request.

The user is the admin name required by the ES to accept the request. The host is the ES IP address or hostname. The file is the path and filename for the SCEP process on the ES. Example:

url `http://192.168.230.101/certsrv/mscep/mscep.dll`

3. Specify the password for the certificate. ACOS includes this password in enrollment and renewal requests for the certificate.
4. (Optional) Configure additional parameters.

SCEP certificates have the following default settings:

- Interval – 5 seconds
- Log level – 1
- Maximum poll time – 180 seconds
- Method – GET

The other parameters are not set by default.

5. Use the `enroll` command to begin the enrollment process for the certificate.

Copying SCEP Certificate

You can copy SCEP certificates and keys using the `pki copy-cert` and `pki copy-key` commands.

Refer to the *Command Line Interface Reference* for details.

Viewing SCEP Certificate

To view SCEP information, use the `show pki scep-cert` command.

For more details about SCEP CLI commands, refer to *Command Line Interface Reference*.

Configuration Example

The following commands configure an the inside ACOS device in an SSLi deployment. The wildcard VIP on this device receives SSL-encrypted traffic from inside users, and decrypts the traffic before sending it to the security stack.

The deployment uses a certificate administered by an SCEP ES. Based on the configuration, ACOS automatically renews the certificate on a monthly basis.

For brevity, this example shows only the inside device, where SCEP configuration occurs, and uses only one certificate. The certificate is used both as the root certificate and as a forward-proxy certificate, which uses SNI support.

On the outside device, the only required command related to SSLi is forward-proxy-enable, to enable support for the SSLi feature on the device.

- The following commands enroll the certificate. You need to enroll each certificate only once. After a certificate is enrolled, ACOS uses SCEP to administer the certificate. This includes renewing the certificate before it expires. You do not need to manually administer the certificates after you enroll them.

```
ACOS(config)# pki scep-cert mycert
ACOS(config-scep cert:mycert)# url
http://192.168.230.101/certsrv/mscep/mscep.dll
ACOS(config-scep cert:mycert)# password sample_password
ACOS(config-scep cert:mycert)# renew-every month 1
```

- The following commands configure the client-SSL template:

```
ACOS(config)# slb template client-ssl ssl_int
ACOS(config-client ssl)# forward-proxy-ca-certificate mycert key mycert
ACOS(config-client ssl)# forward-proxy-enable
```

- The following command copies the SCEP certificate and key to a destination certificate file along with the rotation number of the SCEP generated certificate file.

```
ACOS(config)# pki copy-cert mycert rotation 2 testcert
```

- The following shows the configuration the wildcard VIP. This includes configuration of the other resources, in addition to the client-SSL template, that are required by the wildcard VIP: an ACL that matches on the inside clients, the real server configuration, and the service group.

```
access-list 101 permit ip any 10.2.2.0 0.0.0.255 log
!
slb server rs1 10.3.3.1
    health-check-disable
    port 443 tcp
```

```

        health-check-disable
    !
    slb service-group sg1-tcp tcp
        member rs1:443
    !
    slb virtual-server vs1-v4 0.0.0.0 acl 101
        extended-stats
        port 8080 http
        service-group sg1-tcp
        template client-ssl ssl_int
        no-dest-nat port-translation
    !

```

- The following commands show information about the SCEP certificate:

```

ACOS(config)# show pki cert
Name: mycert Type: certificate/key Expiration: Dec 8 22:23:48 2014
GMT [Expired, Bound] SCEP Enrolled

ACOS(config)# show pki scep-cert mycert
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            1d:5b:42:30:00:00:00:00:24:8f
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: DC=com, DC=a10lab, CN=AD03-CA
        Validity
            Not Before: Dec 8 18:23:48 2014 GMT
            Not After : Dec 8 22:23:48 2014 GMT
        Subject: C=CH, O=Linux strongSwan, CN=AX1030
        X509v3 extensions:
            X509v3 Subject Key Identifier:

DA:53:59:9C:EC:52:E3:58:6C:E5:84:11:E7:5C:F4:C9:FC:59:6B:A3
            X509v3 Authority Key Identifier:

keyid:06:18:97:1C:58:B4:E4:95:5F:61:61:5D:DB:9C:1B:85:39:48:87:37

            X509v3 CRL Distribution Points:

```

```

        URI:ldap:///CN=AD03-
CA, CN=AD03, CN=CDP, CN=Public%20Key%20Services, CN=Services, CN=Configurati
on, DC=a10lab,
DC=com?certificateRevocationList?base?objectClass=cRLDistributionPoint

        Authority Information Access:
        CA Issuers - URI:ldap:///CN=AD03-
CA, CN=AIA, CN=Public%20Key%20Services, CN=Services, CN=Configuration, DC=a1
0lab,
DC=com?cACertificate?base?objectClass=certificationAuthority
        OCSP - URI:http://ad03.a10lab.com/ocsp

        X509v3 Key Usage: critical
        Digital Signature, Key Encipherment
        1.3.6.1.4.1.311.21.7:
        0-.%+.....7.....E.....+.....Ks...M.....d...
        X509v3 Extended Key Usage:
        1.3.6.1.5.5.8.2.2
        1.3.6.1.4.1.311.21.10:
        0.0

```

- The following command clears the SCEP certificate logs and debug information.

```

ACOS(config)# clear pki scep-cert log my-cert
ACOS(config)#

```

Configuring Automatic Certificate Management Environment (ACME) Certificates

ACME is used to obtain the certificates for websites (HTTPS). The purpose is to validate domain names for issuing certificates in the web PKI. This protocol is based on passing JSON-formatted messages over HTTPS and was designed by the Internet Security Research Group (ISRG) in RFC 8555 for their Let's Encrypt service. It also enables automating a few aspects of certificate management.

Using both Let's Encrypt and the ACME protocol, you can set up an HTTPS server and automatically obtain a browser-trusted certificate. Generally, ACME client runs as an

agent on a web server. The ACME client is integrated into ACOS to support obtaining and renewing the certificate.

Additionally, the ACME certificate chain (chain-cert) is also supported for secured communication between the client and server. The chain-cert allows clients to trust the server certificates by validating them against a chain of trusted authorities.

After successful ACME certificate enrollment, renewal, or removal, ACOS will automatically fetch or update the ACME full chain certificates.

Supported Features and Limitations

- Only the HTTP-01 method is supported for domain validation challenges.
- Supports aVCS with VRRP-A for high availability.
- [External Account Binding \(EAB\)](#) is supported to bind an external data source.
- Lets Encrypt or ACME feature is not supported in a VRRP-A standalone environment mode.

The following topics are covered:

Key Considerations	273
ACME Directory URL	274
Enrollment and Renewal Process	274
Configuring ACME Certificate	276
Viewing ACME Certificate	277

Key Considerations

- The domain is verified using challenge HTTP-01, which is used to provision an HTTP resource with a well-known URI. Hence, you need to first enable reply to ACME HTTP-01 challenge option for the CA server. This option is allowed only on HTTP port 80.
- HTTP port 80 should not be blocked by the firewall. The virtual ports 80 replying ACME challenges must be up and running.
- The domain must be the Certificate's Common Name and the IP address must be mapped to the ACOS's virtual IP address.

- The CA server determines whether the ACME client has domain control. Hence, you must manually configure reverse proxy on the ACOS side.
- ACME challenge from the CA server goes into the data interface.
- ACME supports shared and L3V partitions. ACME protocol traffic follows the routing table rule and DNS setting on the shared partition. Hence, the network interfaces, routing rules, and DNS must be configured correctly in the shared partition, as the ACME certificate is created in the L3V partition.
- In aVCS with VRRP-A high availability deployment, aVCS master accepts the configuration, and the master syncs the configuration to the slave device(s). Data traffic is served by HA primary device(s), so HTTP-01 type challenge from the CA server is also served by HA primary device(s). If you do not want to utilise the default VRID(0), you can change it with the VRID option.

NOTE:

- Lets Encrypt or ACME feature is not supported in a VRRP-A standalone environment mode.
 - The VIP must be publicly accessible for certificate validation. To configure the VIP IP as the same as the data interface on vThunder, use the `use-if-ip` option.
-

ACME Directory URL

By default, Let's encrypt is used as CA server. Let's Encrypt have rate limits to ensure fair usage by as many people as possible.

- Staging URL - <https://acme-staging-v02.api.letsencrypt.org/directory>
- Production URL - <https://acme-v02.api.letsencrypt.org/directory>

NOTE:

A10 strongly recommend user run with staging environment to test your configurations, then switch to production environment.

Enrollment and Renewal Process

After you configure an ACME certificate for enrollment, ACOS performs the following steps:

- Generates an account key and registers this account with the CA server.
- Generates a domain private key and RSA or EC key with the specified key-length for the certificate.
- ACOS proves the CA server that the user domain is in control. If the domain is already verified, ACOS skips this step. If the domain is not verified yet, ACOS deploys the challenge and triggers the CA server to start the verification process.
 - Certificate Issuance - Once the CA verification succeeds, the ACOS account key pair is authorized, requested, renewed, to create/send CSR and sign them with the authorized key pair. Then, ACOS downloads certificates for the domain from the CA server.
 - Store the certificate. After successful verification of the response from the CA, ACOS accepts the certificate and stores it.
 - Notify application layer - The client-SSL template would accept the ACME certificate. ACOS notifies the application that the certificate is ready for use.
 - Schedule renewal - ACOS handles the automatic renewal of the certificate when it is about to expire. ACOS checks the expiration date, depending on the periodic renewal or a specific time before the certificate expires configurations.
 - Rotate and store files - After certificate renewal, the old certificate and key files are still stored for any future reference. Old files are rotated, and the new file replaces the existing files.

NOTE:

- After enrollment, the account-email and domain are already registered with CA. Hence, account-email and domain cannot be changed after certificate is enrolled. To change these, you must remove acme-cert and re-enroll.
 - When the ACME enrollment or renewal fails, the scheduled renewal time continues as planned, and the next renewal time will be scheduled.
-

You can configure different ACME clients (agent-v1 or agent-v2) to perform the ACME enrollment or renewal. The default agent is agent-v1.

To configure agent-v1 as the ACME client to perform ACME enrollment, use the following command:

```
ACOS(config-acme cert:cert1)# acme-client agent-v1
```

You can also configure the number of concurrent ACME processes. The default and the maximum number of ACME concurrent processes depend on the number of control CPUs. To set the number of ACME concurrent processes, use the following command:

```
ACOS(config)# pki acme-concurrent-process-count <num>
```

If the number of control CPUs is greater than 12, the default and the maximum numbers of ACME concurrent processes are limited to the number of 12 control CPUs. For more information, see *Command Line Interface Reference*.

The formula: Default: (<# of ctrl CPU> * 5) - (<# of ctrl CPU> - 1)

Max: <# of default> + 2

Table 15 : Default and Maximum Numbers of ACME Concurrent Process

No. of control CPUs	Default # of concurrent process	Maximum # of concurrent process
1	5	7
2	9	11
3	13	15
4	17	19
5	21	23
6	25	27
7	29	31
8	33	35
9	37	39
10	41	43
11	45	47
12	49	51
# > 12	49	51

Configuring ACME Certificate

To configure ACME using the CLI:

1. Use the `reply-acme-challenge` option under HTTP virtual port.
2. Use the `pki acme-cert` command to create the certificate and change the CLI to edit it.
3. Use the `account-email`, `cert-type`, `domain`, and `url` command.
4. (Optional) Configure additional parameters.
 - Renew Intervals
 - Log level
 - Staging URL
 - SAN domain
 - VRID
5. Use the `enroll` command to begin the enrollment process for the certificate.

NOTE: Make sure that the HTTP port 80 is not blocked by the firewall. The virtual ports 80 replying ACME challenges must be up and running.

Configuration Example

The following commands enroll the certificate with the Let's encrypt CA. You need to enroll each certificate only once. After a certificate is enrolled, ACOS uses ACME to administer the certificate. This includes renewing the certificate before it expires. You do not need to manually administer the certificates after you enroll them.

```
ACOS(config)# pki acme-cert test
ACOS(config-acme cert:test)# account-email test@url.com
ACOS(config-acme cert:test)# cert-type rsa
ACOS(config-acme cert:test)# domain test.com
ACOS(config-acme cert:test)# enroll
ACOS(config-acme cert:test)# run-with-staging-server
ACOS(config-acme cert:test)# exit
```

Viewing ACME Certificate

To view various certificate statistics, use the `show pki cert` and `show pki cert acme-cert` commands.

- The following command shows the brief details of the certificates including ACME certificate and chain-cert.

```
ACOS(config)# show pki cert
```

Name	Type	Expiration	Status
acme-cert	certificate/key	Mar 20 04:14:27 2024 GMT	[Unexpired, Unbound]
acme_full_chain	certificate	Mar 20 04:14:27 2024 GMT	[Unexpired, Unbound]
.....			

- The following commands show information about the ACME certificate.

You can view both the log and status of the certificate. Also, `show pki acme-cert log <cert-name>` can be used to display the detailed log of the ACME protocol happened during the ACME certificate registration or update process.

```
ACOS(config)# show pki cert acme-cert status
```

Certificate name: test status: SUCCESS

Renew every 2 minutes

rotated files: 4

```
ACOS(config)# show pki cert acme-cert
```

Certificate:

Data:

Version: 3 (0x2)

Serial Number: 1770931951 (0x698e46ef)

Signature Algorithm: sha256WithRSAEncryption

Issuer: CN=aws2020.ddns.net

Validity

Not Before: Oct 15 02:30:26 2019 GMT → issued time

Not After : Oct 14 02:30:26 2021 GMT → cert validity time

Subject: CN=hku_ra, O=MyOrganization, C=SE

- The following command shows the detailed information of the ACME chain-cert.

```

ACOS(config)# show pki cert acme_full_chain
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number:
      04:f1:e7:d5:f3:23:f8:de:c2:21:12:54:ea:4f:1f:d9:aa:56
    Signature Algorithm: sha256WithRSAEncryption
    Issuer: C=US, O=Let's Encrypt, CN=R3
    Validity
      Not Before: Dec 21 04:14:28 2023 GMT
      Not After : Mar 20 04:14:27 2024 GMT
    Subject: CN=acme_test.ddns.net
    X509v3 extensions:
      X509v3 Key Usage: critical
        Digital Signature, Key Encipherment
      X509v3 Extended Key Usage:
.....
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number:
      91:2b:08:4a:cf:0c:18:a7:53:f6:d6:2e:25:a7:5f:5a
    Signature Algorithm: sha256WithRSAEncryption
    Issuer: C=US, O=Internet Security Research Group, CN=ISRG Root
X1
    Validity
      Not Before: Sep  4 00:00:00 2020 GMT
      Not After : Sep 15 16:00:00 2025 GMT
    Subject: C=US, O=Let's Encrypt, CN=R3
    X509v3 extensions:
      X509v3 Key Usage: critical
        Digital Signature, Certificate Sign, CRL Sign
      X509v3 Extended Key Usage:
.....

```

```

Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number:
      40:01:77:21:37:d4:e9:42:b8:ee:76:aa:3c:64:0a:b7
    Signature Algorithm: sha256WithRSAEncryption
    Issuer: O=Digital Signature Trust Co., CN=DST Root CA X3
    Validity
      Not Before: Jan 20 19:14:03 2021 GMT
      Not After : Sep 30 18:14:03 2024 GMT
    Subject: C=US, O=Internet Security Research Group, CN=ISRG Root
X1
    X509v3 extensions:
      X509v3 Basic Constraints: critical
        CA:TRUE
      X509v3 Key Usage: critical
        Certificate Sign, CRL Sign
      Authority Information Access:
        CA Issuers -
URI:http://apps.identrust.com/roots/dstrootcax3.p7c
.....

```

- The following command clears the ACME certificate logs and debug information.

```

ACOS(config)# clear pki acme-cert log acme-cert
ACOS(config)#

```


OCSP Certificate Management

The following topics are covered:

Overview	281
ACOS Server Certificate Verification	282
Certificate Revocation List	288
IP-less OCSP and CRL Requests	290
Invalid Certificates Customizable Message	292
Revoking Certificates	293

Overview

The Online Certificate Status Protocol (OCSP) is an IETF protocol that SSL clients, such as ACOS SSL, can use to verify the state of a server's certificate before enabling an SSL session with that server. The Transport Layer Security Protocol (TLS) also provides SSL servers the option to staple their OCSP current status information to their SSL/TLS handshake.

In ACOS SSLi, ACOS_decrypt uses its own certificate and private key to proxy certificates from the outside server when acting as an SSL proxy. Without OCSP, ACOS cannot check whether the certificate of the outside server has become invalid before the expiration date indicated by the Certificate Authority (CA). The ACOS Server Certificate Verification for SSLi feature uses OCSP to dynamically verify the server certification status, whether it is valid or expired.

The ACOS software verifies the current state of the server certification before proxying the session certificates used in SSL proxy connections -- whether or not the CA expiration date has been reached.

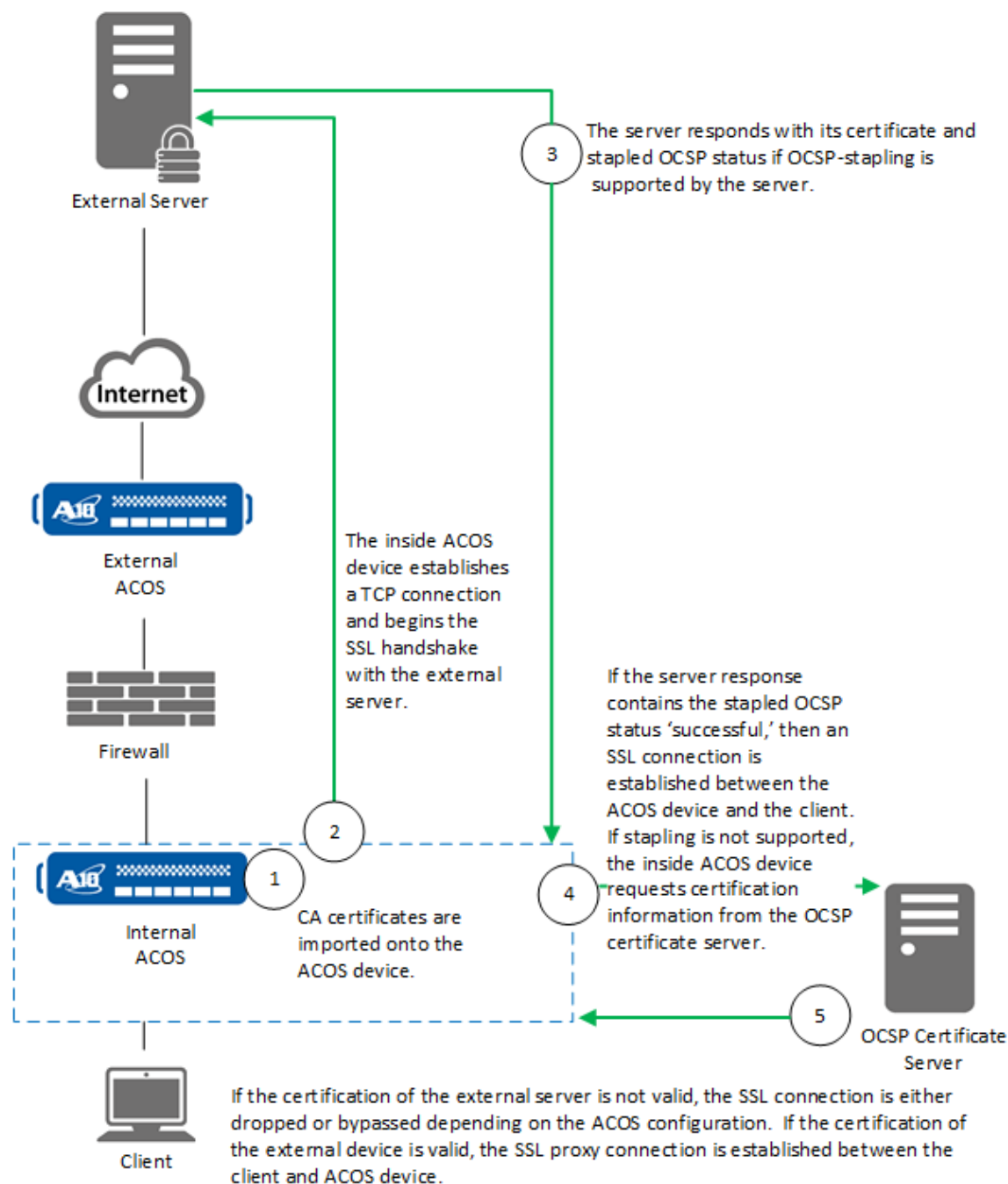
ACOS does not support OCSP verification for HTTPS responder URIs in certificate extensions. OCSP-stapling configuration is not applicable to SSLi. The internal SSLi receives and processes the stapled responses.

After a TCP connection has been established between the ACOS device and the client, the server certificate verification process begins.

ACOS Server Certificate Verification

By default, ACOS server certificate verification is enabled. The `forward-proxy-ocsp-disable` command disables OCSP verification. This feature applies to transparent SSLi for HTTPS sessions.

Figure 25 : ACOS Server Certificate Verification Process



The following is the workflow:

1. ACOS_decrypt is configured with imported trusted CA certificates that it uses to verify the outside server's certificates. The CA certificates are imported prior to the beginning of the message exchange process.
2. A client initiates an SSL connection to a website which is proxied or intercepted by ACOS_decrypt. Assuming that ACOS has not already cached a proxied certificate that it can use to create the requested SSL session, it opens an SSL session with the same outside server that the client is attempting to reach.
3. If the outside server has enabled OCSP stapling, the server responds with a "Certificate Status" SSL/TLS handshake message that tells the ACOS device whether or not the server certificate is valid and the expiration date of that certificate if it is valid.
 - If the "Certificate Status" response contains a "good" stapled OCSP status, the certificate is valid and ACOS_decrypt uses its private key to proxy a public certificate, which it sends to the client. Assuming the client accepts the proxied certificate, an SSL session begins and SSL traffic (for SSLi or SSL offload) is forwarded either to the inspection devices (in SSLi scenarios) or to the outside server (in SSL offload scenarios).
 - If the server response contains a "revoked" staple OCSP status, the certificate is not valid, and depending on the ACOS configuration, ACOS either drops the connection or bypasses SSL proxy to allow the client to connect directly to the outside server.
 - If the server does not support OCSP stapling, the process continues with step 4.
4. ACOS_decrypt looks up the location of the OCSP server embedded within the AIA (Authority Information Access) field in the certificate sent by the Internet Server. An OCSP request is sent to the OCSP URL within the AIA field in each certificate inside the chain, for which ACOS_decrypt does not already have an OCSP cache entry. If the OCSP URL is an HTTP URL, an HTTP connection is initiated to that OCSP responder. If the OCSP URL is an HTTPS URL, the ACOS device will not continue with OCSP verification for that certificate/certificate chain.
5. If the OCSP server responds that the certificate is valid, ACOS_decrypt caches the certificate validity information with its expiration time expressed in seconds. If this OCSP entry expires while a proxied certificate corresponding to it is still in the cache, then that proxied certificate is also aged out. When a new client

request comes to the ACOS device for the same website, the OCSP verification and certificate proxying process repeats again.

6. If the OCSP server responds that the certificate is not valid then depending on the ACOS device configuration, ACOS either drops the connection or bypasses SSL proxy to allow the client to connect directly to the outside server.

The following are some guidelines for the process:

- When ACOS bypasses SSL traffic, it does not proxy the server certificate. It forwards the Server Hello, Certificate, and other SSL handshake messages received from the outside server in response to the client hello message, onto the client. The only changes made to these packets would be at Layer 2, Layer 3, or Layer 4 as applicable for traffic forwarding.
- ACOS considers “revoked” or “unauthorized” responses from the OCSP responder as “not successful”. If the OCSP server/responder is not reachable (connection time out), or responds with a different status code or with a “tryLater” or “status unknown” message, then the client connections corresponding to these certificates are bypassed.

CLI Configuration

1. Configure the SSL client template.

The following SSL client template is enabled for SSL proxy through the following forward-proxy commands.

```
ACOS_decrypt(config)#slb template client-ssl SSLInsight_ClientSide
ACOS_decrypt(config-client ssl)# forward-proxy-ca-certificate
Cert123.pem key key123
ACOS_decrypt(config-client ssl)# forward-proxy-enable
ACOS_decrypt(config-client ssl)# forward-proxy-trusted-ca default_ca_
bundle_jan_2018
ACOS_decrypt(config-client ssl)# forward-proxy-trusted-ca windows_ca_
bundle_jan_2018
ACOS_decrypt(config-client ssl)# enable-tls-alert-logging fatal
ACOS_decrypt(config-client ssl)# forward-proxy-verify-cert-fail-action
drop
ACOS_decrypt(config-client ssl)# forward-proxy-cert-revoke-action drop
ACOS_decrypt(config-client ssl)# forward-proxy-cert-unknown-action drop
```

By default, ACOS drops connections to clients in which the certification of the outside server is invalid. When server verification is configured using the `forward-proxy-trusted-ca` commands in a client-SSL template, the action is to bypass client connections if the certification of the outside server is invalid.

2. If you deploy SSLi and ACOS_decrypt is not provisioned with L3V partitions. the configuration of `port 443 https` of the wildcard VIP on the client is not changed.

```
ACOS_decrypt(config)#slb virtual-server decrypt_VIP 0.0.0.0 acl 100
ACOS_decrypt(config-slbg vserver)#port 443 https
ACOS_decrypt(config-slbg vserver-vport)#no-dest-nat port-translation
ACOS_decrypt(config-slbg vserver-vport)#service-group FW1_Inspect_SG
ACOS_decrypt(config-slbg vserver-vport)#template client-ssl SSLInsight_ClientSide
ACOS_decrypt(config-slbg vserver-vport)#exit
```

3. If you deploy SSLi and ACOS_decrypt is provisioned with L3V partitions, the configuration of `port 443 https` of the wildcard VIP must include the route to the DNS server as shown in the following command lines, and non-HTTP protocols must be bypassed:

```
ACOS_decrypt(config)#slb template dynamic-service DNS-FOR-OCSP
ACOS_decrypt(config-dynamic-service)#dns server 192.168.1.110
ACOS_decrypt(config-dynamic-service)#dns server 8.8.8.8
ACOS_decrypt(config-dynamic-service)#exit
```

The command creates an HTTP template named “non-http-bypass.” When this template is bound to the HTTPS port, it redirects all non-HTTP traffic to the `FW1_Inspect_SG` service group. By default, the ACOS device will drop non-HTTP requests that are sent to an HTTP port.

```
ACOS_decrypt(config)# slb template http non-http-bypass
ACOS_decrypt(config-http)# non-http-bypass service-group FW1_Inspect_SG
ACOS_decrypt(config-http)# exit
```

4. Bind both templates, `non-http-bypass` and `d1`, and the client-SSL template to the virtual server that proxies for the SSL external server.

```
ACOS_decrypt(config)# slb virtual-server decrypt_VIP 0.0.0.0 acl 100
ACOS_decrypt(config-slbg vserver)# port 443 https
ACOS_decrypt(config-slbg vserver-vport)# no-dest-nat port-translation
ACOS_decrypt(config-slbg vserver-vport)# service-group FW1_Inspect_SG
```

```

ACOS_decrypt(config-slb vserver-vport) # use-rcv-hop-for-resp
ACOS_decrypt(config-slb vserver-vport) # template dynamic-service d1
ACOS_decrypt(config-slb vserver-vport) # template http non-http-bypass
ACOS_decrypt(config-slb vserver-vport) # template client-ssl SSLInsight_
ClientSide
ACOS_decrypt(config-slb vserver-vport) # exit

```

- Whether or not ACOS_decrypt is L3V partitioned, the configuration of the wildcard ports of the VIP is not changed:

```

ACOS_decrypt(config-slb vserver) #port 0 tcp
ACOS_decrypt(config-slb vserver-vport) #no-dest-nat
ACOS_decrypt(config-slb vserver-vport) #service-group ALL_TCP_SG
ACOS_decrypt(config-slb vserver-vport) #exit

ACOS_decrypt(config-slb vserver) #port 0 udp
ACOS_decrypt(config-slb vserver-vport) #no-dest-nat
ACOS_decrypt(config-slb vserver-vport) #service-group ALL_UDP_SG
ACOS_decrypt(config-slb vserver-vport) #exit

ACOS_decrypt(config-slb vserver) #port 0 others
ACOS_decrypt(config-slb vserver-vport) #no-dest-nat
ACOS_decrypt(config-slb vserver-vport) #service-group ALL_UDP_SG
ACOS_decrypt(config-slb vserver-vport) #exit
ACOS_decrypt(config-slb vserver) #exit

```

- Enable source-NAT pool for use by the ACOS Server Verification Module (SVM) daemon. Source-NAT is required to dynamically make the TCP connections between ACOS devices and the resources that SVM OCSP needs to reach. In the following example, the TCP connection uses a pool of source addresses reserved for OCSP connections.

```

ACOS_decrypt(config) #ip nat pool OCSP_NAT_v1_50 192.168.51.254
192.168.51.254 netmask /24
ACOS_decrypt(config) #slb svm-source-nat pool OCSP_NAT_v1_50

```

- Configure the IP address of a DNS server that ACOS_decrypt can reach to be able to look up the IP address of the OCSP servers that the ACOS server certificate verification feature will use. The configuration of a default route, interfaces, ports, and service groups that enable ACOS_decrypt to connect to the DNS server

are not shown.

```
ACOS_decrypt(config)#ip dns primary 8.8.8.8
```

8. Use the `show slb ssl-ocsp cache` command to view the status of the OSCP cache:

```
ACOS_decrypt#show slb ssl-ocsp cache
Total: 2
Common Name                                     Status
-----
Company1 Internet Authority G2                  Good
Company2 Root Certificate Authority - G2        Good
```

Certificate Revocation List

Certificate Revocation List (CRL) is an available option for the server-SSL template to validate the service-side server. Each CRL must have a relevant certificate authority (CA) certificate configured in the same SSL template in order to validate whether incoming certificates have been revoked. A maximum of 128 files containing CA or CRL may be configured.

Specify the name of the Certificate Revocation List (CRL) to use for verifying whether server certificates have been revoked. The CRL must be installed on the ACOS device first. The CA certificate relevant to the CRL must also be specified.

CLI Configuration

When you add a CRL to a server-SSL template, the ACOS device checks the CRL to confirm whether or not the servers' certificates have been revoked or not by the issuing Certificate Authority (CA).

This section provides configuration instructions for adding CRL and CA certificates, viewing the CRL and OCSP activity, and retrieving the CRL expiration status.

1. Add CRL and CA certificates to a server-SSL template named, `ssl-svr` along with the import of CA certificates. The CRL section is highlighted for clarity.

```
ACOS(config-server ssl)#slb template server-ssl SSL-Svr
ACOS(config-server ssl)#  crl 10_ca.crt_crl.pem
ACOS(config-server ssl)#  crl 20_ca.crt_crl.pem
```



```
ACOS(config-server ssl)# crl root-ca.pem.crl.pem
ACOS(config-server ssl)#ca-cert 10_ca.crt
ACOS(config-server ssl)# certificate Cert123 key Key123 pass-phrase Pass123
ACOS(config-server ssl)#ca-cert 20_ca.crt
ACOS(config-server ssl)#ca-cert root-ca.pem
```

2. Use the `show slb ssl-cert-revoke-stats` command to view both OSCP and CRL activity:

```
ACOS(config-client ssl)# show slb ssl-cert-revoke-stats
OCSP stapling response good:          0
Certificate chain status good:         0
Certificate chain status revoked:      0
Certificate chain status unknown:      0
OCSP requests:                        0
OCSP responses:                       0
OCSP connection errors:               0
OCSP URI not found:                   0
OCSP URI https:                       0
OCSP URI unsupported:                 0
OCSP response status good:            0
OCSP response status revoked:         0
OCSP response status unknown:         0
OCSP cache status good:               0
OCSP cache status revoked:            0
OCSP cache miss:                      0
OCSP cache expired:                   0
OCSP other errors:                    0
CRL requests:                         0
CRL responses:                        0
CRL connection errors:                0
CRL URI not found:                    0
CRL URI https:                        0
CRL URI unsupported:                  0
CRL response status good:             0
CRL response status revoked:          0
CRL response status unknown:          0
CRL cache status good:                0
CRL cache status revoked:             0
CRL other errors:                     0
```

3. Use the `show slb ssl-crl` command to view the retrieved CRL status for a specific virtual port. If the certificate issuers have listed expiration dates for the certificates, then this command will show you the issuer and the expired or not expired status.

```
ACOS_decrypt#show slb ssl-crl example_vip_name 443
Virtual server(example_vip_name : 443):
----Retrieved CRL----
Issuer: /O=AlphaSSL/CN=AlphaSSL CA - G2
Status: Expired
Issuer: /O=Cybertrust, Inc/CN=Cybertrust Global Root
Status: Not expired
Issuer: /O=Verizon Cybertrust Security/CN=Cybertrust SureServer EV OSCP
CA
Status: Not expired
Issuer: /O=Digital Signature Trust Co./CN=DST Root CA X3
Status: Expired
Issuer: /C=BE/O=GlobalSign nv-sa/CN=AlphaSSL CA - SHA256 - G2
Status: Expired
```

4. You can disable CRL services for SSLi (forward-proxy) with the `forward-proxy-crl-disable` command. The following example shows how to disable CRL services in the client-SSL template named `clientSide_vRouter`.

```
ACOS_decrypt(config)#slb template client-ssl ClientSide_vRouter
ACOS_decrypt(config-client ssl)#forward-proxy-crl-disable
```

IP-less OCSP and CRL Requests

SVM NAT pool is configured to fetch OCSP and CRL requests for normal SSLi setups. However, ACOS also supports using the client IP address to fetch OCSP and CRL requests. This enables the ACOS deployment to be used across different hardware systems as there is no requirement to configure an IP address for OCSP and CRL requests.

Some of the important guidelines are:

- This feature is supported for IP-less Layer-2 SSLi.
- This feature is only applicable for static and dynamic SSLi. The SSLi virtual port does not support this feature.

- In order to resolve the OCSP and CRL URLs, the `ip dns primary` configuration in the shared partition must be set. The `ip dns primary` configuration is required in the shared partition if the ACOS encrypt and ACOS decrypt zones are in private partitions as it is a global configuration.
- The route for `ip dns primary` must also be configured as the default gateway of the management IP.
- Unlike legacy SSLi, the feature does not need to configure `svm-source-nat pool` and `dynamic-service` template on the shared and L3V partitions respectively.
- Instead of `svm-source-nat pool` IP, use the client IP address for sending OCSP and CRL requests.

CLI Configuration

The following is a sample configuration of the shared partition of the ACOS system. The code in blue highlight is with reference to the afore-mentioned configuration guidelines.

```
ACOS# show running-config
!
! multi-ctrl-cpu 2
!The IP address used here is also used as the default gateway.
ip dns primary 192.168.1.50
!
partition test id 21
!
interface management
  ip address 10.6.29.50 255.255.255.0
  ip default-gateway 10.6.29.1
!
interface ethernet 1
!
interface ethernet 2
!
interface ethernet 3
!
interface ethernet 4
!
ip route 192.168.1.50 /32 10.6.29.1
!
```

```
end
```

Invalid Certificates Customizable Message

An invalid certificate is defined as an origin certificate that has issues, such as CN mismatch, self-signed, unknown CA, revoked certificate, expired certificate, broken trust chain, OCSP issues, and so on. When certificate validation fails or OCSP validation fails, the SSLi log includes a unique ID that is referenced by the customizable web page displayed to the user.

In case of a certificate verification failure, a certificate revocation, or an unknown certificate, SSLi enables you to either drop, bypass, or continue the connection. In addition to the three actions, you can also use the **block** option that enables you to display an error page with a customizable message.

CLI Configuration

You can configure a customizable message if you enable the **block** option for SSLi certificate errors.

Perform the following steps to configure a customizable error message:

1. Create the client SSL template.

```
ACOS(config)# slb template client-ssl clientssl
```

2. Configure the block option for a certificate verification failure.

```
ACOS(config-client ssl)# forward-proxy-verify-cert-fail-action block
```

3. Configure the block option for an unknown certificate failure.

```
ACOS(config-client ssl)# forward-proxy-cert-unknown-action block
```

4. Configure the block options for a certificate revocation.

```
ACOS(config-client ssl)# forward-proxy-cert-revoke-action block
```

5. Configure the message to display if SSLi encounters an invalid certificate.

```
ACOS(config-client ssl)# forward-proxy-block-message "This website  
cannot be displayed as there is a certificate issue."
```

Revoking Certificates

ACOS supports revoking certificates generated by SSLi if the certificates are leaked. Revoked certificates are identified by their serial numbers. If a certificate is revoked from the cache, a CRL is generated and provided to the clients connected to SSLi providing information about the revoked certificates.

The following is some important information regarding revoked certificates:

- A certificate, if revoked, cannot be restored.
- When the CRL is generated, the list is read, put into CRL format, and signed by using the forward-proxy-ca-certificate.
- The CRL is generated manually and then exported to a location reachable by the clients.
- The feature is supported both in ACOS GUI and ACOS CLI.

The following topics are covered:

[CLI Configuration](#) 293

CLI Configuration

Step 1: Checking the Certificate Serial Number

Follow the steps below to obtain the server certificate serial number, depending on the type of SSLi configured for your system.

Static Port SSLi

The command syntax for checking the certificate serial number for static SSLi vport is:

```
ACOS(config)# show slb ssl-forward-proxy-cert vip_name vport_number  
ipaddress server_ip_address server_name
```

For static port SSLi, the following is an example:

```
ACOS(config)# show slb ssl-forward-proxy-cert internet 443 ipaddress  
10.10.10.1 www.example.com
```

Output similar to the following is displayed, the certificate serial number is in blue highlight:

```
Virtual server port inside: 443

----Start One Certificate---
real server: 10.10.10.1:443 tcp
server name: example.com
ALPN protocol: ALPN HTTP2
state: ready
hit times: 1
idle time: 26 seconds
timeout after 3574 seconds
expires after 604764 seconds
version: 3
subject: SSL Certificate
common name: example.com
division:
locality: Sunnyvale
state or province: California
country name: US
subject alternative name:
email:
start time: May  2 00:00:00 2023 GMT
expire time: Oct 25 23:59:59 2023 GMT
issuer: A10 Networks
serial(hex): a79c0000000000006aa
----End One Certificate---

Total number of particular certificates that are printed is 1
```

Dynamic Port SSLi

The command syntax for checking the certificate serial number for dynamic port SSLi is:

```
ACOS(config)# show slb ssl-forward-proxy-cert vip_name 0 ip server_ip_
address port_number server_name
```

The port number is the port on which traffic is running. For static port SSLi, the following is an example:

```
ACOS(config)# show slb ssl-forward-proxy-cert inside 0 ip 10.10.10.1 443  
www.example.com
```

Output similar to the following is displayed, the certificate serial number is in blue highlight:

```
Virtual server port inside: 443  
  
----Start One Certificate---  
real server: 10.10.10.1:443 tcp  
server name: example.com  
ALPN protocol: ALPN HTTP2  
state: ready  
hit times: 1  
idle time: 26 seconds  
timeout after 3574 seconds  
expires after 604764 seconds  
version: 3  
subject: SSL Certificate  
common name: example.com  
division:  
locality: Sunnyvale  
state or province: California  
country name: US  
subject alternative name:  
email:  
start time: May  2 00:00:00 2023 GMT  
expire time: Oct 25 23:59:59 2023 GMT  
issuer: A10 Networks  
serial(hex): s76c00000000000006ss  
----End One Certificate---  
  
Total number of particular certificates that are printed is 1
```

Step 2: Revoking a Certificate

The following is the syntax for revoking a certificate:

```
ACOS(config)# pki ssli revoke vip_name vport_number certificate_serial_  
number_hex
```

For a static port SSLi configuration where the VIP is called `internet` and the certificate serial number is `0123e2`, run the following command to revoke the certificate:

```
ACOS(config)# pki ssli revoke internet 443 0123e2
```

Step 3: Generating a CRL

The following is the syntax for generating a CRL:

```
ACOS(config)# pki ssli generate crl vip_name vport_number
```

Run the following command to generate the CRL for a static port SSLi configuration:

```
ACOS(config)# pki ssli generate crl internet 443
```

Step 4: Displaying the CRL

The following is the syntax for displaying the generated CRL:

```
ACOS(config)# show pki crl
```

The following is a sample output:

Output similar to the following is displayed:

```
name: internet-443.crl  
Issuer: /O=Example Inc, Inc./OU=IT SSLi/emailAddress=it@example.com/L=San  
Jose/ST=CA/C=US/CN=A10_Intermediate_CA_SHA256
```

Step 5: Clearing Revoked Certificates and Deleting the CRL

The following is the syntax for clearing the list of revoked certificates and deleting the CRL:

```
ACOS(config)# clear slb ssl-forward-proxy-revoked vip-name vport_number
```

The following is an example:

```
ACOS(config)# clear slb ssl-forward-proxy-revoked internet 443
```


Explicit and Transparent Proxy

The following topics are covered:

Overview	298
Deployment Example	298
CLI Configuration	299
Proxy Chaining SSLi	308
Explicit Proxy with Dual-Stack Forwarding Support	312
Drop and Drop-Redirect-URL Message Responses	321
Virtual Wire with SSLi Deployment	327
AAM for SSLi Transparent Proxy	353
Transparent Proxy with Normal VIPs Associated to ACL	359
Non-standard Port Support	366
Terminating SSLi Session Silently	367

Overview

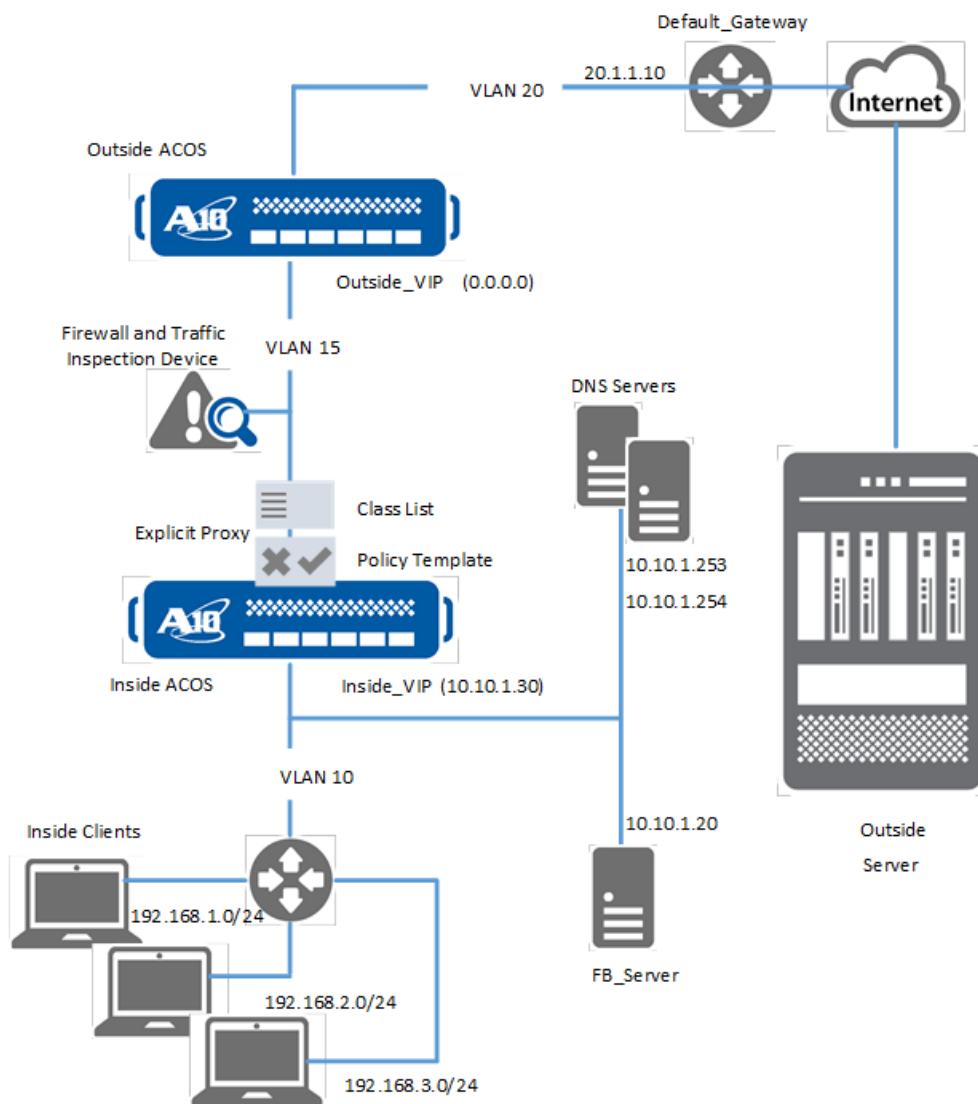
A proxy is an agent that acts in place of the original requester. For a transparent proxy, the client is not aware of the use of a proxy (proxy server). In the case of an explicit proxy, client browsers are configured to send requests to a proxy server, hence the name explicit proxy as the proxy service is known.

In HTTP proxy, browser clients connect to the Internet through proxy servers that make service requests on behalf of the clients. The configuration of the browser specifies the proxy servers it uses. You can configure ACOS to provide both SSLi services and HTTP proxy services in the same HTTP session, and on the same virtual router.

Deployment Example

[Figure 26](#) shows the topology of this SSLi example to which explicit HTTP proxy services are added. To understand the SSLi topology, refer to [Two ACOS Devices, Each With Single Partition Deployment](#).

Figure 26 : Explicit Proxy with Basic Static-Port SSLi Example



CLI Configuration

This section describes how to add an explicit HTTP proxy to an SSLi solution consisting of two ACOS devices, ACOS_decrypt and ACOS_encrypt. Both SSLi and explicit proxy are configured on the same virtual port.

The following topics are covered:

ACOS_decrypt Configuration	300
ACOS_encrypt Configuration	302
Verifying the Configuration	302
Consolidated Configuration	304

ACOS_decrypt Configuration

The following are the recommended steps for configuring explicit proxy on ACOS_decrypt.

Prior to configuring explicit proxy, determine what port number and what IP address are to be used for explicit proxy. It is this address that the clients will configure in their browser's proxy option. In example, 10.10.1.30:1234 will be used.

1. Create the source-NAT pool of IP addresses required by the `forward-to-internet` action.

The configuration of the NAT pool used by source-NAT for Internet-bound traffic provides a source address that is the same as the IP interface of ACOS_decrypt.

```
ip nat pool Internet_Pool 10.10.1.30 10.10.1.30 netmask /32
```

2. Enter the following commands to define the template for the explicit proxy policy.

The policy template defines what actions are applied to upstream traffic by the client-facing virtual server on the ACOS_decrypt device. The configuration of this policy template follows:

```
slb template policy Explicit_Proxy
forward-policy
action Permit_to_Internet
    forward-to-internet FW1_Inspect_SG snat Internet_Pool
log
source Any_Source
match-any
destination any action Permit_to_Internet
```

3. Enter the following commands to create a template that is bound to the client-facing virtual server to provide the IP addresses of DNS servers used by the VIP. The dynamic service template can contain two default DNS servers and / or class-

lists with two DNS servers each. You can enable ACOS_decrypt to look up the IP address of the EnterpriseABC servers that the clients request SSL connections to.

```
slb template dynamic-service DNS
  dns server 10.10.1.253
  dns server 10.10.1.254
  class-list cl2 priority 30
  dns server 10.20.2.2
  dns server 10.20.2.3
```

4. Configure a static route to a gateway, 10.10.1.2, that can reach the clients on the 192.168.1.0 /24 subnet. No route to the DNS servers is necessary because ACOS_decrypt and the DNS servers are both on the same subnet, 10.10.1.0 /24.

```
ip route 192.168.1.0 /24 10.10.1.2
!
```

5. Modify the configuration of the decrypt_VIP to enable explicit proxy. The decrypt_VIP is a static-port virtual router that manages explicit proxy traffic and provides SSLi services. The policy template, the SSL client template, and the dynamic services template are all bound to the client-facing virtual router on ACOS_decrypt.
6. Specify the IP address of the decrypt_VIP as 10.10.1.30. The IP address must be explicit and matches the proxy configurations of the clients.
7. Begin the configuration of virtual port 1234 on 10.10.1.30 as the interface of this VIP. This too matches the proxy configuration on the clients.
8. Bind the Explicit_Proxy policy template to the 1234 HTTP port of the VIP.
9. Bind the DNS dynamic services template to the 1234 HTTP port of the VIP.
10. Bind the SSLInsight_decrypt template to the 1234 HTTP port of the VIP.

```
slb virtual-server decrypt_VIP 10.10.1.30
  port 1234 http
  service-group FW1_Inspect_SG
  template client-ssl SSLInsight_decrypt
  template policy Explicit_Proxy
  template dynamic-service DNS
  no-dest-nat port-translation
```

ACOS_encrypt Configuration

The only change is the addition of a default route to the gateway router to Internet.

```
ip route 0.0.0.0 /0 20.1.1.10
```

Verifying the Configuration

Enter the following commands to verify the configuration and operation of this explicit proxy example:

1. Show the configuration of the SLB policy template.

```
ACOS_decrypt# show slb template policy Explicit_Proxy
slb template policy Explicit_Proxy
  forward-policy
    action Permit_to_Internet
      forward-to-internet SSL snat Internet_Pool fallback SSL snat
Fallback_Pool
    log
    source Any_Source
    match-any
    destination any action Permit_to_Internet
```

2. Show the IP addresses of the source-NAT pool.

```
ACOS_decrypt# show ip nat pool
Pool Name          Start Address      End Address        Mask  Gateway
  Vrid
-----
Internet_Pool      203.0.113.5        203.0.113.5        /32   0.0.0.0
  default
```

3. Show the status of the client-facing VIP on ACOS_decrypt.

```
ACOS_decrypt# show slb virtual-server decrypt_VIP
Virtual server: EP_VIP State: Functional Up    IP: 10.10.1.30
  Port                               Curr-conn  Total-conn  Rsv-Pkt    Fwd-
Pkt   Peak-conn
```

```
-----
Virtual Port:8080 / service:To_Internet / state:Functional Up
    port 8080  http                0                0                0                0
    0
```

4. Show the detailed status of the client-facing VIP on ACOS_decrypt.

```
ACOS_decrypt# show slb virtual-server decrypt_VIP detail
Virtual server name:                decrypt_VIP
Virtual server IP address:          10.10.1.30
Virtual server MAC:                 001f:a003:5fc3
Virtual server template:            default
Current connection:                 0
Current request:                    0
Total connection:                   0
Total request:                      0
Total request success:              0
Total forward bytes:                0
Total forward packets:              0
Total reverse bytes:                0
Total reverse packets:              0
Peak connections:                   0
Current connection rate:            0 per second
```

5. Show the statistics of the forward-policy to verify the forward-policy managed packet flow through the ACOS_decrypt virtual router.

```
ACOS_decrypt# show slb template policy Explicit_Proxy forward-policy-
stats
slb template policy name: Explicit_Proxy
Source NAT failure:                 0
Unresolved DNS requests:            0
Outstanding DNS requests:           0
Hits:                               0
Requests forward to Internet:        0
Requests forward to Service Group:   0
Requests dropped:                    0
Source Match not found:              0
Expected Client HELLO requests not found: 0
```

Consolidated Configuration

The configuration of ACOS_decrypt is shown first: The **highlighted** lines of the configuration show items specifically described in the preceding configuration instructions.

```
ACOS_decrypt#show running-config
!
access-list 100 permit ip any any vlan 10
!
!
ip nat pool Internet_Pool 10.10.1.30 10.10.1.30 netmask /32
!
ip route 192.168.1.0 /24 10.10.1.2
!
vlan 10
    tagged ethernet 1
    router-interface ve 10
!
vlan 15
    tagged ethernet 1
    router-interface ve 15
!
hostname ACOS_decrypt
!
interface ethernet 1
    enable
!
interface ve 10
    ip address 10.10.1.2 255.255.255.0
    ip allow-promiscuous-vip
!
interface ve 15
    ip address 10.15.1.2 255.255.255.0
!
!
slb template dynamic-service DNS
    dns server 10.10.1.253
    dns server 10.10.1.254
    class-list c12 priority 30
```



```
    dns server 10.20.2.2
    dns server 10.20.2.3
!
slb server FW1_Inspect 10.15.1.12
    port 8080 tcp
        health-check-disable
    port 0 tcp
        health-check-disable
    port 0 udp
        health-check-disable
!
slb service-group ALL_TCP_SG tcp
    member FW1_Inspect 0
!
slb service-group ALL_UDP_SG udp
    member FW1_Inspect 0
!
slb service-group FW1_Inspect_SG tcp
    member FW1_Inspect 8080
!
slb template policy Explicit_Proxy
    forward-policy
        action Permit_to_Internet
            forward-to-internet FW1_Inspect_SG snat Internet_Pool
            log
        source Any_Source
            match-any
            destination any action Permit_to_Internet
!
slb template client-ssl SSLInsight_decrypt
    forward-proxy-ca-certificate Cert123.pem key key123
    forward-proxy-enable
!
slb virtual-server decrypt_VIP 10.10.1.30
    port 1234 http
        service-group FW1_Inspect_SG
        template client-ssl SSLInsight_decrypt
        template policy Explicit_Proxy
        template dynamic-service DNS
        no-dest-nat port-translation
```

```
port 0 tcp
  service-group ALL_TCP_SG
  no-dest-nat
port 0 udp
  service-group ALL_UDP_SG
  no-dest-nat
port 0 others
  service-group ALL_UDP_SG
  no-dest-nat
!
end
```

Use the `show running-config` command to check your configuration of ACOS_encrypt. A default route to the Internet gateway is added; otherwise explicit proxy configuration does not change the configuration. The highlighted lines of the configuration show items specifically described in the preceding configuration instructions.

```
ACOS_encrypt# show running-config
!
access-list 101 permit ip any any vlan 15
!
vlan 20
  tagged ethernet 1
  router-interface ve 20
!
vlan 15
  tagged ethernet 1
  router-interface ve 15
!
ip route 0.0.0.0 /0 20.1.1.10
!
hostname ACOS_encrypt
!
interface ethernet 1
  enable
!
interface ve 20
  ip address 20.1.1.2 255.255.255.0
```

```
!  
interface ve 15  
  ip address 10.15.1.12 255.255.255.0  
  ip allow-promiscuous-vip  
!  
slb server Default_Gateway 20.1.1.10  
  port 443 tcp  
    health-check-disable  
  port 0 tcp  
    health-check-disable  
  port 0 udp  
    health-check-disable  
!  
slb service-group DG_SSL_SG tcp  
  member Default_Gateway 443  
  
slb service-group DG_TCP_SG tcp  
  member Default_Gateway 0  
  
slb service-group DG_UDP_SG udp  
  member Default_Gateway 0  
!  
slb template server-ssl SSLInsight_encrypt  
  forward-proxy-enable  
!  
slb virtual-server decrypt_VIP 0.0.0.0 acl 101  
  port 8080 http  
    no-dest-nat port-translation  
    service-group DG_SSL_SG  
    template server-ssl SSLInsight_decrypt  
    use-rcv-hop-for-resp  
  port 0 tcp  
    no-dest-nat  
    service-group DG_TCP_SG  
    use-rcv-hop-for-resp  
  port 0 udp  
    no-dest-nat  
    service-group DG_UDP_SG  
    use-rcv-hop-for-resp  
  port 0 others
```

```

no-dest-nat
  use-rcv-hop-for-resp
  service-group DG_UDP_SG
!
end

```

Proxy Chaining SSLi

For a general overview of proxy chaining, see the *Application Delivery Controller Guide*.

In an SSLi environment, when traffic is routed to an upstream proxy server, to handle HTTPS traffic, some configuration points need to be kept in mind to handle upstream proxy explicit proxy traffic and transparent proxy traffic.

This section provides general explicit proxy configuration steps required for an upstream proxy server and certificate validation setup in a SSLi deployment.

The following topics are covered:

Configuration with Explicit Proxy	308
Configuration with Transparent Proxy	309
CLI Configuration	309

Configuration with Explicit Proxy

Follow the guidelines for ACOS_decrypt device:

- It must contain an SLB server template for the proxy server that includes the upstream proxy's IP address and port.
- In a SLB server policy template, replace `forward-to-service-group` with the `forward-to-proxy` CLI command.
- The virtual server template will specify the ACOS_decrypt IP address.
- The virtual server template's virtual port number must match that of the upstream proxy server port.

Configuration with Transparent Proxy

Follow the guidelines for ACOS_decrypt device:

1. It must contain an SLB server template for the proxy server that includes the upstream proxy's port.
2. In a SLB server policy template, replace `forward-to-internet` with the `forward-to-proxy` CLI command.
3. The virtual server template will have a wildcard VIP (0.0.0.0).

For ACOS_encrypt, the SLB server template must include the following from the upstream proxy

1. In SLB server template, the port of the upstream proxy sever must be specified.
2. In the virtual server template, bind the upstream proxy port (using the service group) with the vport (ACOS_encrypt port).
3. Set `no-dest-nat port-translation` with ACOS_encrypt port in your slb virtual-server template.

CLI Configuration

ACOS_decrypt configuration

1. Create a server template for the upstream proxy server (which is 192.168.90.71) and define its service group for the ACOS_encrypt (port 8080) and port of the proxy server (port 3128). The IP address for the upstream proxy server is required for handling explicit proxy and is not necessary for transparent proxy.

```
slb server proxy 192.168.90.71
  health-check-disable
  port 8080 tcp
    health-check-disable
  port 3128 tcp
    health-check-disable
slb service-group sg-proxy-8080 tcp
  member proxy 8080
slb service-group sg-proxy-3128 tcp
```

```
member proxy 3128
```

2. Traffic will need to be distinguished between HTTP and HTTPS. A class-list of Aho-Corasick string type is created to identify http traffic.

```
class-list HTTP ac
  starts-with http://
```

3. Create a placeholder for ACOS_decrypt and service group for port 80.

```
slb server svr 2.2.2.2
  health-check-disable
  port 80 tcp
    health-check-disable
slb service-group sg tcp
  member svr 80
```

4. Create a policy template for explicit proxy or transparent proxy. This replaces the prior explicit proxy template from the prior example (slb template policy Explicit_Proxy). Create two actions, act-3128 and act-8080. To direct traffic to the upstream proxy server, the `forward-to-proxy` CLI command must be used to ensure the HTTP header remains intact. HTTP traffic is routed through port 3128 directly while HTTPS traffic is inspected through SSLi.

```
slb template policy EP-TP
  forward-policy
    action act-3128
forward-to-proxy sg-proxy-3128 snat Internet_Pool
    action act-8080
forward-to-proxy sg-proxy-8080 snat Internet_Pool
  source src
    match-any
    destination class-list HTTP action act-3128 url priority 1
    destination any action act-8080
```

5. Create a policy template for explicit proxy to enable the server certificate to fetch the traffic and forward it through the explicit proxy instead of the real server as the original SSLi.

```
slb template policy chain
  forward-policy
    action act-8080
```

```

    forward-to-proxy ep8080 support-cert-fetch
    source src
    match-any
    destination any action act-8080
!
```

6. Bind everything with the virtual server template VS_EP. With explicit proxy, provide the ACOS_decrypt ip address (10.10.1.30) and set the upstream proxy's port (3128). The virtual port number in VS_EP is configured to match the upstream explicit proxy port number. The original slb virtual-server template (decrypt_VIP) changes to the following:

```

slb virtual-server VS_EP 10.10.1.30
port 3128 http
    source-nat auto
    service-group sg
    template policy EP-TP
    template dynamic-service DNS
    template client-ssl SSLInsight_decrypt
```

7. With transparent proxy, we use the wildcard vip (0.0.0.0).

```

slb virtual-server VS_TP 0.0.0.0
port 3128 http
    source-nat auto
    service-group sg
    template policy EP-TP
    template dynamic-service DNS
    template client-ssl SSLInsight_decrypt
```

ACOS_encrypt configuration

A placeholder internal server, s1, is created to allow us to add the port and service group, sg-proxy-server-port, for association with the upstream proxy server's port (3128).

```

slb server s1 1.1.1.1
    health-check-disable
port 3128 tcp
    health-check-disable
slb service-group sg-proxy-server-port tcp
member s1 3128
```

The slb virtual-server encrypt_VIP will have a minor change made to the original configuration. The port of the ACOS_encrypt device needs to be set (port 8080 http), so leave this as is. The service group needs to be modified so that the HTTPS traffic that comes in with destination port 8080 leaves with the destination port of the upstream proxy server. This is accomplished by changing service-group DG_SSL_SG to service-group sg-proxy-server-port, which has the upstream proxy server's port of 3128 to move traffic from the ACOS_encrypt device to the upstream proxy server.

```
slb virtual-server encrypt_VIP 0.0.0.0 acl 101
  port 8080 http
    no-dest-nat port-translation
    service-group sg-proxy-server-port
    template server-ssl SSLInsight_encrypt
  port 0 tcp
    no-dest-nat
    service-group DG_TCP_SG
  port 0 udp
    no-dest-nat
    service-group DG_UDP_SG
  port 0 others
    no-dest-nat
    service-group DG_UDP_SG
```

Explicit Proxy with Dual-Stack Forwarding Support

ACOS is enhanced to support IPv4 and IPv6 dual stack forwarding in explicit proxy mode, starting with the DNS resolution. This solution allows the explicit proxy VIP to serve as a gateway between IPv4 (or IPv6) clients and dual-stacked servers, giving preference to IPv6.

Prior to this feature, although the dual-stacked operation was already supported in ACOS through DNS dynamic service templates as well as separate SLB policy templates, it lacked dynamic address translation or interconnectivity, which could lead to service outages or delays in some cases. For instance, you could configure IPv4 and IPv6 DNS servers on the same dynamic service template, but an IPv4 explicit proxy VIP serving an IPv4 client could only reach out to remote servers over IPv4.

Similarly, an IPv6 explicit proxy VIP serving an IPv6 client could only reach out to remote servers over IPv6.

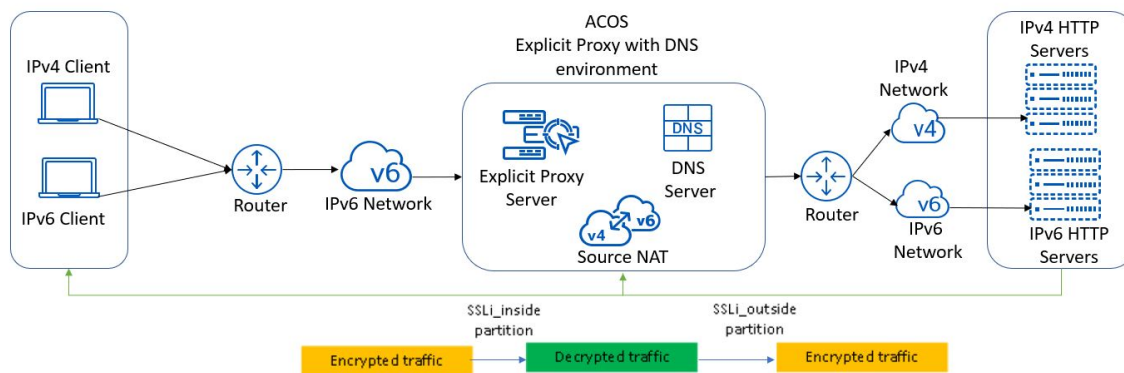
With this feature, once you configure IPv4 and IPv6 DNS servers under the dynamic service template, it behaves as follows:

- An IPv6 DNS server is given preference over an IPv4 server to send DNS requests.
- AAAA (IPv6) and A (IPv4) record requests are sent to the DNS server as two separate requests unless a cached record exists.
- AAAA record replies are preferred over A records.
- Based on the DNS response, the appropriate source NAT pool i.e., either IPv6 NAT-pool or IPv4 NAT-pool is used.

This enables an explicit proxy virtual port to forward HTTP or HTTPS requests via IPv4 or IPv6 networks. Additionally, when this feature is configured, the SSLi certificate fetching module also allows connections with IPv4 addresses on the client side and IPv6 addresses on the server side and vice versa.

NOTE: This feature is not applicable to Transparent Proxy and Proxy chaining.

Deployment

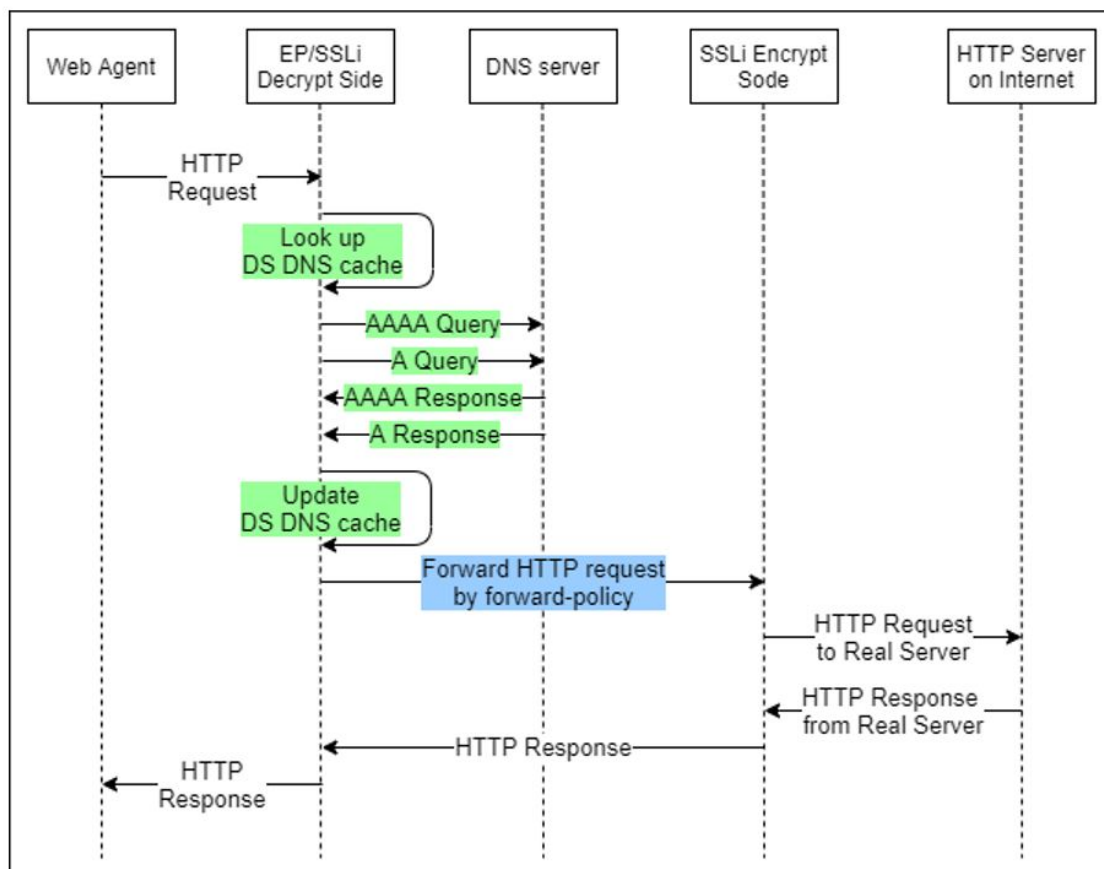


For dual-stack support, the following configuration updates are needed in the existing explicit proxy environment:

1. Create or modify the SLB template dynamic service to configure IPv4 and IPv6 DNS servers on the same template. You can define a maximum of two default DNS servers and 64 class-lists with maximum two DNS server IPs within each dynamic-service template.
2. Create or modify the SLB policy template to enable dual-stack forward policy action and bind the action to any destination rule of the forward policy.
3. Configure IPv4 and IPv6 connectivity using appropriate VIPs, NAT pools, and service groups.
4. Create or modify the SSLi certificate fetching module to support connections with IPv4/IPv6 at the client as well as server side.

Traffic Workflow

The following figure illustrates the traffic workflow for this configuration.



1. After the Explicit Proxy VIP (IPv4 or IPv6) receives the client's HTTP request (over IPv4 or IPv6), a DNS lookup is first performed in the DNS cache.
2. If no records are found, the dynamic service module sends two separate DNS requests, one for AAAA and one for A record to the DNS server.
3. The DNS server responds with an AAAA type and A type record (in that order).
4. The DNS response is updated in the DNS cache.
5. Based on the DNS response, the explicit proxy uses the appropriate source NAT pool i.e., either an IPv4 NAT pool or an IPv6 NAT pool to forward the requests. However, preference is given to AAAA records over A records.

CLI Configuration

In the topology shown in the above figure, the client device is connected to the Explicit Proxy SSLi solution, which is then connected to the external gateway. The ACOS device is connected to the client with a partition called **ACOS_decrypt**. The ACOS device is also connected to the external gateway with a partition called **ACOS_encrypt**.

ACOS_decrypt Configuration

1. Create an SLB dynamic service template and specify default IPv4 and IPv6 Internet DNS resolvers.

```
ACOS(config)# slb template dynamic-service DNS
ACOS(config-dynamic-service)# dns server 10.10.10.27
ACOS(config-dynamic-service)# dns server 1001::1:127
```

Alternatively, you can also configure class-lists with specific DNS server IPs. ACOS will check if the requested domain name matches the entry in the class-list and accordingly based on the destination rule set in forward policy, will select the corresponding DNS server.

```
ACOS(config)# slb template dynamic-service DNS
ACOS(config-dynamic-service)# class-list test priority 30
ACOS(config-dynamic-service-class-list)# dns server 10.2.2.140
ACOS(config-dynamic-service-class-list)# dns server 10.10.20.440
```

2. Configure source NAT pools for the IPv4 service group, IPv6 service group, and

IPv6 fallback service group.

```
ACOS(config)# ip nat pool EP_POOL_v4 5.5.5.100 5.5.5.100 netmask /32
gateway 5.5.5.177
ACOS(config)# ipv6 nat pool EP_POOL_v6 5001::1:50 5001::1:50 netmask
128 gateway 5001::1:171
ACOS(config)# ipv6 nat pool fall_back_v6 5001::1:175 5001::1:175
netmask 128 gateway 5001::1:11
```

3. Create real servers and service groups for IPv4 and IPv6 networks.

```
ACOS(config)# slb server Internet_v4_Server 5.5.5.5
ACOS(config-real server)# health-check-disable
ACOS(config-real server)# port 80 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit
ACOS(config-real server)# port 443 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit

ACOS(config)# slb server Internet_v6_server 7001::1:10
ACOS(config-real server)# health-check-disable
ACOS(config-real server)# port 80 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit
ACOS(config-real server)# port 443 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit

ACOS(config)# slb service-group To_Internet_v4 tcp
ACOS(config-slb svc group)# health-check-disable
ACOS(config-slb svc group)# member Internet_v4_Server 80
ACOS(config-slb svc group-member:80)# exit
ACOS(config-slb svc group)# exit
ACOS(config)# slb service-group To_internet_v6 tcp
ACOS(config-slb svc group)# health-check-disable
ACOS(config-slb svc group)# member Internet_v6_server 80
ACOS(config-slb svc group-member:80)# exit
ACOS(config-slb svc group)# exit
```

4. Configure the client-SSL template to enable SSLi (forward-proxy).

```
ACOS(config)# slb template client-ssl SSLInsight_ClientSide
ACOS(config-client ssl)# forward-proxy-ca-certificate Cert123.pem key
key123456
ACOS(config-client ssl)# forward-proxy-enable
```

5. Configure the forward policy with dual-stack action.

Specify the service groups and source NAT pools for IPv4 and IPv6 networks. Also, specify the fallback service group and corresponding source NAT pool to handle requests when a domain name is not resolved. The unresolved requests are forwarded to the fallback service group.

Bind the dual-stack action to any destination rule of the forward policy template.

```
ACOS(config)# slb template policy DUAL_Stack_6n4
ACOS(config-policy)# forward-policy
ACOS(config-policy-forward-policy)# dual-stack-action FWD_6n4
ACOS(config-policy-forward-policy-dual-st...)# ipv4 To_Internet_v4 snat
EP_POOL_v4
ACOS(config-policy-forward-policy-dual-st...)# ipv6 To_Internet_v6 snat
EP_POOL_v6
ACOS(config-policy-forward-policy-dual-st...)# fallback To_Internet_v6
snat fall_back_v6
ACOS(config-policy-forward-policy-dual-st...)# log
ACOS(config-policy-forward-policy-dual-st...)# exit
ACOS(config-policy-forward-policy)# source ANY
ACOS(config-policy-forward-policy-source)# match-any
ACOS(config-policy-forward-policy-source)# destination any dual-stack-
action FWD_6n4
```

6. Create an explicit proxy virtual server and bind the dual-stack template policy (configured in step 5) to it.

In this configuration, *EP_VIP* is an IPv4 virtual server that functions as an HTTP/HTTPS explicit proxy.

```
ACOS(config)# slb virtual-server EP_VIP 10.10.10.150
ACOS(config-slb vserver)# port 3128 http
ACOS(config-slb vserver-vport)# support-http2
ACOS(config-slb vserver-vport)# template policy DUAL_Stack_6n4
ACOS(config-slb vserver-vport)# source-nat auto
ACOS(config-slb vserver-vport)# service-group To_Internet_v4
```

```
ACOS(config-slb vserver-vport)# template dynamic-service DNS
ACOS(config-slb vserver-vport)# template client-ssl SSLInsight_
ClientSide
```

In this configuration, *EP_VIP_v6* is an IPv6 virtual server that functions as an HTTP/HTTPS explicit proxy.

```
ACOS(config)# slb virtual-server EP_VIP_v6 1001::1:150
ACOS(config-slb vserver)# port 3128 http
ACOS(config-slb vserver-vport)# support-http2
ACOS(config-slb vserver-vport)# template policy DUAL_Stack_6n4
ACOS(config-slb vserver-vport)# source-nat auto
ACOS(config-slb vserver-vport)# service-group To_internet_v6
ACOS(config-slb vserver-vport)# template dynamic-service DNS
ACOS(config-slb vserver-vport)# template client-ssl SSLInsight_
ClientSide
```

ACOS_encrypt Configuration

1. Configure the access list to capture the NATed (as well as unNATed) traffic from the ACOS_decrypt partition.

```
ACOS(config)# ip access-list from_eth5
ACOS(config-access-list:from_eth5)# permit ip any any ethernet 5
ACOS(config-access-list:from_eth5)# exit-module
ACOS(config)# ipv6 access-list v6_from_eth5
ACOS(config-access-list:v6_from_eth5)# permit ipv6 any any ethernet 5
ACOS(config-access-list:v6_from_eth5)# exit-module
```

2. Configure the server SSL template:

```
ACOS(config)# slb template server-ssl Server_SSL
ACOS(config-server ssl)# forward-proxy-enable
```

3. Configure the servers (IPv4 and IPv6) and their ports.

```
ACOS(config)# slb server Webserver 192.168.50.1
ACOS(config-real server)# port 0 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit
ACOS(config-real server)# port 0 udp
ACOS(config-real server-node port)# health-check-disable
```

```
ACOS(config-real server-node port)# exit  
ACOS(config-real server)# port 443 tcp  
ACOS(config-real server-node port)# health-check-disable  
ACOS(config-real server-node port)# exit
```

```
ACOS(config)# slb server Webserver_v6 2001::1:10  
ACOS(config-real server)# port 0 tcp  
ACOS(config-real server-node port)# health-check-disable  
ACOS(config-real server-node port)# exit  
ACOS(config-real server)# port 0 udp  
ACOS(config-real server-node port)# health-check-disable  
ACOS(config-real server-node port)# exit  
ACOS(config-real server)# port 443 tcp  
ACOS(config-real server-node port)# health-check-disable  
ACOS(config-real server-node port)# exit
```

4. Configure service groups to handle different types of encrypted traffic (IPv4 and IPv6).

```
ACOS(config)# slb service-group Serv443 tcp  
ACOS(config-slbg svc group)# health-check-disable  
ACOS(config-slbg svc group)# member Webserver 443  
  
ACOS(config)# slb service-group Serv443_v6 tcp  
ACOS(config-slbg svc group)# health-check-disable  
ACOS(config-slbg svc group)# member Webserver_v6 443  
  
ACOS(config)# slb service-group Web_TCP tcp  
ACOS(config-slbg svc group)# health-check-disable  
ACOS(config-slbg svc group)# member Webserver 0  
  
ACOS(config)# slb service-group Web_TCP_v6 tcp  
ACOS(config-slbg svc group)# health-check-disable  
ACOS(config-slbg svc group)# member Webserver_v6 0  
  
ACOS(config)# slb service-group Web_UDP udp  
ACOS(config-slbg svc group)# health-check-disable  
ACOS(config-slbg svc group)# member Webserver 0
```

```
ACOS(config)# slb service-group Web_UDP_v6 udp
ACOS(config-slb svc group)# health-check-disable
ACOS(config-slb svc group)# member Webserver_v6 0
```

5. Configure the SLB virtual server `ssli_out` and `ssli_out_v6` for encrypting traffic (IPv4 and IPv6):

```
ACOS(config)# slb virtual-server ssli_out 0.0.0.0 acl name from_eth5
ACOS(config-slb vserver)# port 0 others
ACOS(config-slb vserver-vport)# source-nat pool NAT
ACOS(config-slb vserver-vport)# service-group Web_UDP
ACOS(config-slb vserver-vport)# no-dest-nat
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# port 0 tcp
ACOS(config-slb vserver-vport)# source-nat pool NAT
ACOS(config-slb vserver-vport)# service-group Web_TCP
ACOS(config-slb vserver-vport)# no-dest-nat
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# port 0 udp
ACOS(config-slb vserver-vport)# source-nat pool NAT
ACOS(config-slb vserver-vport)# service-group Web_UDP
ACOS(config-slb vserver-vport)# no-dest-nat
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# port 80 http
ACOS(config-slb vserver-vport)# source-nat pool NAT
ACOS(config-slb vserver-vport)# service-group Serv443
ACOS(config-slb vserver-vport)# template server-ssl Server_SSL
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# port 8080 http
ACOS(config-slb vserver-vport)# source-nat pool NAT
ACOS(config-slb vserver-vport)# service-group Serv443
ACOS(config-slb vserver-vport)# template server-ssl Server_SSL
ACOS(config-slb vserver-vport)# no-dest-nat port-translation

ACOS(config)# slb virtual-server ssli_out_v6 :: ipv6-acl v6_from_eth5
ACOS(config-slb vserver)# port 0 others
ACOS(config-slb vserver-vport)# source-nat pool NAT_pool6
ACOS(config-slb vserver-vport)# service-group Web_UDP_v6
ACOS(config-slb vserver-vport)# no-dest-nat
```



```

ACOS(config-slb vserver-vport) # exit
ACOS(config-slb vserver) # port 0 tcp
ACOS(config-slb vserver-vport) # source-nat pool NAT_pool6
ACOS(config-slb vserver-vport) # service-group Web_TCP_v6
ACOS(config-slb vserver-vport) # no-dest-nat
ACOS(config-slb vserver-vport) # exit
ACOS(config-slb vserver) # port 0 udp
ACOS(config-slb vserver-vport) # source-nat pool NAT_pool6
ACOS(config-slb vserver-vport) # service-group Web_UDP_v6
ACOS(config-slb vserver-vport) # no-dest-nat
ACOS(config-slb vserver-vport) # exit
ACOS(config-slb vserver) # port 80 http
ACOS(config-slb vserver-vport) # source-nat pool NAT_pool6
ACOS(config-slb vserver-vport) # service-group Serv443_v6
ACOS(config-slb vserver-vport) # template server-ssl Server_SSL
ACOS(config-slb vserver-vport) # no-dest-nat port-translation
ACOS(config-slb vserver-vport) # exit
ACOS(config-slb vserver) # port 8080 http
ACOS(config-slb vserver-vport) # source-nat pool NAT_pool6
ACOS(config-slb vserver-vport) # service-group Serv443_v6
ACOS(config-slb vserver-vport) # template server-ssl Server_SSL
ACOS(config-slb vserver-vport) # no-dest-nat port-translation
ACOS(config-slb vserver-vport) # exit

```

Drop and Drop-Redirect-URL Message Responses

NOTE: This section requires that the explicit proxy configuration is known for ACOS_decrypt and ACOS_encrypt. To configure explicit proxy, refer to [Deployment Example](#).

Starting from ACOS 4.1.4, there is support for configuring drop-message and drop-redirect-url options for HTTPS traffic in explicit proxy for SSLi. This feature enables the network administrator to either configure a customized drop message or a customized redirect URL for specific websites tagged under the explicit proxy configuration. The SSLi deployment must complete the SSL intercept before being able to send a drop or redirect message.

Configure ACOS_decrypt with some additional actions for the explicit proxy template. There are no additional changes required for ACOS_encrypt. The following are the guidelines for the configuration of drop and drop-redirect-url messages in ACOS_decrypt:

- Prior to configuring explicit proxy, determine what port number and what IP address are to be used for explicit proxy.
- (Optional) Create the source-NAT pool of IP addresses required by the forward-to-internet action.
- Create a service group under the vPort for the forward-to-internet action.
- Define the class-lists or web-category for specifying the destinations for the actions created for explicit proxy.
- Create the explicit proxy template with the drop and drop-redirect actions. Also include the forward-to-internet option for allowed traffic.
- Bind the explicit proxy template and the client-SSL template to the VIP.

ACOS_decrypt Configuration

The following is a sample configuration that displays the following logic:

- If a user accesses **www.netflix.com**, a drop message is displayed.
- If a user accesses **www.game.com**, the user is redirected to the HTTPS website configured, which is **https://www.apple.com**.
- If a user accesses **www.poker.com**, the user is redirected to the website configured, which is **http://192.168.98.115**.

Configure the explicit proxy policy template called `ep-template`, with the following properties:

- For the action `FORWARD`, the service group `sg-8080` is added for forwarding the allowed traffic to the Internet.
- For the action `GAMBLE`, the action is to `drop` and the redirect the URL is `http://192.168.98.115`.
- For the action `GAME`, the action is to `drop` and the redirect the URL is `https://www.apple.com`.

- For the action `NETFLIX`, the action is to drop and a message is displayed as `This website is not allowed, contact your networks admin for more info.`

```
ACOS(config)# slb template policy ep-template
ACOS(config-policy)# forward-policy

ACOS(config-policy-forward-policy)# action forward
ACOS(config-policy-forward-policy-action)# forward-to-internet sg-8080
snat NAT
ACOS(config-policy-forward-policy-action)# exit

ACOS(config-policy-forward-policy)# action GAMBLE
ACOS(config-policy-forward-policy-action)# drop
ACOS(config-policy-forward-policy-action)# drop-redirect-url
http://192.168.98.115
ACOS(config-policy-forward-policy-action)# exit

ACOS(config-policy-forward-policy)# action GAME
ACOS(config-policy-forward-policy-action)# drop
ACOS(config-policy-forward-policy-action)# drop-redirect-url
https://www.apple.com
ACOS(config-policy-forward-policy-action)# exit

ACOS(config-policy-forward-policy)# action NETFLIX
ACOS(config-policy-forward-policy-action)# drop
ACOS(config-policy-forward-policy-action)# drop-message "This website is
not allowed, contact your networks admin for more info"
ACOS(config-policy-forward-policy-action)# exit

ACOS(config-policy-forward-policy)# source SRC
ACOS(config-policy-forward-policy-source)# match-any
ACOS(config-policy-forward-policy-source)# destination class-list gamble
action GAMBLE host priority 800
ACOS(config-policy-forward-policy-source)# destination class-list game
action GAME host priority 900
ACOS(config-policy-forward-policy-source)# destination class-list netflix
action NETFLIX host priority 1000
ACOS(config-policy-forward-policy-source)# destination any action FORWARD
ACOS(config-policy-forward-policy-source)# exit
ACOS(config-policy-forward-policy)# exit
```

```
ACOS(config-policy)# exit
```

Configure the client-SSL template called `c1` and enable forward proxy.

```
ACOS(config)# slb template client-ssl c1  
ACOS(config-client ssl)# forward-proxy-ca-certificate Cert123.pem key  
key123  
ACOS(config-client ssl)# forward-proxy-enable
```

Configure the virtual server called `vs`. Associate the explicit proxy template, the client-SSL template, and the fake-sg service group to the Vport of port 80.

```
ACOS(config)# slb virtual-server vs 192.168.91.105  
ACOS(config-slb vserver)# port 80 http  
ACOS(config-slb vserver-vport)# source-nat auto  
ACOS(config-slb vserver-vport)# service-group fake-sg  
ACOS(config-slb vserver-vport)# template policy ep-template  
ACOS(config-slb vserver-vport)# template client-ssl c1  
ACOS(config-slb vserver-vport)# exit  
ACOS(config-slb vserver-vport)# exit
```

Consolidated Configuration

The following is an excerpt of the `ACOS_decrypt` configuration for configuring message-drop and redirect-URLs:

```
!Configure the class-list.  
class-list gamble ac  
    contains poker  
!  
class-list game ac  
    contains game  
!  
class-list netflix ac  
    contains netflix  
!  
class-list permit  
    192.168.99.24/32  
!  
!Configure the NAT pool.
```

```
ip nat pool NAT 192.168.91.24 192.168.91.24 netmask /32 gateway
192.168.91.254
```

!Configure the real servers.

```
slb server fake 1.1.1.1
    health-check-disable
    port 1111 tcp
    health-check-disable
!
slb server s1 192.168.221.70
    health-check-disable
    port 80 tcp
    health-check-disable
    port 8080 tcp
    health-check-disable
```

!Configure the service groups.

```
slb service-group fake-sg tcp
    member fake 1111
!
slb service-group sg-8080 tcp
    member s1 8080
!
```

!Configure the client-ssl template.

```
slb template client-ssl c1
    forward-proxy-ca-certificate Cert123.pem key key123
    forward-proxy-enable
!
```

!Configure the explicit policy template.

```
slb template policy ep-template
    forward-policy
    action FORWARD
        forward-to-internet sg-8080 snat NAT
    action GAMBLE
        drop
        drop-redirect-url http://192.168.98.115
    action GAME
        drop
        drop-redirect-url https://www.apple.com
    action NETFLIX
        drop
```

```
        drop-message "This website is not allowed, contact your networks
admin for more info"
    source SRC
    match-any
    destination class-list gamble action GAMBLE host priority 800
    destination class-list game action GAME host priority 900
    destination class-list netflix action NETFLIX host priority 1000
    destination any action FORWARD
!
!Configure the virtual server.
slb virtual-server vs 192.168.91.105
    port 80 http
    source-nat auto
    service-group fake-sg
    template policy ep-template
    template client-ssl c1
```

Drop and Drop-Redirect-URL Priorities

For an explicit proxy, the priority of the match condition determines which action is selected for the request. For example, the following is a sample configuration excerpt:

```
!
class-list cnn ac
    contains cnn
!
class-list sport ac
    contains sport
!
slb template policy ep-template
    forward-policy
    action A1
        forward-to-internet sg-8080 snat NAT
    action drop
        drop
        drop-message "Not allowed"
    source ANY
    match-any
    destination class-list cnn action drop host priority 500
```

```
destination class-list sport action A1 url priority 1000
```

In the configuration sample, if the request contains the word `cnn`, the action is to drop the request.

If the request contains the word `sport`, the action is to forward to internet. The URL `cnn.com/sport` matches both conditions, however the priority of the `sport` action is higher than the priority of the `cnn` action. As a result, the request is forwarded to internet under the action `A1`.

For the same configuration, if the priorities are reversed as `destination class-list cnn action drop host priority 1000` and `destination class-list sport action A1 url priority 500`, the request `cnn.com/sport` is dropped and the message "Not Allowed" is displayed.

The drop-message and redirect-URL options in the configuration changes the behavior of the explicit proxy from previous ACOS versions. If none of the options are configured, the ACOS device drops the CONNECT request immediately, instead of doing an SSL negotiation.

Virtual Wire with SSLi Deployment

ACOS supports virtual wire (vwire) or bump-in-the-wire, which streamlines the deployment process in an SSLi solution, allowing the Thunder to be inserted into the network using an IP-Less configuration. Virtual wire logically bridges two Ethernet interfaces like a physical wire, and VLAN or routing configuration changes are not required. For packets that traverse the vwire, the Layer 2 headers are not modified.

In the case of virtual wires for SSLi with Explicit Proxy deployment, the `support-cert-fetch` option under `forward-to-proxy` is used for certificate fetch traffic to go to the Explicit Proxy (EP). ACOS uses the management interface to resolve the domain name of the CA certificate and subsequently uses the virtual wire endpoint to forward the validation.

NOTE:

- In a virtual wire IP-less SSLi deployment, it is required to configure DNS and the default gateway for the management interface to resolve the CA certificate domain name.
- In any typical SSLi deployment, the `support-cert-fetch` option is supported only when the virtual server is wildcard VIP 0.0.0.0 (transparent proxy).

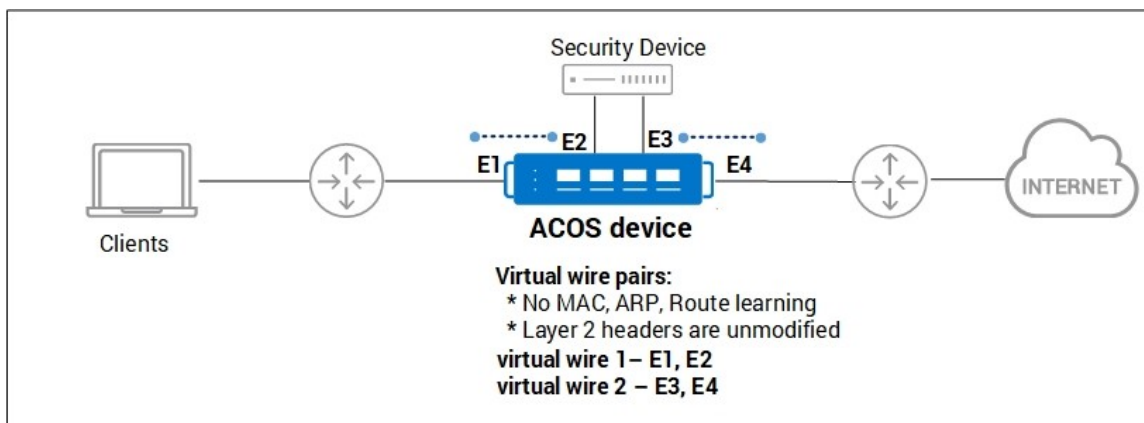
For more information on virtual wire configuration, see *Network Configuration Guide*.

The following topics are covered:

SSLi IP-Less Deployment	328
SSLi IP-Less Deployment with Explicit Proxy and Cert Validation	331
SSLi IP-Less Deployment with Layer 2 Health Monitoring	333
SSLi IP-Less Deployment with Multiple Ports for Failover	337
SSLi IP-Less Deployment with Layer 3 Health Monitoring	344

SSLi IP-Less Deployment

Figure 27 : Virtual Wire SSLi deployment



The ACOS device in [Figure 27](#) uses multiple virtual wire pairs. SSL traffic is decrypted and sent to the security device using virtual wire 1, is re-encrypted using virtual wire 2, and sent to the destination on the Internet.

CLI Configuration

1. Configure the virtual wire endpoint.

```
ACOS(config)# interface management
ACOS(config-if:management)# ip address 192.168.93.140 255.255.255.0
ACOS(config-if:management)# ip default-gateway 192.168.93.1
ACOS(config-if:management)# interface ethernet 1
ACOS(config-if:ethernet:1)# enable
ACOS(config-if:ethernet:1)# virtual-wire
ACOS(config-if:ethernet:1)# ip allow-promiscuous-vip
ACOS(config-if:ethernet:1)# interface ethernet 2
ACOS(config-if:ethernet:2)# enable
ACOS(config-if:ethernet:2)# virtual-wire
ACOS(config-if:ethernet:2)# interface ethernet 3
ACOS(config-if:ethernet:3)# enable
ACOS(config-if:ethernet:3)# virtual-wire
ACOS(config-if:ethernet:3)# ip allow-promiscuous-vip
ACOS(config-if:ethernet:3)# interface ethernet 4
ACOS(config-if:ethernet:3)# enable
ACOS(config-if:ethernet:3)# virtual-wire
ACOS(config-virtual-wire)# virtual-wire 1
ACOS(config-virtual-wire:1)# ethernet 1 ethernet 2
ACOS(config-virtual-wire:1)# virtual-wire 2
ACOS(config-virtual-wire:2)# ethernet 3 ethernet 4
ACOS(config-virtual-wire:2)# exit
```

2. Configure the SLB server and service-group components for the security device.

```
ACOS(config)# slb server sec-dev 10.20.1.142
ACOS(config-real server)# health-check-disable
ACOS(config-real server)# port 443 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit
ACOS(config-real server)# exit
ACOS(config-real server)# port 8080 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit
ACOS(config)# slb service-group sec-dev_tcp443 tcp
ACOS(config-slb svc group)# member sec-dev 443
ACOS(config-slb svc group)# exit
```

```
ACOS(config)# slb service-group sec-dev_tcp8080 tcp
ACOS(config-slb svc group)# member sec-dev 8080
ACOS(config-slb svc group)# exit
```

3. Configure the Server SSL and Client SSL templates.

```
ACOS(config)# slb template server-ssl serverssl
ACOS(config-server ssl)# forward-proxy-enable
ACOS(config-server ssl)# exit
ACOS(config)# slb template client-ssl clientssl
ACOS(config-client ssl)# forward-proxy-ca-certificate Cert123.pem key
key123
ACOS(config-client ssl)# forward-proxy-enable
ACOS(config-client ssl)# exit
```

4. Configure the SLB Virtual Server ssli_in for decrypting traffic client traffic and sending it to the security device. ACLs are added for permitting traffic to the VIPs from client side and server side.

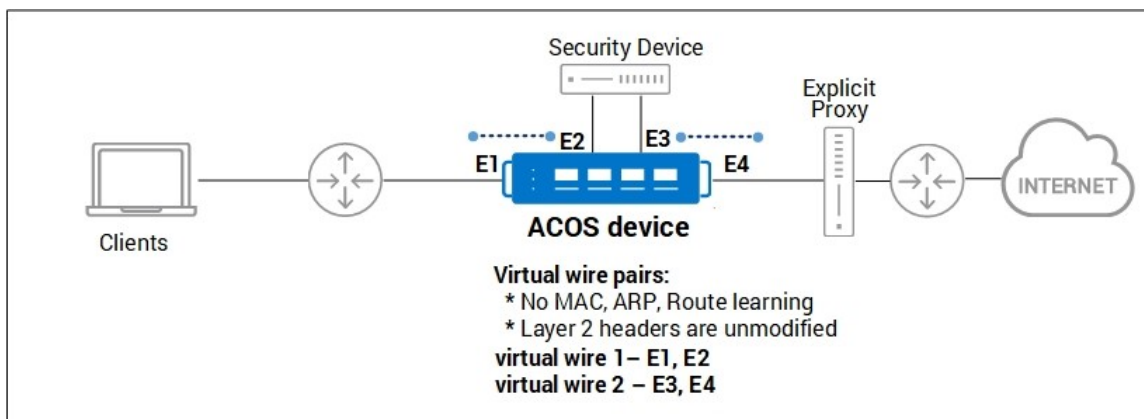
```
ACOS(config)# access-list 198 permit ip any any ethernet 3
ACOS(config)# access-list 199 permit ip any any ethernet 1
ACOS(config)# slb virtual-server ssli_in 0.0.0.0 acl 199
ACOS(config-slb vserver)# port 443 https
ACOS(config-slb vserver-vport)# service-group sec-dev_tcp8080
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# template client-ssl clientssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
```

5. Configure the SLB Virtual Server ssli_out for encrypting traffic.

```
ACOS(config)# slb virtual-server ssli_out 0.0.0.0 acl 198
ACOS(config-slb vserver)# port 443 tcp
ACOS(config-slb vserver-vport)# service-group sec-dev_tcp443 tcp
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# no-dest-nat
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# port 8080 http
ACOS(config-slb vserver-vport)# service-group sec-dev_tcp443 tcp
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# template server-ssl serverssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
```

SSLi IP-Less Deployment with Explicit Proxy and Cert Validation

Figure 28 : Virtual Wire and SSLi with Explicit Proxy



The ACOS device in [Figure 28](#) uses virtual wires for SSLi with Explicit Proxy and certificate validation. The following commands configure the ACOS device with virtual wires for SSLi with Explicit Proxy and certificate validation.

CLI Configuration

1. Configure the IP address for the DNS server for domain resolution and the default route for the management interface. This configuration is required for IP-Less since the interface IP is not used for CA certificate validation.

```
ACOS(config)# ip dns primary 8.8.8.8
ACOS(config)# ip route 8.8.8.8 /32 192.168.93.1
```

2. Configure the explicit proxy server with TCP ports 3128 and 8080 and the service-group, disable the health checks.

```
ACOS(config)# slb server ep 10.20.1.145
ACOS(config-real server)# health-check-disable
ACOS(config-real server)# port 3128 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit
ACOS(config-real server)# port 8080 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit
ACOS(config)# slb service-group ep3128 tcp
ACOS(config-slb svc group)# member ep 3128
```

```
ACOS(config-slb svc group)# exit
ACOS(config)# slb service-group ep8080 tcp
ACOS(config-slb svc group)# member ep 8080
ACOS(config-slb svc group)# exit
```

3. Configure the SSL certificate.

```
ACOS(config)# slb template client-ssl clientssl
ACOS(config-client ssl)# forward-proxy-ca-certificate Cert123.pem key
key123
ACOS(config-client ssl)# forward-proxy-trusted-ca new_self.crt
ACOS(config-client ssl)# forward-proxy-enable
ACOS(config-client ssl)# exit
```

4. Configure the SSL template policy. Use the `support-cert-fetch` command in the virtual wire deployment.

```
ACOS(config)# slb template policy chain
ACOS(config-policy)# forward-policy
ACOS(config-policy-forward-policy)# action act-8080
ACOS(config-policy-forward-policy-action)# forward-to-proxy ep8080
support-cert-fetch
ACOS(config-policy-forward-policy-action)# exit
ACOS(config-policy-forward-policy)# source Any_Source
ACOS(config-policy-forward-policy-source)# match-any
ACOS(config-policy-forward-policy-source)# destination any action act-
8080
```

5. Configure the SLB virtual servers, the explicit proxy, and SSL templates.

```
ACOS(config)# access-list 198 permit ip any any ethernet 3
ACOS(config)# access-list 199 permit ip any any ethernet 1
ACOS(config)# slb virtual-server ssli_in 0.0.0.0 acl 199
ACOS(config-slb vserver)# port 3128 http
ACOS(config-slb vserver-vport)# template policy chain
ACOS(config-slb vserver-vport)# service-group ep8080
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# template client-ssl clientssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit

ACOS(config)# slb virtual-server ssli_out 0.0.0.0 acl 198
```

```

ACOS(config-slb vsrver)# port 443 tcp
ACOS(config-slb vsrver-vport)# service-group g_tcp0
ACOS(config-slb vsrver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vsrver-vport)# no-dest-nat port
ACOS(config-slb vsrver-vport)# exit
ACOS(config-slb vsrver)# port 8080 http
ACOS(config-slb vsrver-vport)# service-group ep3128
ACOS(config-slb vsrver-vport)# template server-ssl serverssl
ACOS(config-slb vsrver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vsrver-vport)# no-dest-nat port-translation

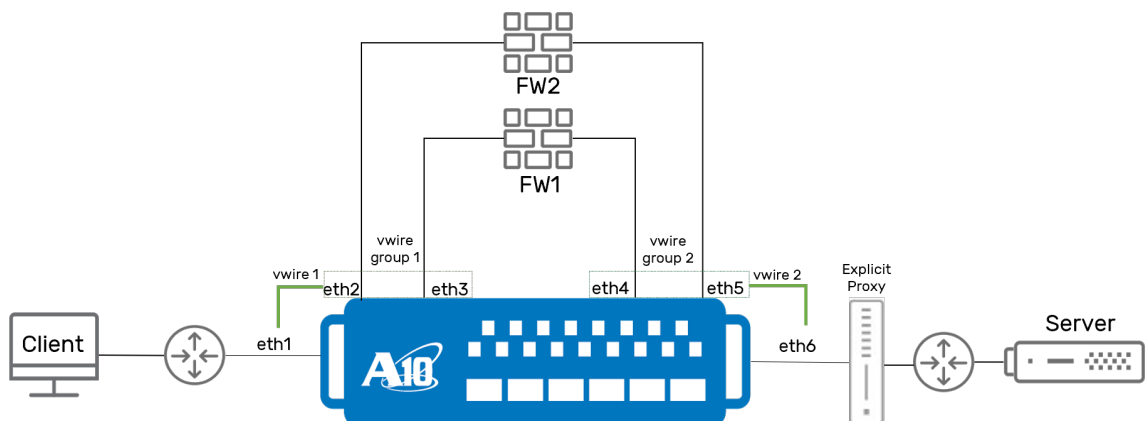
```

SSLi IP-Less Deployment with Layer 2 Health Monitoring

ACOS provides the layer 2 health monitoring infrastructure support for Virtual Wire with SSLi (IP-less) deployments where monitoring other network devices is required for failover. For example, in the case of firewall load balance by virtual wire or firewall service in SSLi deployment, the device like a firewall between decrypt VIP and encrypt VIP could be down and such health checks should be notified to the server.

This feature can detect the status of the IP-Less device and can connect the path status to the real server interface. For example, if the internal path is blocked due to a firewall failure, Thunder can detect the change in the state and notify the interface server on the path to reflect the condition. Once the real server's state has been updated, the application can select a working server to process the subsequent request.

Figure 29 : Deployment Example



CLI Configuration

1. Configure access-list, interface servers, and virtual wire endpoints:

```
ACOS(config)# access-list 198 permit ip any any ethernet 4
ACOS(config)# access-list 198 permit ip any any ethernet 5
ACOS(config)# access-list 199 permit ip any any ethernet 1
ACOS(config)# bfd enable
ACOS(config)# ipv6 access-list in6
ACOS(config-access-list:in6)# permit tcp any any ethernet 1
ACOS(config-access-list:in6)# exit
ACOS(config)# ipv6 access-list out6
ACOS(config-access-list:out6)# permit tcp any any ethernet 4
ACOS(config-access-list:out6)# permit tcp any any ethernet 5
ACOS(config-access-list:out6)# exit
ACOS(config)# interface ethernet 1
ACOS(config-if:ethernet:1)# enable
ACOS(config-if:ethernet:1)# virtual-wire
ACOS(config-if:ethernet:1)# ip allow-promiscuous-vip
ACOS(config-if:ethernet:1)# interface ethernet 2
ACOS(config-if:ethernet:2)# enable
ACOS(config-if:ethernet:2)# virtual-wire
ACOS(config-if:ethernet:2)# interface ethernet 3
ACOS(config-if:ethernet:3)# enable
ACOS(config-if:ethernet:3)# virtual-wire
ACOS(config-if:ethernet:3)# ip allow-promiscuous-vip
ACOS(config-if:ethernet:3)# interface ethernet 4
ACOS(config-if:ethernet:4)# enable
ACOS(config-if:ethernet:4)# virtual-wire
ACOS(config-if:ethernet:4)# ip allow-promiscuous-vip
ACOS(config-if:ethernet:4)# interface ethernet 5
ACOS(config-if:ethernet:5)# enable
ACOS(config-if:ethernet:5)# virtual-wire
ACOS(config-if:ethernet:5)# ip allow-promiscuous-vip
ACOS(config-if:ethernet:5)# interface ethernet 6
ACOS(config-if:ethernet:6)# enable
ACOS(config-if:ethernet:6)# virtual-wire
ACOS(config-if:ethernet:6)# virtual-wire-ethernet-group 1
ACOS(config-virtual-wire-ethernet-group:1)# ethernet 2 to 3
```

```

ACOS(config-virtual-wire-ethernet-group:1)# virtual-wire-ethernet-group
2
ACOS(config-virtual-wire-ethernet-group:2)# ethernet 4 to 5
ACOS(config-virtual-wire-ethernet-group:2)# virtual-wire 1
ACOS(config-virtual-wire:1)# ethernet 1 virtual-wire-ethernet-group 1
ACOS(config-virtual-wire:1)# virtual-wire 2
ACOS(config-virtual-wire:2)# ethernet 6 virtual-wire-ethernet-group 2
ACOS(config-virtual-wire:2)# exit

```

2. Configure the individual health monitor/check object and then configure the layer 2 BFD settings.

```

ACOS(config)# system health-check hm-test
ACOS(config-health-check)# 12-bfd rx-interval 50 tx-interval 50
multiplier 3

```

3. Configure the layer 2 health monitor path object at the global configuration mode.

```

ACOS(config)#system path hm-obj-test
ACOS(config-path)#interface-pair ethernet 2 ethernet 5
ACOS(config-path)# use hm-test
ACOS(config)# system path hc-test
ACOS(config-path)# interface-pair ethernet 3 ethernet 4
ACOS(config-path)# use hm-test

```

4. Configure the SLB server and service-group components for the security device:

```

ACOS(config)# slb server gw ethernet 6
ACOS(config-real server)# port 443 tcp
ACOS(config)# exit
ACOS(config)# slb server S1 ethernet 2
ACOS(config-real server)# 12-health-check-path hm-obj-test
ACOS(config-real server)# port 443 tcp
ACOS(config-real server)# port 8080 tcp
ACOS(config)# exit
ACOS(config)# slb server S2 ethernet 3
ACOS(config-real server)# 12-health-check-path hc-test
ACOS(config-real server)# port 443 tcp
ACOS(config-real server)# port 8080 tcp
ACOS(config-real server)# exit
ACOS(config)# slb service-group sg_fw_443 tcp

```

```

ACOS(config-slb svc group)# member S2 443
ACOS(config-slb svc group)# member S3 443
ACOS(config-slb svc group)# exit
ACOS(config)# slb service-group sg_fw_8080 tcp
ACOS(config-slb svc group)# member S2 8080
ACOS(config-slb svc group)# member S3 8080
ACOS(config-slb svc group)# exit
ACOS(config)# slb service-group sg_gw_443 tcp
ACOS(config-slb svc group)# member gw 443
ACOS(config-slb svc group)# exit

```

5. Configure the Server SSL and Client SSL templates:

```

ACOS(config)# slb template server-ssl serverssl
ACOS(config-server ssl)# forward-proxy-enable
ACOS(config-server ssl)# exit
ACOS(config)# slb template client-ssl clientssl
ACOS(config-client ssl)# forward-proxy-ca-certificate Cert123.pem key
key123
ACOS(config-client ssl)# forward-proxy-enable
ACOS(config-client ssl)# exit

```

- Configure the SLB Virtual Server for decrypting traffic client traffic and sending it to the security device.

```

ACOS(config)# slb virtual-server ssli_in 0.0.0.0 acl 199
ACOS(config-slb vserver)# port 443 https
ACOS(config-slb vserver-vport)# service-group sg_fw_8080
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# template client-ssl clientssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
ACOS(config)# slb virtual-server ssli_ig6 :: ipv6-acl in6
ACOS(config-slb vserver)# port 443 https
ACOS(config-slb vserver-vport)# service-group sg_fw_8080
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# template client-ssl clientssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit

```

- Configure the SLB Virtual Server for encrypting traffic:


```
ACOS(config)# slb virtual-server ssli_out 0.0.0.0 acl 198
ACOS(config-slb vserver)# port 443 tcp
ACOS(config-slb vserver-vport)# service-group sg_gw_443
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# no-dest-nat
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# port 8080 http
ACOS(config-slb vserver-vport)# service-group sg_gw_443
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# template server-ssl serverssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# exit
ACOS(config)# slb virtual-server ssli_o6 :: ipv6-acl out6
ACOS(config-slb vserver)# port 443 tcp
ACOS(config-slb vserver-vport)# service-group sg_gw_443
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# no-dest-nat
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# port 8080 http
ACOS(config-slb vserver-vport)# service-group sg_gw_443
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# template server-ssl serverssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# exit
```

SSLi IP-Less Deployment with Multiple Ports for Failover

ACOS supports the detection and tracking of active VLAN pairs in the virtual wire environment. Traffic is thus distributed seamlessly if the VLAN is changed during failover scenarios or in topologies where the traffic uses a different VLAN for forward and reverse traffic. This feature applies to both inbound and outbound SSLi.

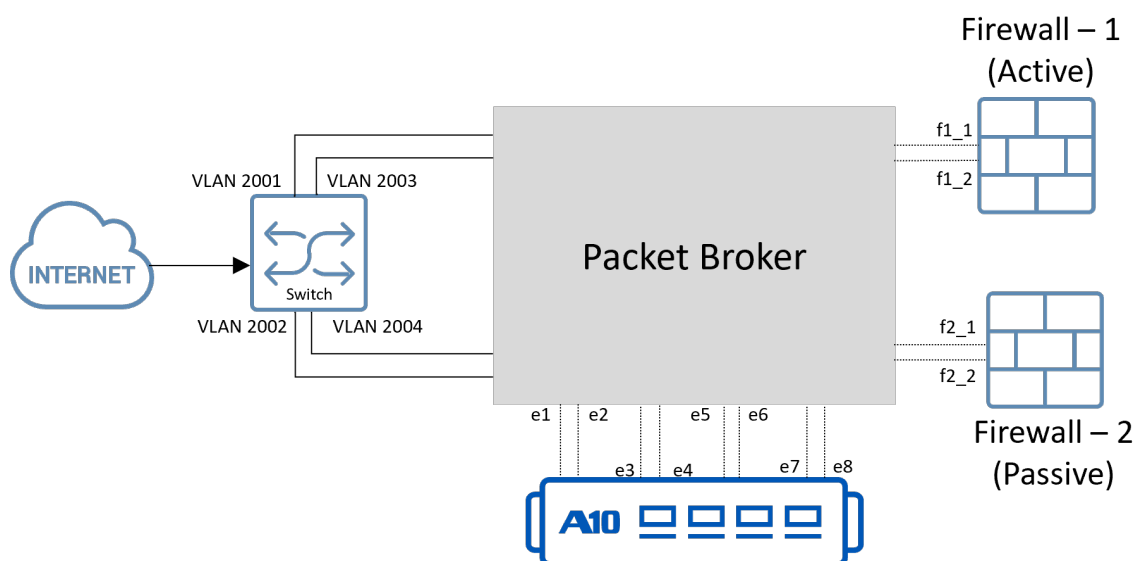
In a virtual wire with SSLi (IP-less) deployment, various topologies can be considered. For example, see [Figure 30](#). In such deployments, packet brokers play a crucial role in managing network traffic. Packet brokers have two types of ports: network ports and tool ports. The network ports route the traffic to the network, and the tool ports send the service traffic to the security devices. Network ports are in pairs, with one

facing the upstream port (for example, a switch) and the other facing the downstream port (for example, a firewall). The packet broker is inserted between the switch and the firewall, which acts as a “bump in the wire.”

When there are multiple upstream and downstream ports, for example, multiple switches or firewall devices in a cluster to act as active-standby for failover scenarios, packet brokers use VLAN tagging to determine the source and destination of each packet. Packet broker inserts an outer VLAN tag using Q-in-Q technology for each network port pair. This tag acts as a signature for that specific port pair, allowing the packet broker to route traffic correctly.

In this kind of topology, it is important for ACOS to know about the VLAN pairs in the network and keep a track of which VLAN pair is active during failover. The virtual wire VLAN pairs facilitate ACOS to track the active VLAN pairs for virtual wire sessions. ACOS detects and updates the active VLAN pair in the network and forwards the traffic to the active server.

Figure 30 : Virtual Wire SSLi Firewall Deployment Example



In the above example,

- The switch has 2 VLAN tags in a stable state that work concurrently. In the above example, VLAN 2001 and 2003 maps with Firewall - 1 and VLAN 2002 and 2004 maps with Firewall – 2. After firewall failover, ACOS chooses another pair of VLANs to route the traffic.

- The switch inserts an outer VLAN tag into each network port pair. It inserts outer VLAN tag 2001 for the link between Firewall - 1 and switches; and outer VLAN tag 2002 for the connection between Firewall - 2 and switches.
- There are four network port pairs: f1_1, f1_2, f2_1, and f2_2. So, the Q-in-Q VLAN kapping will be as follows:
 - f1_1:2001
 - f1_2:2003
 - f2_1:2002
 - f2_2:2004
- Each firewall consists of two ports that are used as a trunk. The traffic flows either on f1_1 and f1_2 or f2_1 and f2_2. In this case, Firewall-1 can choose either f1_1 or f1_2.
- There are two separate virtual-wire paths: e1 to e4 and e5 to e8. The packet broker distributes the traffic seamlessly to both these paths.

To ensure that SSLi sessions are forwarded to the active VLAN during failover, you can configure virtual wire VLANs to facilitate ACOS to track the active VLAN tags for virtual wire sessions.

For more information, see *Network Configuration Guide*.

CLI Configuration

1. Configure access-list, interface servers, and virtual wire endpoints.

```
ACOS(config)# access-list 196 permit ip any any ethernet 1
ACOS(config)# access-list 197 permit ip any any ethernet 3
ACOS(config)# access-list 198 permit ip any any ethernet 5
ACOS(config)# access-list 199 permit ip any any ethernet 7
ACOS(config)# interface ethernet 1
ACOS(config-if:ethernet:1)# enable
ACOS(config-if:ethernet:1)# virtual-wire update-l2-info vlan-learning
enable
ACOS(config-if:ethernet:1)# ip allow-promiscuous-vip
ACOS(config-if:ethernet:1)# exit
ACOS(config)# interface ethernet 2
ACOS(config-if:ethernet:2)# enable
```

```
ACOS(config-if:ethernet:2)# virtual-wire update-l2-info vlan-learning enable
ACOS(config-if:ethernet:2)# exit
ACOS(config)# interface ethernet 3
ACOS(config-if:ethernet:3)# enable
ACOS(config-if:ethernet:3)# virtual-wire update-l2-info vlan-learning enable
ACOS(config-if:ethernet:3)# ip allow-promiscuous-vip
ACOS(config-if:ethernet:3)# exit
ACOS(config)# interface ethernet 4
ACOS(config-if:ethernet:4)# enable
ACOS(config-if:ethernet:4)# virtual-wire update-l2-info vlan-learning enable
ACOS(config-if:ethernet:4)# exit
ACOS(config)# interface ethernet 5
ACOS(config-if:ethernet:5)# enable
ACOS(config-if:ethernet:5)# virtual-wire update-l2-info vlan-learning enable
ACOS(config-if:ethernet:5)# ip allow-promiscuous-vip
ACOS(config-if:ethernet:5)# exit
ACOS(config)# interface ethernet 6
ACOS(config-if:ethernet:6)# enable
ACOS(config-if:ethernet:6)# virtual-wire update-l2-info vlan-learning enable
ACOS(config-if:ethernet:6)# exit
ACOS(config)# interface ethernet 7
ACOS(config-if:ethernet:7)# enable
ACOS(config-if:ethernet:7)# virtual-wire update-l2-info vlan-learning enable
ACOS(config-if:ethernet:7)# ip allow-promiscuous-vip
ACOS(config-if:ethernet:7)# exit
ACOS(config)# interface ethernet 8
ACOS(config-if:ethernet:8)# enable
ACOS(config-if:ethernet:8)# virtual-wire update-l2-info vlan-learning enable
ACOS(config-if:ethernet:8)# exit
```

```
ACOS(config)# virtual-wire 1
ACOS(config-virtual-wire:1)# ethernet 1 ethernet 2
ACOS(config)# virtual-wire 2
ACOS(config-virtual-wire:2)# ethernet 3 ethernet 4
ACOS(config)# virtual-wire 3
ACOS(config-virtual-wire:3)# ethernet 5 ethernet 6
ACOS(config)# virtual-wire 4
ACOS(config-virtual-wire:4)# ethernet 7 ethernet 8
```

2. Configure the active and passive VLANs.

The two VLAN pairs are configured. Active **VLAN 2001** and **2003**, and passive **VLAN 2002** and **2004**.

```
ACOS(config)# vlan 2001
ACOS(config-vlan:2001)# exit
ACOS(config)# vlan 2002
ACOS(config-vlan:2002)# exit
ACOS(config)# vlan 2003
ACOS(config-vlan:2003)# exit
ACOS(config)# vlan 2004
ACOS(config-vlan:2004)# exit
```

3. Configure the global options in the virtual wire. This is an optional configuration and affect how ACOS updates and tracks the active VLAN.

The virtual wire update period option is set to **60** seconds. The update active VLAN option is set to **all**, forcing ACOS to update VLAN by any packet every 60 seconds.

```
ACOS(config)# virtual-wire-global
ACOS(config-virtual-wire-global)# vlan-update-period 60
ACOS(config-virtual-wire-global)# update-active-vlan all
```

4. Configure the bridge VLAN group to match the session for packets with VLAN in both VLAN pairs.

The bridge VLAN group **1** is configured. The previously defined active and passive **VLAN 2001 to 2004** is configured for VLAN-to-VLAN bridging.

```
ACOS(config)# bridge-vlan-group 1
ACOS(config-bridge-vlan-group:1)# vlan 2001 to 2004
```

5. Configure and map the two active VLAN with the virtual wire.

The virtual wire VLAN pair **1** is configured. The previously defined active **VLAN 2001** and **2003** are mapped.

```
ACOS(config)# virtual-wire-vlan-pair 1
ACOS(config-virtual-wire-vlan-pair:1)# vlan 2001 vlan 2003
```

6. Configure and map the two passive VLAN with the virtual wire, which will be used after failover.

The virtual wire VLAN pair **2** is configured. The previously defined passive **VLAN 2002** and **2004** are mapped.

```
ACOS(config)# virtual-wire-vlan-pair 2
ACOS(config-virtual-wire-vlan-pair:2)# vlan 2002 vlan 2004
```

7. Configure and map the two VLAN pairs in a virtual wire pair set.

The virtual wire VLAN pair set **1** is configured. The previously defined active and passive virtual wire VLAN pairs **1** and **2** are configured.

```
ACOS(config)# virtual-wire-vlan-pair-set 1
ACOS(config-virtual-wire-vlan-pair-set:1)# virtual-wire-vlan-pair 1
ACOS(config-virtual-wire-vlan-pair-set:1)# virtual-wire-vlan-pair 2
```

8. Configure the SLB server and service-group components for the security device.

```
ACOS(config)# slb server S1 ethernet 4
ACOS(config-real server)# port 443 tcp
ACOS(config-slb vserver-vport)# exit
ACOS(config)# slb server S2 ethernet 8
ACOS(config-real server)# port 443 tcp
ACOS(config-slb vserver-vport)# exit
ACOS(config)# slb server FW1 ethernet 2
ACOS(config-real server)# port 8080 tcp
ACOS(config-slb vserver-vport)# exit
ACOS(config)# slb server FW2 ethernet 6
ACOS(config-real server)# port 8080 tcp
ACOS(config-real server)# exit
ACOS(config)# slb service-group S1_443 tcp
ACOS(config-slb svc group)# member S1 443
ACOS(config-slb svc group)# exit
ACOS(config)# slb service-group S2_443 tcp
ACOS(config-slb svc group)# member S2 443
ACOS(config-slb svc group)# exit
ACOS(config)# slb service-group FW1_8080 tcp
ACOS(config-slb svc group)# member FW1 8080
ACOS(config-slb svc group)# exit
ACOS(config)# slb service-group FW2_8080 tcp
ACOS(config-slb svc group)# member FW2 8080
ACOS(config-slb svc group)# exit
```

9. Configure the Server SSL and Client SSL templates.

```
ACOS(config)# slb template server-ssl serverssl
ACOS(config-server ssl)# forward-proxy-enable
ACOS(config-server ssl)# exit
ACOS(config)# slb template client-ssl clientssl
ACOS(config-client ssl)# forward-proxy-ca-certificate Cert123.pem key
key123
ACOS(config-client ssl)# forward-proxy-enable
ACOS(config-client ssl)# exit
```

10. Configure the SLB virtual server for decrypting traffic client traffic and sending it to the security device.

```
ACOS(config)# slb virtual-server vs_dec1 0.0.0.0 acl 196
ACOS(config-slb vserver)# port 443 https
ACOS(config-slb vserver-vport)# service-group FW1_8080
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# template client-ssl clientssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
ACOS(config)# slb virtual-server vs_dec2 0.0.0.0 acl 198
ACOS(config-slb vserver)# port 443 https
ACOS(config-slb vserver-vport)# service-group FW2_8080
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# template client-ssl clientssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
```

11. Configure the SLB virtual server for encrypting traffic.

```
ACOS(config)# slb virtual-server vs_enc1 0.0.0.0 acl 197
ACOS(config-slb vserver)# port 8080 http
ACOS(config-slb vserver-vport)# service-group S1_443
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# template server-ssl serverssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
ACOS(config)# slb virtual-server vs_enc2 0.0.0.0 acl 199
ACOS(config-slb vserver)# port 8080 http
ACOS(config-slb vserver-vport)# service-group S2_443
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# template server-ssl serverssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# exit
```

SSLi IP-Less Deployment with Layer 3 Health Monitoring

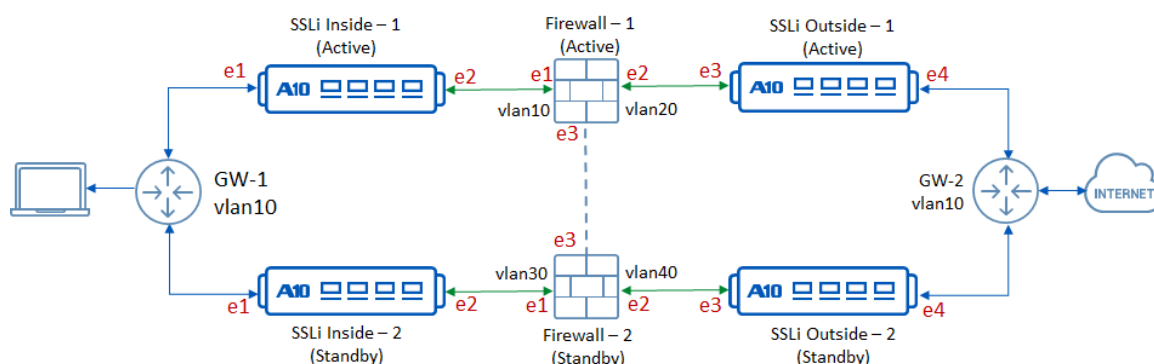
ACOS provides layer 3 health monitoring infrastructure support for virtual wire and virtual wire with SSLi (IP-less), which operates transparently in high-availability

deployments. In these deployments, two firewall devices manage the active and standby paths during failover scenarios.

The layer 3 health checks can be configured on the VLAN that needs monitoring. The health check uses ping, gratuitous ARP (GARP), and packet count methods on the target device, allowing ACOS to detect which VLAN is in active or standby mode.

This feature ensures that the paths are active and detects any failovers or changes in the network. Additionally, the health check prevents ACOS from sending the packets to the ACOS standby devices, thus avoiding network disturbances.

Figure 31 : Virtual Wire in High Availability Firewall Deployment



In the above deployment example,

- Two SSLi devices (SSLi-1 and SSLi-2) are positioned transparently inside and outside of the network. These devices are configured to handle traffic with virtual-wire connections.
- Two firewall devices (Firewall-1 (active - vlan 10 and 20) and Firewall-2 (standby- vlan 30 and 40)) are configured with VRRP-A for redundancy. This allows one firewall to act as the active path while the other is on standby.
- Two gateways (GW-1 and GW-2) are configured to manage traffic between different VLANs and network paths.

During a switch from active to standby, ACOS might continue to send packets for previously established sessions, causing traffic to route incorrectly and disturbing network traffic.

To address these scenarios, Layer 3 health checks can be configured on the VLAN that needs monitoring. The health check uses ping, gratuitous ARP (GARP), and packet count methods, allowing ACOS to detect which VLAN is in active or standby mode.

Let's say, both SSLi devices are configured with Layer 3 health checks on specific VLANs using the GARP method, which monitors the status of the network paths. The following workflow takes place:

- SSLi Inside -1 and SSLi Outside-1 detect the failover through the GARP health check method. When the GARP request is received from the target device (Firewall-1) in the defined interval and monitors the responses.
- If GARP packets are not received within the configured interval, the VLAN status transitions to standby.
- If SSLi devices detect stale sessions, the packets are silently dropped that belong to the session.

For detailed information on layer 3 health check methods, see [Network Configuration Guide](#) and [Command Line Interface Reference](#).

CLI Configuration

The example shows the Thunder SSLi-1 inside and outside configuration.

ACOS Inside (Decrypt)

1. Configure access-list, interface servers, and virtual wire endpoints:

```
ACOS(config)# access-list 198 permit ip any any ethernet 1
ACOS(config)# interface ethernet 1
ACOS(config-if:ethernet:1)# enable
ACOS(config-if:ethernet:1)# virtual-wire
ACOS(config-if:ethernet:1)# ip allow-promiscuous-vip
ACOS(config-if:ethernet:1)# exit
ACOS(config)# interface ethernet 2
ACOS(config-if:ethernet:2)# enable
ACOS(config-if:ethernet:2)# virtual-wire
ACOS(config-if:ethernet:1)# exit
ACOS(config)# virtual-wire 1
ACOS(config-virtual-wire:1)# ethernet 1 ethernet 2
ACOS(config-virtual-wire:1)# exit
```

2. Configure the layer 3 health monitor at the global configuration mode.

```
ACOS(config)# virtual-wire-health-check vlan 10 method ping
ACOS(config-virtual-wire-health-check:10)# interface ethernet 2
```

```

ACOS(config-virtual-wire-health-check:10)# nexthop-ip 10.5.10.100
nexthop-mac 021f.a000.0001
ACOS(config-virtual-wire-health-check:10)# source-ip 10.5.10.1
source-mac 000c.292f.ae34
ACOS(config-virtual-wire-health-check:10)# interval 1
ACOS(config-virtual-wire-health-check:10)# enable

```

3. Configure the SLB server and service-group components for the security device:

```

ACOS(config)# slb server fw ethernet 2
ACOS(config-real server)# port 8080 tcp
ACOS(config-real server)# port 0 tcp
ACOS(config)# exit
ACOS(config)# slb service-group sg_fw_8080 tcp
ACOS(config-slb svc group)# member fw 8080
ACOS(config-slb svc group)# exit
ACOS(config)# slb service-group sg_fw_0 tcp
ACOS(config-slb svc group)# member fw 0
ACOS(config-slb svc group)# exit

```

4. Configure the client SSL templates:

```

ACOS(config)# slb template client-ssl clientssl
ACOS(config-client ssl)# ssli-logging all
ACOS(config-client ssl)# forward-proxy-ca-certificate Cert123.pem key
key123
ACOS(config-client ssl)# forward-proxy-enable
ACOS(config-client ssl)# exit

```

- Configure the SLB virtual server for decrypting traffic client traffic and sending it to the security device.

```

ACOS(config)# slb virtual-server vs_dec1 0.0.0.0 acl 198
ACOS(config-slb vserver)# port 0 tcp
ACOS(config-slb vserver-vport)# service-group sg_fw_0
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# port 443 https
ACOS(config-slb vserver-vport)# service-group sg_fw_8080
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# template client-ssl clientssl

```

```
ACOS(config-slb vserver-vport)# no-dest-nat port-translation  
ACOS(config-slb vserver-vport)# exit
```

ACOS Outside (Encrypt)

1. Configure access-list, interface servers, and virtual wire endpoints.

```
ACOS(config)# access-list 197 permit ip any any ethernet 3  
ACOS(config)# interface ethernet 3  
ACOS(config-if:ethernet:3)# enable  
ACOS(config-if:ethernet:3)# virtual-wire  
ACOS(config-if:ethernet:3)# ip allow-promiscuous-vip  
ACOS(config-if:ethernet:3)# exit  
ACOS(config)# interface ethernet 4  
ACOS(config-if:ethernet:4)# enable  
ACOS(config-if:ethernet:4)# virtual-wire  
ACOS(config-if:ethernet:4)# exit  
ACOS(config)# virtual-wire 2  
ACOS(config-virtual-wire:2)# ethernet 3 ethernet 4  
ACOS(config-virtual-wire:2)# exit
```

2. Configure the layer 3 health monitor at the global configuration mode.

```
ACOS(config)# virtual-wire-health-check vlan 10 method ping  
ACOS(config-virtual-wire-health-check:10)# interface ethernet 4  
ACOS(config-virtual-wire-health-check:10)# nexthop-ip 10.5.10.100  
ACOS(config-virtual-wire-health-check:10)# interval 1  
ACOS(config-virtual-wire-health-check:10)# enable  
ACOS(config-virtual-wire-health-check:10)# exit
```

1. Configure the SLB server and service-group components for the security device.

```
ACOS(config)# slb server gw ethernet 4  
ACOS(config-real server)# port 0 tcp  
ACOS(config-real server)# port 443 tcp
```

```

ACOS(config-real server)# exit
ACOS(config)# slb service-group sg_gw_443 tcp
ACOS(config-slb svc group)# member gw 443
ACOS(config-slb svc group)# exit
ACOS(config)# slb service-group sg_gw_0 tcp
ACOS(config-slb svc group)# member gw 0
ACOS(config-slb svc group)# exit

```

2. Configure the server SSL templates.

```

ACOS(config)# slb template server-ssl serverssl
ACOS(config-server ssl)# forward-proxy-enable
ACOS(config-server ssl)# exit

```

3. Configure the SLB virtual server for encrypting traffic.

```

ACOS(config)# slb virtual-server vs_enc1 0.0.0.0 acl 197
ACOS(config-slb vserver)# port 0 http
ACOS(config-slb vserver-vport)# service-group sg_gw_0
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# port 8080 http
ACOS(config-slb vserver-vport)# service-group sg_gw_443
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# template server-ssl serverssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# exit

```

The example shows the Thunder SSLi-2 inside and outside configuration.

ACOS Inside (Decrypt)

1. Configure access-list, interface servers, and virtual wire endpoints:

```

ACOS(config)# access-list 198 permit ip any any ethernet 1
ACOS(config)# interface ethernet 1
ACOS(config-if:ethernet:1)# enable
ACOS(config-if:ethernet:1)# virtual-wire
ACOS(config-if:ethernet:1)# ip allow-promiscuous-vip
ACOS(config-if:ethernet:1)# exit

```

```
ACOS(config)# interface ethernet 2
ACOS(config-if:ethernet:2)# enable
ACOS(config-if:ethernet:2)# virtual-wire
ACOS(config-if:ethernet:1)# exit
ACOS(config)# virtual-wire 1
ACOS(config-virtual-wire:1)# ethernet 1 ethernet 2
ACOS(config-virtual-wire:1)# exit
```

2. Configure the layer 3 health monitor at the global configuration mode.

```
ACOS(config)# virtual-wire-health-check vlan 40 method ping
ACOS(config-virtual-wire-health-check:10)# interface ethernet 2
ACOS(config-virtual-wire-health-check:10)# nexthop-ip 10.6.10.100
ACOS(config-virtual-wire-health-check:10)# interval 1
ACOS(config-virtual-wire-health-check:10)# enable
```

3. Configure the SLB server and service-group components for the security device:

```
ACOS(config)# slb server fw ethernet 2
ACOS(config-real server)# port 8080 tcp
ACOS(config-real server)# port 0 tcp
ACOS(config)# exit
ACOS(config)# slb service-group sg_fw_8080 tcp
ACOS(config-slb svc group)# member fw 8080
ACOS(config-slb svc group)# exit
ACOS(config)# slb service-group sg_fw_0 tcp
ACOS(config-slb svc group)# member fw 0
ACOS(config-slb svc group)# exit
```

4. Configure the client SSL templates:

```
ACOS(config)# slb template client-ssl clientssl
ACOS(config-client ssl)# ssli-logging all
ACOS(config-client ssl)# forward-proxy-ca-certificate Cert123.pem key
key123
ACOS(config-client ssl)# forward-proxy-enable
ACOS(config-client ssl)# exit
```

- Configure the SLB virtual server for decrypting traffic client traffic and sending it to the security device.

```
ACOS(config)# slb virtual-server vs_dec 0.0.0.0 acl 198
```

```

ACOS(config-slb vserver)# port 0 tcp
ACOS(config-slb vserver-vport)# service-group sg_fw_0
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# port 443 https
ACOS(config-slb vserver-vport)# service-group sg_fw_8080
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# template client-ssl clientssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit

```

ACOS Outside (Encrypt)

1. Configure access-list, interface servers, and virtual wire endpoints.

```

ACOS(config)# access-list 197 permit ip any any ethernet 3
ACOS(config)# interface ethernet 3
ACOS(config-if:ethernet:3)# enable
ACOS(config-if:ethernet:3)# virtual-wire
ACOS(config-if:ethernet:3)# ip allow-promiscuous-vip
ACOS(config-if:ethernet:3)# exit
ACOS(config)# interface ethernet 4
ACOS(config-if:ethernet:4)# enable
ACOS(config-if:ethernet:4)# virtual-wire
ACOS(config-if:ethernet:4)# exit
ACOS(config)# virtual-wire 2
ACOS(config-virtual-wire:2)# ethernet 3 ethernet 4
ACOS(config-virtual-wire:2)# exit

```

2. Configure the layer 3 health monitor at the global configuration mode.

```

ACOS(config)# virtual-wire-health-check vlan 40 method ping
ACOS(config-virtual-wire-health-check:10)# interface ethernet 4
ACOS(config-virtual-wire-health-check:10)# nexthop-ip 10.5.10.100
ACOS(config-virtual-wire-health-check:10)# interval 1
ACOS(config-virtual-wire-health-check:10)# enable
ACOS(config-virtual-wire-health-check:10)# exit

```

1. Configure the SLB server and service-group components for the security device.

```

ACOS(config)# slb server gw ethernet 4
ACOS(config-real server)# port 0 tcp

```

```
ACOS(config-real server)# port 443 tcp
ACOS(config-real server)# exit
ACOS(config)# slb service-group sg_gw_443 tcp
ACOS(config-slb svc group)# member gw 443
ACOS(config-slb svc group)# exit
ACOS(config)# slb service-group sg_gw_0 tcp
ACOS(config-slb svc group)# member gw 0
ACOS(config-slb svc group)# exit
```

2. Configure the server SSL templates.

```
ACOS(config)# slb template server-ssl serverssl
ACOS(config-server ssl)# forward-proxy-enable
ACOS(config-server ssl)# exit
```

3. Configure the SLB virtual server for encrypting traffic.

```
ACOS(config)# slb virtual-server vs_enc 0.0.0.0 acl 197
ACOS(config-slb vserver)# port 0 http
ACOS(config-slb vserver-vport)# service-group sg_gw_0
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# port 8080 http
ACOS(config-slb vserver-vport)# service-group sg_gw_443
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# template server-ssl serverssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# exit
```

In the above example, the health check method ping is enabled and configured for **vlan 10** and **vlan 40**.

- The **interface** is the output interface through which the ICMP echo request (ping) will be sent.
- The nexthop IPv4 address of the target device is configured, which will be used for the ping request.
- The time interval is set to **5** seconds, which determines how frequently to check the health check entry.

As only `next-hop-ip` is configured and not the `source-ip`, ACOS will analyze the traffic matching the virtual port to gather the necessary fields and then send the ping request. If ACOS does not receive a response to two consecutive echo requests, the VLAN 10 status is marked as standby. If ACOS finds any outdated sessions, they close these sessions silently to avoid network loops.

Limitation

IPv6 addresses are unsupported and cannot be configured for the target device or interface.

AAM for SSLi Transparent Proxy

ACOS supports configuring AAM for transparent proxy for SSLi. This feature is applicable to HTTPS and HTTP sessions where AAM performs authentication and authorization checking inside the SSL tunnel. For HTTPS, if the auth-session-mode is cookie-based, the deployment requires an HTTPS server. The hostname of the HTTPS server must be configured in the `redirect-hostname` command in the auth-template. When the client browser attempts a proxy-auth against this hostname, the hostname must be resolvable at the browser side.

The following are supported for transparent proxy for SSLi:

- Single sign-on including for cross-domains
- HTTP authentication logon and form-based logon
- IP-based and cookie-based auth-session tracking mode
- Kerberos WIA authentication

For Content Security Policy (CSP), this feature provides a new command called `modify-content-security-policy` under the auth-template. When the command is enabled, ACOS checks all packets from the server. If the packet contains a CSP header (keywords: Content-Security-Policy, X-Content-Security-Policy and X-Webkit-CSP), ACOS inserts a redirect-url (from `redirect-hostname` or kerberos SPN) into the 'default-src' field. If no CSP header is found, ACOS does nothing.

The following topics are covered:

[Topology Example](#) 354

CLI Configuration	355
Decrypt_VIP Support	357
Forward Policy Support	358
HTTP Authenticate Logon Support	358

Topology Example

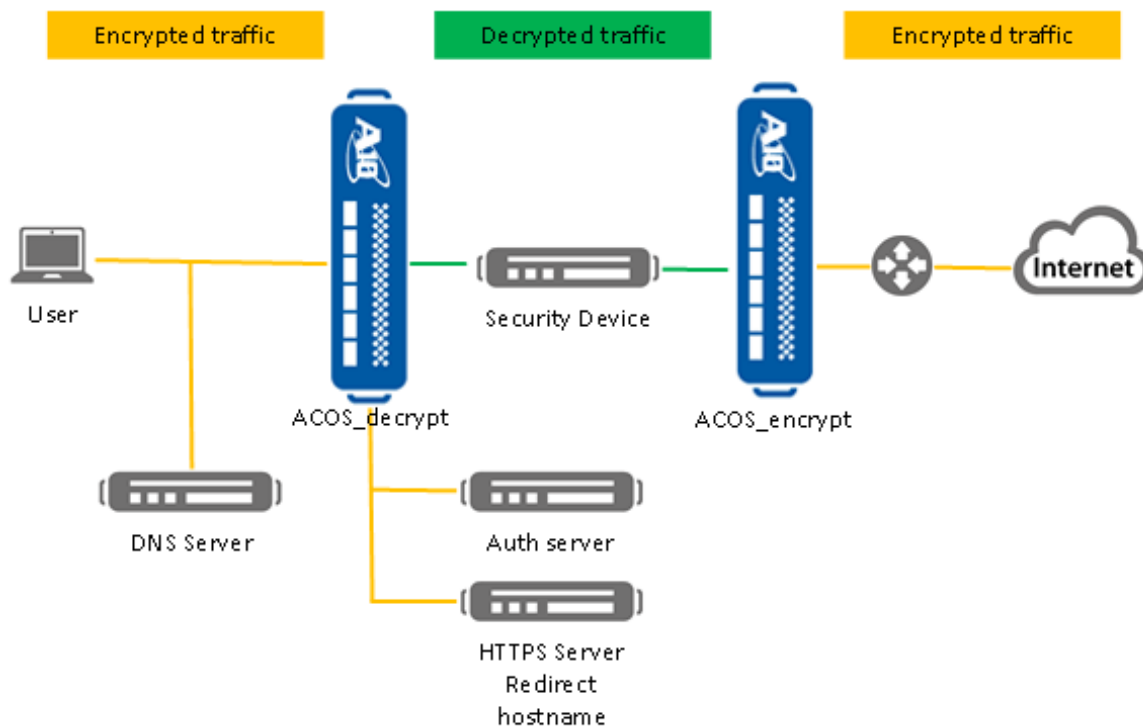
The following figure is an example topology for configuring AAM for SSLi and transparent proxy. This is a cookie-based auth-session, so there is an HTTPS server connected to ACOS_decrypt. The authorization server is LDAP-based and uses basic logon. Since the AAM authentication happens in the SSL tunnel, there are no configuration changes required for ACOS_encrypt.

NOTE: Ensure that the CA is imported to the user browser.

The expected behavior of the deployment is as follows:

- Clear the auth session.
 - Access HTTP page `www.apple.com`. User is asked for credentials.
 - Access HTTP page `www.nokia.com` which is another domain. User is not asked for credentials.
- Clear the auth session.
 - Access HTTPS page `https://www.google.com`. User is asked for credentials.
 - Access HTTPS page `https://github.com` which is another domain. User is not asked for credentials.

Figure 32 : Configuring AAM for SSLi with Transparent Proxy



CLI Configuration

1. Configure HTTP-authenticate logon with the profile name as **BASIC**.

```
ACOS_decrypt(config)# aam authentication logon http-authenticate BASIC
ACOS_decrypt(config-form-based auth logon:BASIC)# auth-method basic
enable
ACOS_decrypt(config-form-based auth logon:BASIC)# exit
```

2. Configure an authentication-server profile for the LDAP server called **LDAP_98_172**.

```
ACOS_decrypt(config)# aam authentication server ldap LDAP_98_172
ACOS_decrypt(config-ldap auth server)# host 192.168.98.172
ACOS_decrypt(config-ldap auth server)# base ou=People,dc=lex-
ldap,dc=com
ACOS_decrypt(config-ldap auth server)# admin-dn cn=Admin,dc=lex-
ldap,dc=com
```

```
ACOS_decrypt(config-ldap auth server)# admin-secret encrypted
37048xvi8uY8EIy41dsA5zwQjLjV2wDnPBCMuNXbAOc8EIy41dsA5zwQjLjV2wDn dn-
attribute uid
ACOS_decrypt(config-ldap auth server)# exit
```

3. Configure an authentication template called **BASIC_LDAP_C**. Associate the logon profile **BASIC** and LDAP server **LDAP_98_172** with the authentication template. Modify the hostname as **tp.10.com**.

```
ACOS_decrypt(config)# aam authentication template BASIC_LDAP_C
ACOS_decrypt(config-auth template:BASIC_LDAP_C)# logon BASIC
ACOS_decrypt(config-auth template:BASIC_LDAP_C)# server LDAP_98_172
ACOS_decrypt(config-auth template:BASIC_LDAP_C)# redirect-hostname
tp.a10.com
ACOS_decrypt(config-auth template:BASIC_LDAP_C)# modify-content-
security-policy
ACOS_decrypt(config-auth template:BASIC_LDAP_C)# exit
```

4. Create a AAA policy called **BASIC_LDAP_C** and associate the authentication template.

```
ACOS_decrypt(config)# aam aaa-policy BASIC_LDAP_C
ACOS_decrypt(config-aaa policy:1)# aaa-rule 10
ACOS_decrypt(config-aaa policy:1-aaa rule:10)# authentication-template
BASIC_LDAP_C
```

5. Create a virtual server called **TP_AAM**. For port 80 HTTP and port 443 HTTPS, associate aaa-policy **BASIC_LDAP_C**.

```
ACOS_decrypt(config)# slb virtual-server TP_AAM 0.0.0.0 acl 2
ACOS_decrypt(config-slb vserver)# port 80 http
ACOS_decrypt(config-slb vserver-vport)# aaa-policy BASIC_LDAP_C
ACOS_decrypt(config-slb vserver-vport)# exit
ACOS_decrypt(config-slb vserver)# port 443 https
ACOS_decrypt(config-slb vserver-vport)# aaa-policy BASIC_LDAP_C
ACOS_decrypt(config-slb vserver-vport)# exit
```

Consolidated Configuration

```
!
aam authentication logon http-authenticate BASIC
auth-method basic enable
!
```

```
!  
aam authentication server ldap LDAP_98_172  
  host 192.168.98.172  
  base ou=People,dc=lex-ldap,dc=com  
  admin-dn cn=Admin,dc=lex-ldap,dc=com  
  admin-secret encrypted  
37048xvi8uY8EIy41dsA5zwQjLjV2wDnPBCMuNXbAOc8EIy41dsA5zwQjLjV2wDn  
  dn-attribute uid  
!  
aam authentication template BASIC_LDAP_C  
  logon BASIC  
  server LDAP_98_172  
  redirect-hostname tp.a10.com  
  modify-content-security-policy  
!  
aam aaa-policy BASIC_LDAP_C  
  aaa-rule 10  
    authentication-template BASIC_LDAP_C  
!  
slb virtual-server TP_AAM 0.0.0.0 acl 2  
  port 80 http  
    aaa-policy BASIC_LDAP_C  
  port 443 https  
    aaa-policy BASIC_LDAP_C
```

Decrypt_VIP Support

If you configure ACOS SSLi with explicit proxy, you also can configure the decrypt_VIP with the AAM features described in the *Application Access Management Guide*.

When configuring AAM with an explicit proxy, the HTTP-basic and Kerberos logon methods are supported for HTTP authentication. Use the `aam authentication logon http-authenticate` command and its sub-commands to configure these methods.

Form-based and SAML authentication is also supported. Use the `aam authentication logon form-based` command to configure form-based authentication.

Forward Policy Support

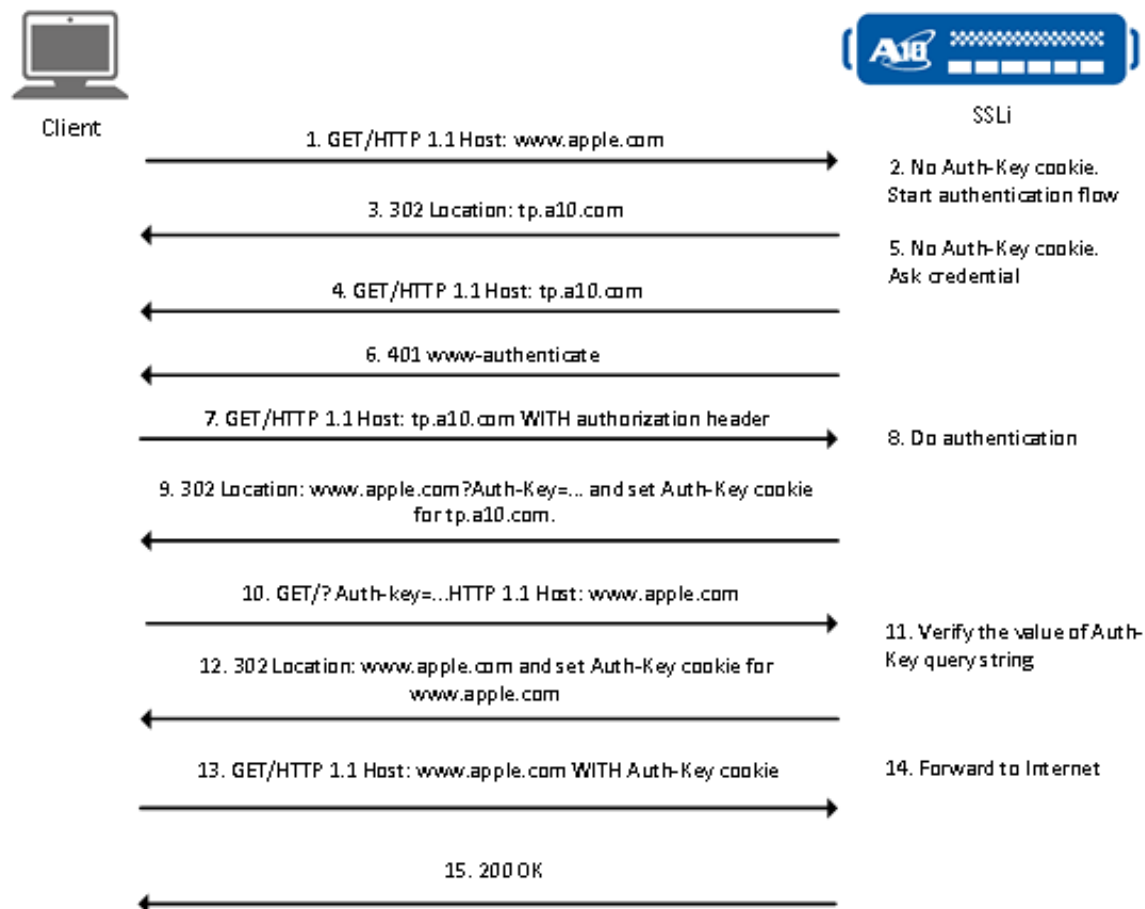
For SSLi explicit and transparent proxy, AAM authorization policy can also be configured as the forward-policy source matching criteria. Therefore, the ACOS can provide JWT and SAML authorization feature for forward-policy.

- For implementing JWT and SAML authorization, see *Application Access Management Guide*.
- For implementing forward policy, see *Explicit Proxy with AAM Policy* section in the *Application Delivery Controller Guide*.

HTTP Authenticate Logon Support

The following workflow is for the authentication for HTTP-authenticate logon. User tries to access `www.apple.com`. The redirectr hostname is `tp.a10.com`.

Figure 33 : Workflow for Authentication Flow for HTTP-authenticate Logon



The authentication template has the following configuration:

```

ACOS_decrypt(config)# aam authentication template TP-AAM
ACOS_decrypt(config-auth template:TP-AAM)#redirect-hostname tp.a10.com
ACOS_decrypt(config-auth template:TP-AAM)# exit
  
```

Transparent Proxy with Normal VIPs Associated to ACL

This section describes the transparent proxy with normal VIP associated to ACL, its features, configuration, and limitations.

The following topics are covered:

Overview

ACOS can be configured to associate normal VIPs with ACLs. The destination IP in ACL can be transformed to the VIP, and the VIP bound to an ACL. The source IP defined in the associated ACL will continue to determine if the traffic matches the specified virtual port and accordingly permit or deny the traffic. You can continue to configure wildcard VIPs associated with ACLs along with the normal VIP and ACL configuration.

Key Considerations

- The ACL associated to the VIP should not contain any destination IP.
- Both IPv4 and IPv6 addresses can be configured.
- If ACOS is unable to match the traffic with the virtual port in one VIP with ACL, it tries to match the virtual port in another VIP with ACL.
- If ACOS is unable to match the traffic with any virtual port in the VIPs, it tries to match the virtual port in the wildcard VIP.
- After creating a normal VIP with ACL, you can modify the IP address and ACL. However, you cannot remove the associated ACL after it is created.
- Multiple VIPs can use the same IP address if they are associated with ACLs. If there are multiple VIPs with ACL configured with the same IP in the system, incoming traffic will match only one of the VIP and will be rejected by other VIPs.
- The VIP associated to ACL can have the same IP address as the IP of the SLB server.
- The VIP associated to ACL cannot use the same IP as a normal VIP and subnet VIP.
- An ACL can be associated with two virtual servers, only if the two virtual servers have different VIPs.
- If a virtual port is matched by IP and L4 port but is rejected by the ACL bound in a virtual server, ACOS skips the virtual port and tries to match the next virtual port.
- The VIP associated to ACL uses the same resource pool as a normal virtual server. So, like virtual servers, the maximum number of configurable VIPs depends on the platform and system memory size and is greater than the maximum configurable number of wildcard VIPs.

Limitations

This feature has the following limitations:

- It is only supported for slow paths.
- Subnet VIPs cannot be associated with an ACL.
- The matching order for VIPs with ACL having the same IP address is not guaranteed. The matching does not follow the alphabetic order of ACL, the way it is done in wildcard VIPs.

Virtual Wire Configuration

This section describes the guidelines for configuring a normal VIP associated to an ACL in transparent proxy deployment in virtual wire configuration. For the basic configuration of normal VIP associated to an ACL in transparent proxy deployment, see *Application Delivery Controller Guide*.

1. Enable ethernet interfaces.

```
ACOS(config)# interface ethernet 1
ACOS(config-if:ethernet:1)# enable
ACOS(config-if:ethernet:1)# virtual-wire
ACOS(config-if:ethernet:1)# ip allow-promiscuous-vip
ACOS(config-if:ethernet:1)# exit
ACOS(config)# interface ethernet 2
ACOS(config-if:ethernet:2)# enable
ACOS(config-if:ethernet:2)# virtual-wire
ACOS(config-virtual-wire)# virtual-wire 1
ACOS(config-virtual-wire:1)# ethernet 1 ethernet 2
....
ACOS(config)# interface ethernet 6
ACOS(config-if:ethernet:6)# enable
ACOS(config-if:ethernet:6)# virtual-wire
ACOS(config-if:ethernet:6)# ip allow-promiscuous-vip
ACOS(config-if:ethernet:6)# exit
ACOS(config)# interface ethernet 7
ACOS(config-if:ethernet:7)# enable
ACOS(config-if:ethernet:7)# virtual-wire
ACOS(config-if:ethernet:7)# virtual-wire 2
```

```
ACOS(config-virtual-wire:7)# ethernet 6 ethernet 7
```

2. Configure ACLs for permitting or denying traffic from specific source IP addresses.

```
ACOS(config)#ip access-list myacl1
ACOS(config-ext-access-list:myacl1)# permit ip 152.145.30.5 0.0.0.0 any
ACOS(config-ext-access-list:myacl1)# deny ip 152.145.30.6 0.0.0.0 any
ACOS(config-ext-access-list:myacl1)# exit

ACOS(config)#ip access-list myacl2
ACOS(config-ext-access-list:myacl2)# permit ip 152.145.30.5 0.0.0.0 any
ACOS(config-ext-access-list:myacl2)# deny ip 152.145.30.6 0.0.0.0 any
ACOS(config-ext-access-list:myacl2)# exit
```

3. Configure real servers.

```
ACOS(config)# slb server fw 152.145.30.1
ACOS(config-real server)# port 80 tcp
ACOS(config-real server-node port)# exit
ACOS(config-real server)# port 8080 tcp
ACOS(config-real server-node port)# exit

ACOS(config)# slb server gw 152.145.30.1
ACOS(config-real server)# port 0 tcp
ACOS(config-real server-node port)# exit
ACOS(config-real server)# port 443 tcp
ACOS(config-real server-node port)# exit
```

4. Configure service groups.

```
ACOS(config)# slb service-group fw1 tcp
ACOS(config-slb svc group)# member fw 80
ACOS(config-slb svc group-member:80)# exit

ACOS(config)# slb service-group fw2 tcp
ACOS(config-slb svc group)# member fw 8080
ACOS(config-slb svc group-member:8080)# exit

ACOS(config)# slb service-group gw1 tcp
ACOS(config-slb svc group)# member gw 0
ACOS(config-slb svc group-member:0)# exit
```

```
ACOS(config)# slb service-group gw2 tcp
ACOS(config-slb svc group)# member gw 443
ACOS(config-slb svc group-member:443)# exit
```

5. Configure a client SSL template and a server SSL template.

```
ACOS(config)# slb template client-ssl c1
ACOS(config-client ssl)# forward-proxy-ca-certificate mycert key mycert
ACOS(config-client ssl)# forward-proxy-enable
ACOS(config-client ssl)# exit
ACOS(config)# slb template server-ssl s1
ACOS(config-server ssl)# forward-proxy-enable
ACOS(config-server ssl)# exit
```

6. Configure virtual servers and virtual ports. Bind the client SSL template and server SSL template to the virtual server ports. Associate the virtual server with an IP-based VIP with ACL.

```
ACOS(config)# slb virtual-server inside-vip 152.145.30.1 acl name
myacl1
ACOS(config-slb vserver)# port 80 tcp
ACOS(config-slb vserver-vport)# service-group fw1
ACOS(config-slb vserver-vport)# no-dest-nat
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# port 443 https
ACOS(config-slb vserver-vport)# service-group fw2
ACOS(config-slb vserver-vport)# template client-ssl c1
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver)# exit

ACOS(config)# slb virtual-server outside-acl-vip 152.145.30.1 acl name
myacl2
ACOS(config-slb vserver)# port 0 tcp
ACOS(config-slb vserver-vport)# service-group gw1
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# port 8080 http
ACOS(config-slb vserver-vport)# service-group gw2
ACOS(config-slb vserver-vport)# template server-ssl s1
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver)# exit
```

Virtual Wire Consolidated Configuration Example

This section describes the consolidated configuration of IP-based VIPs associated to ACLs in virtual wire environment.

```
ACOS(config)#show running-config
!Current configuration: 2015 bytes
!Configuration last updated at 10:09:08 GMT Wed Dec 6 2023
!Configuration last saved at 10:09:12 GMT Wed Dec 6 2023
!64-bit Advanced Core OS (ACOS) version 6.0.3
!
ip access-list myacl1
 4 permit ip 152.145.30.5 0.0.0.0 any ethernet 1
 8 deny ip 152.145.30.6 0.0.0.0 any
!
ip access-list myacl2
 4 permit ip 152.145.30.5 0.0.0.0 any ethernet 7
 8 deny ip 152.145.30.6 0.0.0.0 any
!
interface ethernet 1
 enable
 virtual-wire
 ip allow-promiscuous-vip
!
interface ethernet 2
 enable
 virtual-wire
!
 virtual-wire 1
   ethernet 1 ethernet 2
....
!
interface ethernet 6
 enable
 virtual-wire
 ip allow-promiscuous-vip
!
interface ethernet 7
 enable
 virtual-wire
!
```

```
virtual-wire 2
  ethernet 6 ethernet 7
!
slb template server-ssl s1
  forward-proxy-enable
!
slb server fw ethernet 2
  port 80 tcp
  port 8080 tcp
!
slb server gw ethernet 7
  port 0 tcp
  port 443 tcp
!
slb service-group fw1 tcp
  member fw 80
!
slb service-group fw2 tcp
  member fw 8080
!
slb service-group gw1 tcp
  member gw 0
!
slb service-group gw2 tcp
  member gw 443
!
slb template client-ssl c1
  forward-proxy-ca-certificate mycert key mycert
  forward-proxy-enable
!
slb virtual-server inside-vip 152.145.30.1 acl name myacl1
  port 80 tcp
    service-group fw1
    no-dest-nat
  port 443 https
    service-group fw2
    template client-ssl c1
    no-dest-nat port-translation
!
slb virtual-server outside-acl-vip 152.145.30.1 acl name myacl2
```

```
port 0 http
  service-group gw1
  no-dest-nat port-translation
port 8080 http
  service-group gw2
  template server-ssl s1
  no-dest-nat port-translation
!
```

Non-standard Port Support

ACOS provides full non-standard port support without using SNAT to preserve the original client IP address. This feature benefits in various scenarios, for example,

- Collect statistics and perform analysis based on client IP address.
- Use IP-address-based filters such as security groups or security group rules on Application Load Balancers to filter traffic.

To implement this feature, you must enable the 'port-translation-only' option in the SLB policy template under forward-policy actions: 'forward-to-internet', 'forward-to-service-group', and 'forward-to-proxy'. The forward-policy flow will select a unique internal NAT pool and only perform the source port translation.

This feature is supported in the following environments:

- Explicit proxy and SSLi/Dynamic Port SSLi
- Transparent proxy and SSLi/Dynamic Port SSLi

CLI Configuration

- To enable the feature, create an SLB policy template with forward-policy action and then bind it to inside virtual server/partition.

```
ACOS(config)# slb template policy p1
ACOS(config-policy)# forward-policy
ACOS(config-policy-forward-policy)# action aaa
ACOS(config-policy-forward-policy-action)# forward-to-internet http snat port-translation-only
```

```
ACOS(config-policy-forward-policy-action)# forward-to-service-group http snat
port-translation-only
ACOS(config-policy-forward-policy-action)# forward-to-proxy http support-cert-
fetch snat port-translation-only bypass
```

NOTE: The 'port-translation-only' command and 'NAT pool' config are mutually exclusive. When the 'port-translation-only' option is set, only source port translation will be done.

- To view the source port translation only NAT pool instance statistics in current partition, use the show command:

```
ACOS# show slb forward-policy-source-port-translation-only-stats
```

Group	Port	Usage	Total	Used	Total	Freed	Failed
IPv4	0		0		0		0
IPv6	0		0		0		0

GUI Configuration

- To enable the **Port Translation Only** feature, navigate to **Security > Forward Proxy > Template > Policy > Create**.
- To view the source port translation statistics, navigate to **Security > Forward Proxy > Statistics > Policy Source Port Translation Only** tab.

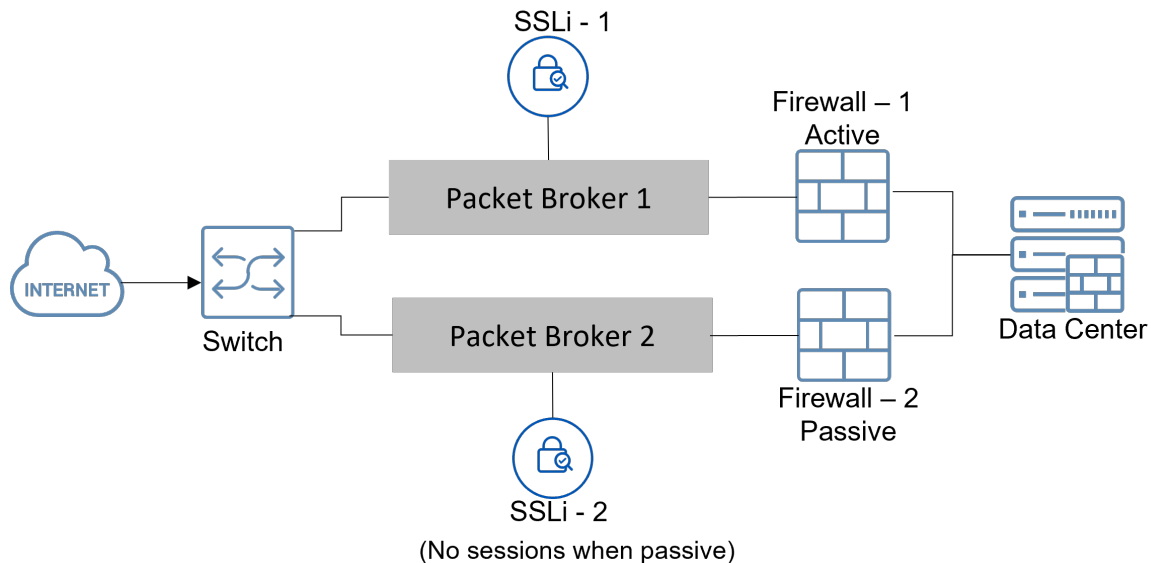
Terminating SSLi Session Silently

When SSLi terminates idle sessions, it sends out RST and FIN packets in both directions, i.e., to the client and the server. This ensures that all sessions between clients, servers, and ACOS are always gracefully terminated. However, this behavior can be problematic in pure Layer-2 SSLi deployments with redundant paths, when the upstream and downstream networking devices, such as firewalls, govern the active-standby paths. See [Figure 34](#).

In such deployments, the upstream firewall selects an active path, and sessions are established on the active SSLi appliance. When a failover occurs to the standby path, gratuitous ARPs are sent out, and sessions start re-establishing on the standby SSLi appliance. However, in due time, the previously active SSLi appliance sends RSTs and

FINs to terminate stale sessions, but this disrupts the firewall's MAC address tables, resulting in network loops.

Figure 34 : Layer 2 SSLi Deployment



To address this behavior, ACOS now supports silently terminating stale SSLi sessions so that the passive SSLi appliance does not send out any RSTs or FINs.

This feature supports both HTTP/1.1 and HTTP/2 connections.

NOTE: Enabling this feature may cause sessions to stall on either the client or the server side.

CLI Configuration

Make sure to enable layer 7 request accounting using the `enable-17-req-acct` command.

NOTE: When using the least-request load-balancing method in a service group, layer 7 request accounting is automatically enabled for the service group's members, and the virtual service ports that are bound to the service group's members.

To enable the feature, use the `ssli-silent-termination-enable` command (per partition).


```
ACOS(config)# slb common  
ACOS(config-common)# enable-l7-req-acct  
ACOS(config-common)# ssli-silent-termination-enable
```

When this command is enabled and a client-SSL or server-SSL template is bound to the virtual port, the RST or FIN packets are not sent when the `idle-timeout`, `close-notify`, or `clear session` commands are triggered.

GUI Configuration

To enable the option, use the 'Enable SSLi Silent Termination' option under **ADC > SLB > Global**.

Bypass, Inspect, and Exception

This section describes how to configure outbound SSLi for static port type STARTTLS.

The following topics are covered:

Overview	371
Priority of Rules	372
SNI and Server Certificate Based Inspection	375
Convert an SNI List to an AC Class List	376
User Name and Group Name Based Bypass	377
HTTPS Traffic Bypass	378
CLI Configuration	378
GUI Configuration	382
Client Authentication Bypass	386
IP Based Bypass	393
SSLi Bypass for "no shared cipher" Error	398
Consolidated Configuration	399

Overview

ACOS enables configuring of rules that determine if a packet is to be bypassed or inspected based on the configured criteria by using the `forward-proxy-bypass` command or as configured in the **Policies** tab of the SSLi services. The exception class list is used to decide if a packet passing through an SSLi solution is to be inspected even if `forward-proxy-bypass` is configured.

For example, a rule can be configured to bypass inspection of all financial services. However, using an `exception-class-list` option, it is possible to inspect packets from specific financial services. Additionally, ACOS supports client authentication bypass that requires configuring a list of server names that bypass SSLi forward proxy processing when CAC is requested by the server.

ACOS supports the following criteria for taking inspection decisions:

- Server Name Indication (SNI)
- Certificate Subject Alternative Name (SAN)
- Certificate Subject
- Certificate Issuer

ACOS supports the following criteria for taking bypass decisions:

- SNI
- SAN
- Certificate Subject
- Certificate Issuer
- User Name
- AD Group
- Web Category (requires license)
- Web Reputation (uses Web Category license)

NOTE: If Bypass Decrypt is enabled, exception lists can also be configured so that ACOS is forced to inspect specific packets.

Priority of Rules

There are three ways you can apply rules in ACOS that specify which server connections bypass ACOS SSLi services or which ones are intercepted. The rules and/or class lists are bound to a client SSL template which in turn is bound to a virtual router port.

- Add each rule directly
- Create an Aho-Corasick (AC) class list containing the matching rules
- Import an AC class list.

Both ACOS CLI and GUI are supported for creating these rules.

The following match options are used by the rules that you configure:

- Equals—Matches only if the value completely matches the specified string.
- Contains—Matches if the specified string appears anywhere within the value.

These match options are always applied in the order shown, regardless of the order in which the rules appear in the configuration. If a template has more than one rule with the same match option (equals, starts-with, contains, or ends-with) and a value matches on more than one of them, the most-specific match is always used.

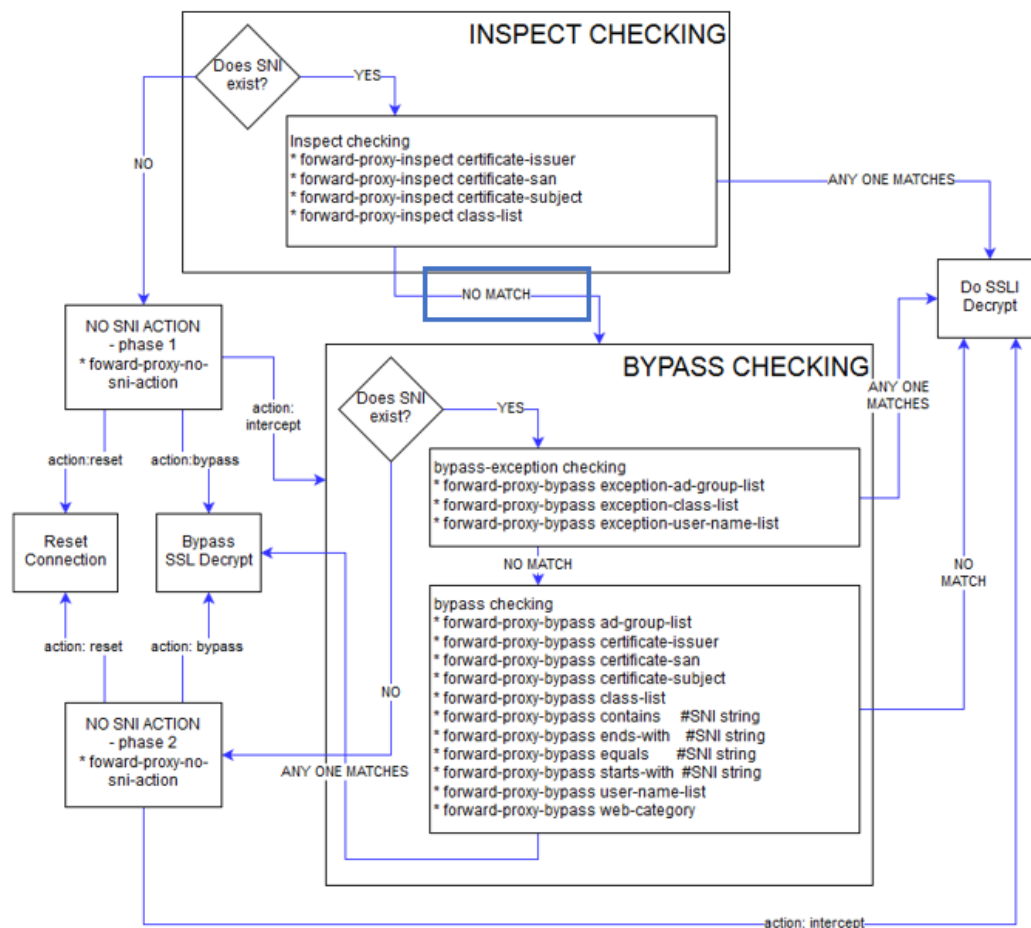
NOTE: When one string matches multiple rules, the first matched string wins. Users expecting multiple rule hit should be aware of this behavior and revise their class-list as needed.

By default, matching is case sensitive. For example, the forward-proxy-bypass contains aa rule searches for matches on SNI strings that contain “aa” but not on strings that contain “AA”. You can also enable or disable case-sensitive matching. In this case, the rule shown above matches SNI strings that contain any of the following: “aa”, “AA”, “aA”, or “Aa”. You can disable case sensitivity on a template-wide basis. The setting applies to all match rules in the template.

At a top level, the priority of rules is as follows:

1. aFlex SSLi commands
2. forward-proxy-no-sni-action
3. forward-proxy-inspect commands
4. forward-proxy-no-sni-action
5. forward-proxy-bypass commands

Figure 35 : Hierarchy of SSLi Rules

**NOTE:**

No Match decision box: If there is no match on Inspect Checking, SSLi checks if there is bypass configuration available. If there is no bypass configuration, action is bypass for bypass SSL decrypt. If there is bypass configuration available, there is bypass checking done as shown in the figure.

The following is the priority of rules configured for SSLi.

- If no SNI is configured, **forward-proxy-no-sni-action** is run. For intercept action, the current decision for the checkpoint is to inspect and the check continues. For bypass action, the final decision is bypass, and for reset action, the final decision is reset.
- If the SNI inspection class-list is configured but not matched, the final decision is bypass.
- If forward-proxy-bypass exception-user-name-list or exception-ad-group-list is configured and matched, the final decision is inspect.
- If forward-proxy-bypass user-name-list or ad-group-list is configured and matched, the final decision is bypass.
- If the SNI bypass strings (contains/starts-with/equals/ends-with) are configured and matched, the final decision is bypass.
- If the SNI bypass exception class list is configured and matched, the final decision is inspect.
- If the SNI bypass class-list is configured and matched, the final decision is bypass.
- If web category bypass is configured and matched, the final decision is bypass.
- Else, the decision is inspect for now and continue to perform the remaining checks.

Next, SNI URL filtering is checked as follows:

- If intercepted-sni-enable is not configured (the option is disabled by default), SNI URL filtering is skipped.
- If bypass-sni-disable is configured for bypassed URL, SNI URL filtering is continued.
- In the event that there is no SNI and if no_sni_allow is not configured, the connection is dropped.
- If enable-san is configured and there is a match, the server certificate is fetched even for a bypass decision.
- SNI URL filtering is continued, and if the class-list is matched, the configured action is run.

Next, server certificate checkpoint is run as follows:

- If the certificate subject/issuer/SAN inspect class-list is configured but not matched, then the final decision is bypass.
- If forward-proxy-bypass exception-user-name-list/exception-ad-group-list is configured and matched, the final decision is inspect.
- If forward-proxy-bypass user-name-list/ad-group-list is configured and matched, the final decision is bypass.
- If the certificate subject/issuer/SAN bypass strings (contains/starts-with/equals/ends-with) are configured and matched, the final decision is bypass.
- If the certificate subject/issuer/SAN bypass exception class list is configured and matched, the final decision is inspect.
- If the certificate subject/issuer/SAN bypass class-list is configured and matched, the final decision is bypass.
- Else, the decision is inspect.

Next, SAN URL filtering is checked as follows:

- If enable-san is not configured, SAN URL filtering is skipped.
- If enable-san is configured, check if intercepted-san-enable is configured for intercepted URL. If the option is not enabled (disabled by default), SAN URL filtering is not continued.
- Check if bypassed-san-disable is configured for bypassed URL. If the option is disabled (enabled by default), SAN URL filtering is not continued.
- In NO SAN case, if no_san_allow is not configured, the connection is dropped.
- Else, SAN URL filtering is continued, the class-list is matched against certificate subject/issuer/SAN and the rule action with the highest priority is run.

SNI and Server Certificate Based Inspection

ACOS supports inspection, bypass, and exception lists that include elements such as IP addresses, SNIs, and matching certificate subject or issuer. Unless this new option is configured, by default, the SNI in the client-hello message is used for deciding bypass or inspection.

Server Name Indication (SNI) is an extension of the TLS protocol and indicates the hostname that is being contacted by the browser at the beginning of the SSL handshake. SNI enables multiple secure websites to be served off the same IP address without requiring all those sites to use the same certificate. In an SSL Insight deployment, SNI support allows multiple self-signed certificates to be used. In SSLi deployments, you can map each certificate to the domain name of an outside resource that is being accessed by clients.

Subject Alternative Name (SAN) certificates can secure a number of fully qualified domain names with a single certificate. The SAN field enables you to specify additional host names such as sites, IP addresses, common names, and so on, to be protected by a single SSL Certificate. SAN Certificates allow you to secure a primary domain and then add additional domains to the subject alternative name field of the certificate.

Convert an SNI List to an AC Class List

The class lists used in the SSLi policies must conform to the A10 Aho-Corasick (AC) implementation. The `class-list list-name ac` command combined with the `contains`, `ends-with`, `equals`, and `starts-with` sub-commands can create the required list, but you must enter each SNI individually.

To convert a newline-delimited text SNI list to an AC class list for SSLi bypass, use the `import class-list-convert filename class-list-type ac` command.

The file `mySNIs.txt` is a newline delimited list of domain names. Its contents are as follows:

```
www.armardo.com
www.pickature.com
mail.ispgen.com
```

The conversion procedure takes the following steps:

1. Enter the following command in global configuration mode:

```
import class-list-convert mySNIs.txt class-list-type ac
scp://username@10.10.10.224/home/username/test_import
```

2. Verify the converted list file. Use the `show class-list class-list-name debug`

command:

```
AX5100# show class-list mySNIs.txt debug
```

```
Name:                name
Total String:        2
Total hash chain:    0
Total trie node:     0
Reference count:     0
File size:           N/A
File date:           N/A
Content:
  equals mail.ispgen.com
  equals www.pickature.com
  equals www.armardo.com
File content:
class-list class-list1 ac file

; AC (Total: 3)
equals mail.ispgen.com
equals www.pickature.com
equals www.armardo.com
```

3. Use a text editor to edit the class-list as required by your network. For example, you might wish to alter the first domain in the list:

```
A10 Aho-Corasick Class-List
ends-with armardo.com
equals www.pickature.com
equals mail.ispgen.com
```

User Name and Group Name Based Bypass

SSLi traffic can be bypassed based on user name and group name. For example, SSLi can bypass all traffic from users who belong to a specific group such as **optout**. SSLi can also bypass traffic for specific users. This feature leverages AAM, or inline authentication and challenges the user for credentials if no IP information is known.

NOTE: Only supported AAM authentication methods for HTTP are enabled for this feature.

To enable the feature, you must bind an AAM authentication template which contains logon on virtual port to collect the user names. For user-group based bypassing, enable AAM authorization for retrieving group information for authorization service. The authorization server must be an LDAP server which supports the memberOf attribute.

Since the username and group name is retrieved from the AAM module, the actual matching procedure is processed in the AAM module (after authentication and authorization pass). The results are marked in the authentication-session and the SSL module makes a bypass decision according to the results in the authentication session.

To do group matching, authorization must be configured to retrieve group membership information.

NOTE: This feature is not applicable to reverse proxy scenarios.

HTTPS Traffic Bypass

Some known internal domains cannot be resolved by Thunder. For these internal domains, the traffic should bypass SSLi decrypt and be forwarded to the upstream proxy.

When the interception for explicit proxy traffic is enabled, the traffic to specific domains to be chained to group proxy server through local proxy gateway cannot be established. SSLi cannot resolve DNS for requested destination and blocks the traffic.

Use 'bypass' option added under 'action' object under 'policy template' to send all the https traffic to upstream proxy directly. For more information, see *Command Line Interface Reference*.

CLI Configuration

Use the **forward-proxy-bypass** command to create rules for SSLi bypass, inspection, and exceptions.

In this example, assume that ACOS SSLi is configured and that the client-facing VIP on the ACOS decrypt device and the client SSL template are configured as follows:

```

ACOS-Decrypt# show running-config slb virtual-server
!Section configuration: 722 bytes
!
slb virtual-server Decrypt_VIP 0.0.0.0 acl 100
  port 0 tcp
    service-group ALL_TCP_SG
    no-dest-nat
  port 0 udp
    service-group ALL_UDP_SG
    no-dest-nat
  port 0 others
    service-group ALL_UDP_SG
    no-dest-nat
  port 443 https
    service-group FW1_Inspect_SG
    template client-ssl SSLInsight_ClientSide
    no-dest-nat
!
ACOS-Decrypt# show running-config slb template client-ssl
!Section configuration: 330 bytes
!
slb template client-ssl SSLInsight_ClientSide
  forward-proxy-ca-certificate Cert123.pem key key123
  forward-proxy-enable
!

```

1. Enter the configuration mode for the SSL client template named `SSLInsight_ClientSide`:

```

ACOS_decrypt(config)# slb template client-ssl SSLInsight_ClientSide
ACOS_decrypt(config-client ssl)#

```

2. The `forward-proxy-bypass` CLI command configures the SNI match and case rules and/or class-lists that determine whether or not a client is enabled for client-authentication bypass. This section describes adding SNI match rules:

Use the `forward-proxy-bypass` command to enter the SNI match and case rules as needed to specify which servers bypass ACOS SSLi

```

ACOS_decrypt(config-client ssl)# forward-proxy-bypass contains
jsmith.com

```

```
ACOS_decrypt(config-client ssl)# forward-proxy-bypass contains
EnterpriseABC.com
ACOS_decrypt(config-client ssl)# forward-proxy-bypass equals
UofKgmc.edu/admissions
ACOS_decrypt(config-client ssl)# forward-proxy-bypass case-insensitive
```

3. Commit the changes to ACOS memory.

```
ACOS_decrypt(config-client ssl)# write memory
```

4. Enter the configuration mode for the “Decrypt_VIP” and bind the modified SSL client template to the virtual port “port 443 https:”

```
ACOS_decrypt(config)# slb virtual-server Decrypt_VIP 0.0.0.0 acl 100
ACOS_decrypt(config-slb vserver)# port 443 https
ACOS_decrypt(config-slb vserver-vport)# template client-ssl SSLInsight_
ClientSide
ACOS_decrypt(config-slb vserver-vport)#
```

5. Commit the changes to ACOS memory.

```
ACOS_decrypt(config-slb vserver-vport)# write memory
```

Creating a Class List

Use the **class-list** command with the **ac** option to create a class list in ACOS CLI. Assume that the VIP and SSL Client template are configured on ACOS decrypt.

1. To create a class list, use the **class-list** command with the **ac** option.

The **class-list** command creates a class list and gives it a name. The **file** option saves the list as a file that you can export. Without this option, the class list entries are saved in the configuration file instead. The **ac** option is required. This specifies that the list type is Aho-Corasick.

```
ACOS_decrypt# configure
ACOS_decrypt(config)# class-list bypassed-servers-CL ac
ACOS_decrypt(config-class list)# contains jsmith.com
ACOS_decrypt(config-class list)# contains EnterpriseABC.com
ACOS_decrypt(config-class list)# equals UofKgmc.edu/admissions
```

2. Bind the new class list to the SSL client template:

```
ACOS_decrypt# configure
ACOS_decrypt(config)# slb template client-ssl SSLInsight_ClientSide
ACOS_decrypt(config-client ssl)# forward-proxy-bypass class-list
bypassed-servers-CL
```

3. Bind the modified SSL client template the port 443 https of the VIP:

```
ACOS_decrypt(config)# slb virtual-server Decrypt_VIP 0.0.0.0 acl 100
ACOS_decrypt(config-slb vserver)# port 443 https
ACOS_decrypt(config-slb vserver-vport)# template client-ssl SSLInsight_
ClientSide
ACOS_decrypt(config-slb vserver-vport)#
```

4. Commit the changes to ACOS memory.

```
ACOS_decrypt(config-slb vserver-vport)# write memory
```

Importing a Class List

Use the `import class-list` command to import a class list.

Assume that the VIP and SSL Client template are configured on the ACOS Decrypt zone.

The following example shows the importing of a class list file named `CL.tgz`. The imported class list is given the name `bypassed-servers-CL` which identifies it in ACOS commands. The URL where the file is located is `//192.168.20.161`, and the file transfer protocol is `scp`.

```
ACOS_decrypt# import class-list bypassed-servers-CL
scp://192.168.20.161/CL.tgz
```

Bind the imported class list to the SSL client template:

```
ACOS_decrypt# configure
ACOS_decrypt(config)# slb template client-ssl SSLInsight_ClientSide
ACOS_decrypt(config-client ssl)# forward-proxy-bypass class-list CL.tgz
```

Bind the modified SSL client template the port 443 https of the VIP:

```
ACOS_decrypt(config)# slb virtual-server Decrypt_VIP 0.0.0.0 acl 100
ACOS_decrypt(config-slb vserver)# port 443 https
ACOS_decrypt(config-slb vserver-vport)# template client-ssl SSLInsight_
ClientSide
```

```
ACOS_decrypt(config-slb vserver-vport)#
```

Commit the changes to ACOS memory.

```
ACOS_decrypt(config-slb vserver-vport)# write memory
```

The **forward-proxy-bypass class-list** command bypasses SSLi when the SNI of the outside server matches based on the specified class list or class-lists. When enabled by the **multi-class-list** command option, you can enter the names of up to 16 file-type class lists for each **slb template client-ssl** instance. If not enabled by the **multi-class-list** command option, you can enter only one class list name.

```
ACOS_decrypt(config)# slb template client-ssl SSLInsight_ClientSide
ACOS_decrypt(config-client ssl)# forward-proxy-bypass class-list multi-
class-list my-class-list-name1
ACOS_decrypt(config-client ssl)# forward-proxy-bypass class-list multi-
class-list my-class-list-name2
```

Showing the System Resource Usage

Use the **show system resource-usage** command to check the AC class-list entry count and the remaining space available.

```
ACOS# show system resource-usage
```

Resource	Current	Default	Minimum	Maximum
-----	-----	-----	-----	-----
14-session-count	67108864	67108864	16777216	
134217728				
class-list-ipv6-addr-count	4096000	4096000	4096000	8192000
class-list-ac-entry-count	3072000	3072000	3072000	6144000
auth-portal-html-file-size	20	20	4	120
auth-portal-image-file-size	6	6	1	80
max-aflex-file-size	32768	32768	16384	262144
aflex-table-entry-count	102400	102400	102400	10485760

GUI Configuration

You can enter match rules directly, you can create an AC class list, or you can import an AC class list for binding to the client SSL template.

1. Navigate to **Security > SSLi > Templates** and edit your client ssl template (such as `SSLi_vip_001_client_ssl`).
2. In the **Update Client SSL Template** window, click the **Policies** tab.
3. To create inspection rules, select any or a combination of the following options:
 - Inspect if SNI Matches Class List
 - Inspect if Certificate SAN Matches Class List
 - Inspect if Certificate Subject Matches Class List
 - Inspect if Certificate Issue Matches Class List
4. For no SNI, configure the **Forward Proxy No SNI Action** field to either intercept, bypass or drop the packet.
5. For each Inspect field, three options are available, select one:
 - Select from the drop-down
 - Create a class list
 - Import a class list
6. For Bypass Decrypt, select a **Condition** from the drop-down.
7. Select a **Value** and click **Apply**.
8. To add multiple rules, click **Add** as needed.
9. For creating exceptions to the SSLi bypass decrypt rules, the following options are available:
 - Exceptions if SNI Matches Class List
 - Exceptions if User Name Matches Class List
 - Exceptions if AD Group Matches Class List
 - Exceptions if Certificate Subject Matches Class List
 - Exceptions if Certificate Issuer Matches Class List
10. For each Exception field, three options are available, select one:
 - Select from the drop-down
 - Create a class list

- Import a class list

Creating a Class List

Configure an AC class list to add to the SSLi inspection, bypass, or exception lists.

The procedure bellow add an AC class list for the **Bypass Decrypt** option. You can perform the similar steps for creating AC class lists for other fields in the SSLi **Policies** tab.

To create an AC class list for the **Bypass Decrypt** option, perform the following steps:

1. Navigate to **Security > SSLi > Templates** and edit your client ssl template (such as `SSLi_vip_001_client_ssl`).
2. In the Update Client SSL Template window, click the **Policies** tab.
3. For **Bypass Decrypt**, click **Add** and then click a condition from the drop-down. Since the procedure is for adding class lists, select **SNI Match Class List**.
4. For **Value**, click the **+**
5. In the **Name** field, enter a name.
6. To store the list as a file, select **Store as a file**.
7. For AC, select an option from the drop-down list:
 - **Contains**
 - **Ends with**
 - **Starts with**
 - **Equals**
8. Type the key that you wish to match.
9. Click the save icon.
10. To add another item to the class list, click **Add**.
11. Repeat step above steps for additional ACs.
12. Click **OK**.
13. Click **Apply** on the main page to add the condition.

Importing a Class List

SSLi supports importing an AC class list for configuring the SSLi bypass, inspect, and exception lists options.

The procedure bellow add an AC class list for the **Bypass Decrypt** option. You can perform the similar steps for importing AC class lists for other fields in the SSLi **Policies** tab.

To import an AC class list, perform the following steps:

1. Navigate to **Security > SSLi > Templates** and edit your client ssl template (such as `SSLi_vip_001_client_ssl`).
2. In the Update Client SSL Template window, click the **Policies** tab.
3. For Bypass Decrypt, click **Add**.
4. Expand the Condition section and select **SNI Match Class List** (an example).
5. For Value, click the **Import** button.
6. Click whether the class list is **Local** or **Remote**.
7. Enter the class list **Name**.
8. **Browse** to the location if the class list is **Local**, and skip to step 7.
9. If the class list is **Remote**,
10. Click whether or not to **Use Mgmt Port**.
11. Select the file import **Protocol**.
12. Enter the **Host** name.
13. Enter the URL **Location**.
14. If you selected the FTP **Protocol**, enter the protocol port used for FTP, the **User** name, and the **Password**.
15. If you selected the SCP or SFTP **Protocol**, enter the **User** name, and the **Password**.
16. Click **OK**.
17. Click **Apply** on the main page to add the condition.
18. Either add your newly imported class list to an existing template, or create a new template and then add your newly imported class list.

Client Authentication Bypass

Some HTTPS servers might require client certificate authentication (CAC/PKI) when the server authenticates incoming requests based on the certificate in the client’s certificate store. If the ACOS SSLi configuration lacks the necessary client certificate and key information, and if the ACOS SSLi is not enabled for client authentication bypass, CAC fails when requested by the server.

This section describes how to configure a list of server names that bypass SSLi forward proxy processing when CAC is requested by the server. The list is configured in the SSL client template.

The following topics are covered:

- [WorkFlow](#) 387
- [CLI Configuration](#) 388
- [GUI Configuration](#) 388
- [Consolidated Configuration](#) 389
- [Troubleshooting](#) 392

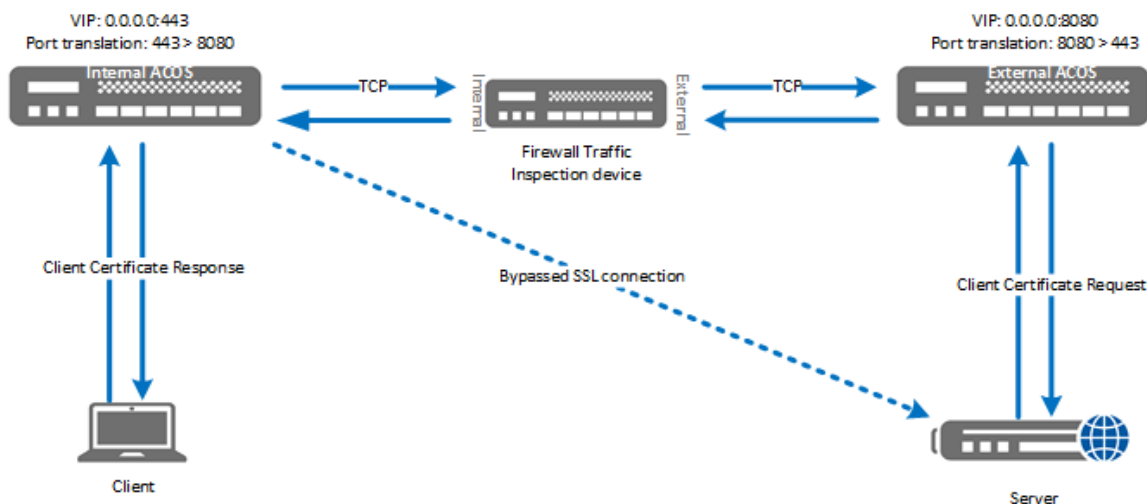


Workflow

[Figure 36](#) shows how client authentication bypass works.

1. After the Inside ACOS device receives the client hello message from the client, the device checks whether the remote server's certificate is saved in the cache.
2. If the certificate has not been saved, the Inside ACOS device starts a server SSL connection to the remote server to retrieve the certificate.
3. The Inside ACOS device also detects whether the remote server requires client certificate authentication. If the server requires client authentication, the Inside ACOS device checks whether the server name or web category matches the configuration condition to bypass this traffic.
4. If a match is found, the Inside ACOS device stops SSLi processing and switches from HTTPS processing to basic TCP proxy processing.
5. A TCP connection to the server is established where client and server can directly negotiate the SSL session bypassing the ACOS SSLi.

Figure 36 : Client Authentication Traffic Network Example



CLI Configuration

The forward-proxy-bypass client-auth CLI command configures the SNI attributes and/or class-lists that determine whether or not a client is enabled for client-authentication bypass. These attributes and class-lists are bound to SSL client template which itself is bound to the inside ACOS device. The forward-proxy-bypass client-auth CLI command options follow:

```
slb template client-ssl Client-SSL
forward-proxy-bypass client-auth case-insensitive
forward-proxy-bypass client-auth class-list testclass
forward-proxy-bypass client-auth contains jsmith
forward-proxy-bypass client-auth ends-with abc
forward-proxy-bypass client-auth equals test.hello.com
forward-proxy-bypass client-auth starts-with efg
```

For more details on the forward-proxy-bypass command see the subcommand table under the `slb template-client-ssl` command.

GUI Configuration

1. Navigate to **Security > SSLi > Templates** and edit your client ssl template (such as `SSLi_vip_001_client_ssl`).
2. In the Update Client SSL Template window, click the **Policy** tab.
3. For Bypass Client Auth, click **Add**.
4. Expand the Condition section and select an option from the drop-down list:
 - **SNI Contains**
 - **SNI Ends with**
 - **SNI Starts with**
 - **SNI Equals**
5. For Value, enter the matching value of the client to bypass authentication.
6. You can add multiple match rules. Click **Add** as needed.
7. Click **Update**.

Consolidated Configuration

ACOS_decrypt

The following sample configuration shows how to configure the inside ACOS device for client authentication bypass:

```
ACOS-inside# show running-config
access-list 101 permit ip 10.10.1.0 0.0.0.255 any
!
class-list Client_Auth_Bypass ac
  starts-with a10a10
  equals ssl-i
  contains hello.com
!
interface ethernet 1
  enable
!
interface ethernet 2
  enable
!
vlan 10
  untagged ethernet 1
  router-interface ve 10
!
vlan 20
  untagged ethernet 2
  router-interface ve 20
!
interface ve 10
  ip address 10.10.1.10 255.255.255.0
  ip allow-promiscuous-vip
!
interface ve 20
  ip address 10.10.2.10 255.255.255.0
!
slb server FW1_SSLei 10.10.2.20
  health-check-disable
  port 0 tcp
  health-check-disable
```

```
port 0 udp
    health-check-disable
port 8080 tcp
    health-check-disable
!
!
slb service-group Outbound_TCP tcp
    member FW1_SSLi 0
!
slb service-group Outbound_UDP udp
    member FW1_SSLi 0
!
slb service-group Outbound_SSLi tcp
    member FW1_SSLi 8080
!
slb template client-ssl Client-SSL
    forward-proxy-ca-certificate Cert123.pem key key123
    forward-proxy-enable
    forward-proxy-bypass client-auth contains abcd
    forward-proxy-bypass client-auth class-list Client_Auth_Bypass
!
slb virtual-server Inside_SSLi_VIP 0.0.0.0 acl 101
    port 443 https
        no-dest-nat port-translation
        service-group Outbound_SSLi
        template client-ssl Client-SSL
    port 0 tcp
        no-dest-nat
        service-group Outbound_TCP
    port 0 udp
        no-dest-nat
        service-group Outbound_UDP
    port 0 others
        no-dest-nat
        service-group Outbound_UDP
!
end
```

Outside ACOS device

The following CLI output shows how to configure the outside ACOS device:

```
ACOS-outside# show running-config
access-list 101 permit ip any any vlan 20
!
interface ethernet 1
    enable
!
interface ethernet 2
    enable
!
vlan 40
    untagged ethernet 1
    router-interface ve 40
!
vlan 20
    tagged ethernet 2
    router-interface ve 20
!
interface ve 40
    ip address 10.10.4.20 255.255.255.0
!
interface ve 20
    ip address 10.10.2.20 255.255.255.0
    ip allow-promiscuous-vip
!
slb server Gateway 10.10.4.1
    health-check-disable
    port 0 tcp
        health-check-disable
    port 0 udp
        health-check-disable
    port 443 tcp
        health-check-disable
!
!
slb service-group Outbound_TCP tcp
    member Gateway 0
!
```

```
slb service-group Outbound_UDP udp
  member Gateway 0
!
slb service-group Outbound_SSL tcp
  member Gateway 443
!
slb template server-ssl Server-SSL
  forward-proxy-enable
!
slb template virtual-port ignore-msl
  ignore-tcp-msl
!
slb virtual-server Outside_nonSSLi_VIP 0.0.0.0 acl 101
  port 8080 http
    service-group Outbound_SSL
    template server-ssl Server-SSL
    no-dest-nat port-translation
    use-rcv-hop-for-resp
  port 0 tcp
    service-group Outbound_TCP
    no-dest-nat
    use-rcv-hop-for-resp
    template virtual-port ignore-msl
  port 0 udp
    service-group Outbound_UDP
    no-dest-nat
    use-rcv-hop-for-resp
  port 0 others
    service-group Outbound_UDP
    no-dest-nat
    use-rcv-hop-for-resp
!
end
```

Troubleshooting

SSLi might fail for one of the following reasons:

- If the configuration of client authentication is present on the client SSL template on the server side but missing on the client side, the ACOS device will not be able retrieve the server certificate during the SSL handshake.
- SSLi could fail in any other generic case such as abrupt connection closure by server FIN due to malformed packet, and so on.

When SSLi fails, a log is generated that includes the following information:

- SNI
- Server IP Address

When the connection is successful, no logs are generated.

NOTE: The log messages are only seen by the inside ACOS device.

Log Example

When "SSLVerifyClient require" and "SSLVerifyDepth 10" is set up on APACHE ssl.conf, on the server, there is a failure when retrieving the certificate because no client side authentication has been configured.

As a result, the following log is generated:

```
ACOS# show log
Log Buffer: 30000
Nov 30 2014 09:03:19 Info      [SYSTEM]:SSL intercept failed, server amogh-
server (ip 20.20.101.50)
ACOS#
```

IP Based Bypass

While the AC class list with the matching rules can be used for SSLi bypass and intercept decision-making, you can also use a specific list of client IP and/or server IP (IPv4 and IPv6) class lists for SSLi bypass or intercept decision-making.

This section describes how to configure the IP based class lists under the client-SSL template to do SSLi bypass or intercept decision-making.

The following topics are covered:

Configure Client IP Class List	394
Configure Server IP Class List	396

Configure Client IP Class List

This section describes how to configure the client IP class list-based SSLi bypass and intercept method.

You can configure:

- Client IPs class list of both IPv4 and IPv6 types.
- Up to eight class-list of each IPv4 and IPv6 type can be configured under one client-SSL template.

CLI Configuration

- To configure the bypass rule if the client IP matches class-list, use the following commands:

1. Create a IPv4 and/or IPv6 class-list and specify the IPv4 and/or IPv6 host or subnet.

```
ACOS(config)#class-list C4 ipv4
ACOS(config-class list)#3.3.3.0/24
ACOS(config-class-list)#class-list C6 ipv6
ACOS(config-class list)#2::2/96
ACOS(config)#exit
```

2. Bind the class-list under client-ssl template.

```
ACOS(config)#slb template client-ssl test1
ACOS(config-client ssl)#forward-proxy-bypass client-ip-list ipv4 C4
ACOS(config-client ssl)#forward-proxy-bypass client-ip-list ipv6 C6
```

- To configure the bypass rule for multiple client IP class-list, use the following commands:

1. Create multiple IPv4 and/or IPv6 class-list and specify the IPv4 and/or IPv6 host or subnet.

```
ACOS(config)#class-list C1 ipv4
```

```
ACOS(config-class list)#1.1.1.0/24
ACOS(config-class list)#class-list C2 ipv4
ACOS(config-class list)#1.1.1.1/24
ACOS(config-class list)#class-list C3 ipv4
ACOS(config-class list)#1.1.1.2/24
```

2. Bind the multiple class-list under client-ssl template.

```
ACOS(config)#slb template client-ssl SSL_Client
ACOS(config-client ssl)#forward-proxy-bypass client-ip-list ipv4 C1
ACOS(config-client ssl)#forward-proxy-bypass client-ip-list ipv4 C2
ACOS(config-client ssl)#forward-proxy-bypass client-ip-list ipv4 C3
```

- To configure the intercept rule if the client IP matches class-list, use the following commands:

1. Create a IPv4 and/or IPv6 class-list and specify the IPv4 and/or IPv6 host or subnet.

```
ACOS(config)#class-list CI1 ipv4
ACOS(config-class list)#1.0.3.4/24
ACOS(config-class list)#class-list CI2 ipv6
ACOS(config-class list)#3::3/26
```

2. Bind the new class-list under client-ssl template.

```
ACOS(config)#slb template client-ssl SSL_Client
ACOS(config-client ssl)#forward-proxy-bypass exception-client-ip-
list ipv4 CI1
ACOS(config-client ssl)#forward-proxy-bypass exception-client-ip-
list ipv6 CI2
```

GUI Configuration

1. To configure a client IP class-list, navigate to **Shared Objects > Shared Object Class List**.
2. To bind the new class-list to client SSL template, navigate to **ADC > Templates > SSL > Client SSL > Forward-Proxy-Bypass** and select the following options:
 - Client IPv4 Class List
 - Client IPv6 Class List

- Exception Client IPv4 Class List
- Exception Client IPv6 Class List

Configure Server IP Class List

This section describes how to configure the server IP class-list based SSLi bypass and intercept method.

You can configure:

- Server IPs class list of both IPv4 and IPv6 types.
- Up to eight class-list of each IPv4 and IPv6 type can be configured under one client-SSL template.

CLI Configuration

- To configure the bypass rule if the server IP matches class-list, use the following commands:

1. Create a IPv4 and/or IPv6 class-list and specify the IPv4 and/or IPv6 host or subnet.

```
ACOS(config)#class-list S4 ipv4  
ACOS(config-class list)#3.3.3.0/24  
ACOS(config-class list)#class-list S6 ipv6  
ACOS(config-class list)#2::2/96  
ACOS(config)#exit
```

2. Bind the class-list under client-ssl template.

```
ACOS(config)#slb template client-ssl test1  
ACOS(config-client ssl)#forward-proxy-bypass server-ip-list ipv4 S4  
ACOS(config-client ssl)#forward-proxy-bypass server-ip-list ipv6 S6
```

- To configure the bypass rule for multiple server IP class-list, use the following commands:

1. Create the multiple IPv4 and/or IPv6 class-list and specify the IPv4 and/or IPv6 host or subnet.

```
ACOS(config)#class-list S1 ipv4
```

```
ACOS(config-class list)#1.1.1.0/24
ACOS(config-class list)#class-list S2 ipv4
ACOS(config-class list)#1.1.1.1/24
ACOS(config-class list)#class-list S3 ipv4
ACOS(config-class list)#1.1.1.2/24
```

2. Bind the multiple class-list under client-ssl template.

```
ACOS(config)#slb template client-ssl SSL_Client
ACOS(config-client ssl)#forward-proxy-bypass server-ip-list ipv4 S1
ACOS(config-client ssl)#forward-proxy-bypass server-ip-list ipv4 S2
ACOS(config-client ssl)#forward-proxy-bypass server-ip-list ipv4 S3
```

- To configure the intercept rule if the server IP matches class-list, use the following commands:

1. Create a IPv4 and/or IPv6 class-list and specify the IPv4 and/or IPv6 host or subnet.

```
ACOS(config)#class-list SI4 ipv4
ACOS(config-class list)#2.3.3.0/24
ACOS(config)#class-list SI6 ipv4
ACOS(config-class list)#2.3.3.0/24
```

2. Bind the new class-list under client-ssl template.

```
ACOS(config)#slb template client-ssl SSL_Client
ACOS(config-client ssl)#forward-proxy-bypass exception-server-ip-
list ipv4 SI4
ACOS(config-client ssl)#forward-proxy-bypass exception-server-ip-
list ipv6 SI6
```

GUI Configuration

1. To configure a server IP class-list, navigate to **Shared Objects > Shared Object Class List**.
2. To bind the new class-list to client SSL template, navigate to **ADC > Templates > SSL > Client SSL > Forward-Proxy-Bypass** and select the following options:
 - Server IPv4 Class List
 - Server IPv6 Class List

- Exception Server IPv4 Class List
- Exception Server IPv6 Class List

SSLi Bypass for "no shared cipher" Error

If the "no shared cipher" error occurs during the client-side handshake, SSLi terminates the connection. Additionally, the `forward-proxy-failsafe` option is ineffective in this case as the cipher check occurs early in the SSL handshake. Hence, ACOS provides a `forward-proxy-no-shared-cipher-action` option to either bypass or drop the connection. This option can be enabled using both the CLI and the GUI.

Known Limitation

The `forward-proxy-no-shared-cipher-action bypass` option is not supported in the SSLi and proxy-chain configurations.

CLI Configuration

Perform the following steps to create a client-SSLi template that bypasses SSLi connections where there is a `no-shared-cipher` error during the SSLi handshake.

1. Create a client SSL template called `SSLInsight_DecryptSide` by running the following command:

```
ACOS(config)# slb template client-ssl SSLInsight_DecryptSide
```

2. Configure bypass for the `forward-proxy-no-shared-cipher-action` option

```
ACOS(config-client ssl)#forward-proxy-no-shared-cipher-?
forward-proxy-no-shared-cipher-action Action taken if handshake fails
due to no shared cipher, close the connection by default
ACOS(config-client ssl)# forward-proxy-no-shared-cipher-action ?
bypass  bypass SSLi processing
drop    close the connection
ACOS(config-client ssl)# forward-proxy-no-shared-cipher-action bypass
```

GUI Configuration for “no shared cipher” Error

Perform the following steps to create a client-SSL template that bypasses SSLi connections where there is a no-shared-cipher error during the SSLi handshake.

1. Navigate to **Security >> SSLi >> Templates >> Create >> Client SSL**.
2. Alternatively, navigate to **ADC >> Templates >> SSL >> Create >> Client SSL**.
The **Create Client SSL Template** page is displayed.
3. Fill in the required fields.
4. Under the **forward-proxy-no-shared-cipher-action** option, select either **Drop** or **Bypass**.
By default, the value is **Drop**.
5. Click **OK**.

Consolidated Configuration

You can configure a number of client-ssl templates for SSLi bypass using a combination of the commands available under **forward-proxy-bypass**.

The following topics are covered:

“no-shared-cipher” Error	399
AAM, User Name, AD Group, Explicit Proxy, and SSLi	400
AAM, User Name, AD Group Name, Transparent Proxy, and SSLi	401

“no-shared-cipher” Error

The following client-SSL template bypasses the SSLi connection under three conditions:

- For financial services
- For financial services

For “no-shared-cipher” error

!

```
slb template client-ssl SSLInsight_DecryptSide
    forward-proxy-ca-certificate Cert123.pem key key123
    forward-proxy-ocsp-disable
    forward-proxy-crl-disable
    forward-proxy-cert-expiry hours 168
    forward-proxy-enable
    forward-proxy-failsafe-disable
    forward-proxy-bypass web-category financial-services
    forward-proxy-bypass web-category health-and-medicine
    forward-proxy-no-shared-cipher-action bypass
```

AAM, User Name, AD Group, Explicit Proxy, and SSLi

In this example, the SSLi solution uses a combination of AAM and user name group and AD name group to create SSLi bypass decisions.

Three class lists are configured as AC lists. These are **UNAME**, **GROUP**, and **BYPASS_EXCEPTION**.

In this template, **BASIC** is the profile for HTTP-based logon and it is associated with the AAM authentication template of **ssli_BYPASS**. The SSLi client template is configured as **USER_BYPASS** and it includes an exception list of **BYPASS_EXCEPTION**, a user name list of **UNAME** and a AD group list of **GROUP** for bypass. There is also an explicit policy template of **EP_SSLi** for forward proxy. Finally, the virtual server is associated with the explicit proxy template, the SSLi client template, and the AAM authentication template.

```
class-list UNAME ac
    equals asmith
    equals jdoe
!
class-list GROUP ac
    equals Employee
!
class-list BYPASS_EXCEPTION ac
    equals TEST
!
aam authentication logon http-authenticate BASIC
    auth-method basic enable
!
```



```

aam authentication template SSLI_BYPASS
  logon BASIC
  server AD_LDAP
!
aam aaa-policy SSLI_BYPASS
  aaa-rule 1
    authentication-template SSLI_BYPASS
!
slb template client-ssl USER_BYPASS
  forward-proxy-ca-certificate ...
  forward-proxy-enable
  forward-proxy-bypass exception-ad-group-list BYPASS_EXCEPTION
  forward-proxy-bypass user-name-list UNAME
  forward-proxy-bypass ad-group-list GROUP
!
slb template policy EP_SSLI
  forward-policy
  ...
!
slb virtual-server EP 10.0.0.1
  port 3128 http
  template policy EP_SSLI
  template client-ssl USER_BYPASS
  aaa-policy SSLI_BYPASS
!

```

AAM, User Name, AD Group Name, Transparent Proxy, and SSLI

In this example, the SSLI solution uses a combination of AAM and user name group and AD name group to create SSLI bypass decisions.

A class list of **UNAME** is configured as an AC list. In this example, **BASIC** is the profile for HTTP-based logon and it is associated with the AAM authentication template of **ssli_BYPASS**. The SSLI client template is configured as **USER_BYPASS** and it includes the user name list of **UNAME** for bypass. There is also transparent proxy template of **TP_SSLI** configured with **forward-policy**.

Finally, the virtual server at port **443 HTTPS** is associated with the transparent proxy template, the SSLi client template, and the AAM authentication template. **no-destination-nat** and **port translation** are enabled.

```
access-list 10 permit 172.16.1.0 0.0.0.255
!
class-list UNAME ac
  equals asmith
  equals jdoe
!
aam authentication logon http-authenticate BASIC
  auth-method basic enable
!
aam authentication template SSLI_BYPASS
  auth-sess-mode ip-based
  logon BASIC
  server AD_LDAP
!
aam aaa-policy SSLI_BYPASS
  aaa-rule 1
    authentication-template SSLI_BYPASS
!
slb template client-ssl USER_BYPASS
  forward-proxy-ca-certificate ...
  forward-proxy-enable
  forward-proxy-bypass user-name-list UNAME
!
slb template policy TP_SSLI
  forward-policy
  ...
!
slb virtual-server TP 0.0.0.0 acl 10
  port 443 https
  service-group DUMMY
  template policy TP_SSLI
  template client-ssl USER_BYPASS
  no-dest-nat port-translation
  aaa-policy SSLI_BYPASS
!
```

Web Category and Web Reputation Bypass

The following topics are covered:

[Web Category Bypass](#) 404

[Web Reputation Bypass](#) 417

Web Category Bypass

Web Category refers to a set of features that includes URL Classification and Asynchronous Lookup. Classifying URLs provides this information that is used to filter unwanted content, adding additional layer of security. The information can also determine which URLs should bypass SSLi decryption to comply with privacy laws.

Installing Web Category License

Web Category features are accessed through a Web Category license and an active URL Classification Database.

ACOS connects with third-party servers (specifically, Webroot's BrightCloud servers), to obtain this information for enhanced protection. To access these servers, a URL Classification license is required. Two Webroot license types are available:

- Local – covers top 20 million URLs
- Cloud-based (plus local) – access to Webroot URL classification database (27 billion URLs)

The following topics are covered:

Step 1: Installing the Web Category License	404
Step 2: Verifying the Web Category License Installation	406
Step 3: Activating the Web Category License	406
Step 4: Verifying the Web Category Library	406
Step 5: Checking Web Category License Status and Expiration	407

Step 1: Installing the Web Category License

The license import method works for both the local and cloud-based (plus local) licenses. The following steps install a Web Category License:

1. Configure your ACOS device with a valid ip route and domain name server (DNS).

The following is an example. Use the `show run ip` command to verify the configuration.

```
ACOS(config)# ip route 0.0.0.0 /0 192.168.200.1
ACOS(config)# ip dns primary 192.168.1.100
ACOS(config)# show run ip
!Section configuration: 69 bytes
!
ip route 0.0.0.0 /0 192.168.200.1
!
ip dns primary 192.168.1.100
```

2. Ensure the ACOS device does not block access to the following URLs:
 - <https://glm.a10networks.com/>
 - <https://database.brightcloud.com>
 - <http://service.brightcloud.com>
3. Save your URL Classification license file on an accessible server.
4. Enter the web-category sub-command mode by entering **web-category**, and configure the use of the management port for communication with the BrightCloud servers using the **use-mgmt-port** CLI command. The **exit** command returns to global configuration mode.

```
ACOS(config)# web-category
ACOS(config-web-category)# use-mgmt-port
ACOS(config-web-category)# exit
```

5. Import your Web Category license file using the CLI command at the global configuration mode level. The *file-name* is the name of the Web Category license file.

```
import web-category-licensefile-name
```

The following example shows the output when the URL Classification license file has been imported.

```
ACOS(config)# import web-category-license test.json use-mgmt-port
scp://example@192.168.1.200/home/example/lic_test/test_URL_C.json
Password []?
Done.
```

Step 2: Verifying the Web Category License Installation

Verify the URL Classification License on an ACOS device, by using the **show log** CLI command. The output verifies the URL Classification license is imported onto the ACOS device.

```
ACOS(config)# show log | grep WEB-CATEGORY.
```

This output example displays the relevant portion (highlighted in blue) of a successful Web Category license installation.

```
ACOS(config)# show log
Log Buffer: 30000
Oct 30 2015 16:23:39 Info      [SYSTEM]:Imported file test.json from
example:192.168.1.200/home/example/lic_test/test_URL_C.json using scp
Oct 30 2015 16:23:39 Info      [WEB-CATEGORY]:BrightCloud license activated
successfully
Oct 30 2015 16:23:38 Info      [WEB-CATEGORY]:license key used for activation:
{"id":"581b839aba28b1d39a55a39dae909b9e7383b564b7b1f7eaa215f851d460f73e","signature":
"61f7b36da2e88cfa2fb3943434563cdafe58e221b83ca44d3b8e73d40183f795","current_
time":1446244661.6663604,"payload":"eyJ0b2t1biI6InZUaGNmOTQ2Y2IxZSJSJ9\n","account_
id":497,"uuid":"AX25061111340044"}
...
```

Step 3: Activating the Web Category License

The Web Category license must be enabled before utilizing the database. Use the **enable** CLI command from the web-category configuration mode to enable web-category functionality.

```
ACOS(config)# web-category
ACOS(config-web-category)# enable
```

Step 4: Verifying the Web Category Library

The Web Category database installation is verified with the **show web-category database** command. The following displays an example command output:

```
ACOS> show web-category database
Database Name           : full_bcdb_rep_7.431.bin
Database Status         : Active
Database Size           : 351 MB
Database Version        : 827
Last Update Time       : Wed Jul  6 19:39:59 2016
```

```
Next Update Time      : Fri Jul  8 00:00:22 2016
Connection Status     : GOOD
Last Successful Connection : Thu Jul  7 00:39:22 2016
```

From the GUI, navigate to **Security >> Web Categories** and click on **License** to view the database information.

Step 5: Checking Web Category License Status and Expiration

After installing a Web Category License, check the expiration date and status by entering `show web-category license`. The following example displays typical command output.

```
ACOS> show web-category license
Module Status      : Enabled
License Status     : License is valid
License Type       : Term License
License Expiry     : 2016-11-30 00:00:00 GMT
Remaining Period   : 145 d 17 hrs 26 min 3 sec
Grace Period Status : License has not expired
Grace Period       : Grace period not in effect
UUID/SN           : EX0000000000000000
```

From the GUI, navigate to **Security >> Web Categories** and click on **License** to view license status and expiration date information.

Using a Proxy Server for BrightCloud Servers

BrightCloud servers are hosted in a location where the IPs are subject to change. This can be a issue to administrators with an upstream firewall in their networks when they need to manage a list of allowed IPs to allow communication between ACOS and the BrightCloud servers. One solution is to have all BrightCloud communication go through a proxy server, so IP management is no longer necessary.

From the web-category sub-configuration, enter `proxy-server` to go to web-category-proxy-server sub-configuration. Here, the following minimum requirements are needed for configuration.

- Authentication protocol - BASIC authentication is supported.
- Server information -
 - IP address or hostname of proxy server
 - port for HTTPS or HTTP communication with proxy server. If only one port type is configured, both HTTP and HTTPS communication go through the configured port type.

The proxy-server sub-configuration has commands to configure the username and password for authentication. Refer to “Web Category” in *Command Line Interface Reference for ADC*.

An example of a configuration to a proxy server is provided. This example configures port 3128 for HTTP communication and port 8080 for HTTPS communication, with the username *exampleadmin* and password *0e1x2a3m4p5l6e7* to sign in to a proxy server at 192.0.2.0.

```
ACOS(config)# web-category
ACOS(config-web-category)# proxy-server
ACOS(config-web-category-proxy-server)# proxy-host 192.0.2.0
ACOS(config-web-category-proxy-server)# http-port 3128
ACOS(config-web-category-proxy-server)# https-port 8080
ACOS(config-web-category-proxy-server)# username exampleadmin
ACOS(config-web-category-proxy-server)# password 0e1x2a3m4p5l6e7
ACOS(config-web-category-proxy-server)# exit
```

A number of options to configure how and when ACOS interacts with the BrightCloud Servers, for example, configuring when an update should occur, is available from the Command Line Interface Reference for ADC in “Web Category”. These options are available through the GUI by navigating to **Security>>Web Categories >> Configure**.

Filtering Web Category for SSLi Bypass

An ACOS device can utilize web category features in `forward-policy source` rules. This links destination and matching rules for an `s1b template policy` through a category-list. For specifying the web categories to SSLi bypass, use the `forward-proxy-bypass` command in an `s1b template client-ssl`.

Configuring SSLi Bypass Filtering

This section describes how to configure ACOS device to bypass SSLi decryption of traffic based on traffic category. Dynamic Web Category classification is provided using the BrightCloud Webroot Web Security Service.

BrightCloud classifies the traffic into one or more web categories. Encrypted traffic from the client is not intercepted if the web category of the traffic is configured to be bypassed (example: Healthcare due to HIPPA regulation). If a specific web category is not bypassed, traffic of that category is decrypted for interception.

When a user's client browser sends a request to a URL, ACOS checks the category of the URL.

- If the category of the URL is allowed by the configuration, the ACOS_decrypt leaves the data encrypted and sends it to ACOS_encrypt, which sends the encrypted data to the server.
- If the category of the URL is not allowed by the configuration, the ACOS_decrypt decrypts the traffic and sends it to the traffic inspection device.

Similarly, reply traffic from the server is decrypted by the ACOS_encrypt for interception, if the web category is not bypassed. ACOS_decrypt then sends the encrypted data to the client.

To configure ACOS to use BrightCloud to classify URLs for SSLi bypass:

- Configure ACOS_encrypt. (The configuration steps for this feature are described in the *Application and Server Load Balancing Guide*.)
- Configure BrightCloud Web Category classification services on the ACOS_decrypt. (This may include installing the BrightCloud license, if not already installed.)
- Configure forward-proxy-bypass web-category rules on ACOS_decrypt.

For simplicity, a simple topology using a single ACOS_decrypt and a single ACOS_encrypt is used.

ACOS_decrypt Configuration

Here is the configuration of the ACOS device on the inside network, connected to clients. Encrypted client traffic to the following categories of URL is bypassed (forwarded without being decrypted):

- financial-services
- educational-institutions
- health-and-medicine

SSLi decrypts traffic to URLs that are not labeled as belonging to any of these bypassed categories.

ACOS_encrypt Configuration

No Web Category classification commands are required on this device. All of the Web Category classification configuration takes place on the ACOS_decrypt.

Viewing the Statistics

As per your requirement, the following show commands are available to view various Web Category statistics.

- The following command shows the current Web Category URL statistics under Client-SSL template.

This lists each bypassed web category, along with the number of times it has been bypassed. Intercepted web categories are counted under `Other Categories`. If the BrightCloud database cannot classify traffic into a Web category, then it is listed under `uncategorized`:

```
ACOS# show slb template client-ssl url-stats
slb template client-ssl ssl_int
Category hits:
uncategorized          0
financial-services     42
nudity-artistic        17
illegal-pornography    17
travel                 3
training-and-tools     0
web-based-email        5
Other Categories       83
Reputation hits:
Trustworthy(81-100)    0
Low-risk(61-80)       0
Moderate-risk(41-60)  0
```

Suspicious (21-40)	0
Malicious (1-20)	0

- The following command shows the Web Category information such as the bypassed-urls, intercepted-urls, and the BrightCloud database:

```
ACOS# show web-category ?
  bypassed-urls      Show list of URL's bypassed
  database           Show information about currently loaded BrightCloud
database
  intercepted-urls    Show list of URL's intercepted
  url-category       Show categories returned by BrightCloud library for
a URL
  version            Show BrightCloud library version
```

- The following command shows the current version of the Web Category engine:

```
ACOS# show web-category version
version: 4.0
```

- The following command shows information about the currently loaded BrightCloud database:

```
ACOS# show web-category database
Database name          : full_bcd_b_4.457.bin
Database size          : 352 MB
Database version       : 457
Last Update Time       : Fri Jan 23 00:00:40 2015
Next Update Time       : Sat Jan 24 00:00:43 2015
Connection Status      : GOOD
Last Successful Connection : Fri Jan 23 15:54:43 2015
```

- The following command shows the 20 most recently bypassed URLs:

```
ACOS# show web-category bypassed-urls 20
paper.example.com
paper.example.com
paper.example.com
paper.example.com
step.example.com
metrics1.example.com
step.example.com
paper.example.com
```

```
online.example.com  
...
```

- The following command shows the 20 most recently intercepted URLs:

```
ACOS# show web-category intercepted-urls 20  
fhr.data.example.com  
fhr.data.example.com  
fhr.data.example.com  
aus3.example.org  
blocklist.addons.example.org  
aus4.example.org  
versioncheck-bg.addons.example.org  
versioncheck-bg.addons.example.org  
services.addons.example.org  
aus3.example.org  
fhr.data.example.com  
...
```

- The following commands show the web categories to which some individual URLs belong.

In this example, the categories for the URLs in the ACOS's local database match the most recent categorization from the BrightCloud server.

```
ACOS# show web-category url-category www.google.com  
Search Engines  
ACOS# show web-category url-category www.google.com local-db-only  
Search Engines  
ACOS# show web-category url-category http://www.youtube.com  
Streaming Media  
ACOS# show web-category url-category www.youtube.com local-db-only  
Streaming Media
```

Deleting or Reimporting the Database

Disabling the Web Category classification feature does not delete the database. Likewise, re-enabling the feature does not cause the database to be downloaded again.

- To delete the database:

```
ACOS(config)# web-category
ACOS(config-web-category)# noenable
ACOS(config-web-category)# exit
ACOS(config)# delete web-category database
```

- To re-import the database, first disable the feature and delete the database that is on the ACOS device (as shown above), then re-enable the Web Category classification feature:

```
ACOS(config)# web-category
ACOS(config-web-category)# use-mgmt-port
ACOS(config-web-category)# enable
```

NOTE: Simply disabling and re-enabling the feature does not delete and reload the database. In this case, the same database is used.

Troubleshooting

The following troubleshooting commands are used for Webroot on the ACOS_decrypt:

- `debug web-category`
- `debug monitor`

Error during database download of Webroot

If you see the following error messages during enable under web-category configuration:

```
[WEB-CATEGORY] downloading full_bcdb_4.445.bin
[WEB-CATEGORY] BcDownloadDb: failed to InitializeSsl context
[WEB-CATEGORY] nDownloadAndApplyDatabaseUpdates( ) 0 - call to
BcDownloadDatabaseUpdates( ) failed.
```

A required certificate file may be missing. Contact A10 Networks.

Verify the ACOS_decrypt Has Downloaded Certificates from the HTTPS Server

```
show slb ssl-forward-proxy-cert SSLi_vip-1 443 all
```

Verify Traffic Flow

On the ACOS_encrypt: `show slb virtual-server`

- Bypassed SSL traffic packet and connection counters will go up under port 0.
- Intercepted SSL traffic and HTTP protocol packet and connection counters will go up under port 8080.

On the ACOS_decrypt: `show slb virtual-server`

- SSL traffic packet and connection counters will go up under port 443.
- HTTP protocol packet and connection counters will go up under port 0.

Logging

ACOS supports remote logging for the Web Category classification feature. The provided information includes the URL accessed by the client, to which category the URL belongs to and action taken by ACOS: intercept or bypass. Logs are provided in Common Event Format (CEF). Remote logging for the feature is disabled by default.

NOTE: To use remote logging, you also must configure a remote syslog server on ACOS using the `logging host host-ipaddr` command. The current release does not support use of the management interface for remote logging for Web Category classification.

CEF format comprises of a syslog prefix, header and an extension. A typical ACOS message in CEF contains the following fields:

```
Timestamp host CEF:Version|Device-Vendor|Device-Product|Device-Version|  
Signature-ID|Name|Severity|[Extensions]
```

Log messages for Web Category classification have the following fields:

- Syslog prefix: the starting of the message with timestamp on syslog server and hostname of ACOS device.
- CEF header: All fields in the header are mandatory.
 - Version: Identifies the version of CEF format. ACOS uses version 0.
 - Device Vendor, Device Product and Device Version: Used to uniquely identify the device.
 - Signature ID and Name: Unique identifier for an event and “name” is a string

giving a description of this event. For this feature, there are two event types: SSLi connection intercepted and SSL connection bypassed:

- SSLi100 -> SSLi request intercepted
- SSLi101 -> SSLi request bypassed
- **Severity:** Integer that reflects importance of the event with range 1-10. 10 indicates most important event. In this example, the value is 5 for both events.
 - **Extensions:** a collection of key-value pairs to provide more information about the event. A predefined set of keys are provided by CEF format. The following keys are used in case of Signature ID 1 (URL lookup).
 - **Request:** URL accessed by the client.
 - **Act** stands for deviceAction: Action taken by device. Values are going to be intercepted or bypassed.
 - **Msg:** An additional message about the log. In our case it will be category is xxx, where xxx is the category into which URL is categorized by the BrightCloud server.
 - **Src** stands for sourceAddress: Source IP address if the address is an IPv4 address.
 - **Dst** stands for destinationAddress: Destination IP address if the address is an IPv4 address.
 - **C6a2** stands for deviceCustomIPv6Address2: This is a custom field used to show the source network address in case of an IPV6 address.
 - **C6a2label** stands for deviceCustomIPv6Address2Label: Explains what the field c6a2 is for. In this case, it will be Source IPv6 address.
 - **C6a3** stands for deviceCustomIPv6Address3: This is a custom field used to show the destination network address in case of an IPV6 address.
 - **C6a3label** stands for deviceCustomIPv6Address3Label: Explains what the field c6a3 is for. In this case, it will be DestinationIPv6 address.
 - **Spt** stands for sourcePort: Source port number on the client.
 - **Dpt** stands for destinationPort: Destination port number client is trying to access.

Implementing Lookup Enforcement

Web-category lookups in the data plane are performed by querying the local database for a URL and returning the URL category when the database contains the URL. When the database does not contain the URL, it returns the value "uncategorized" and the resolution of the unknown URL is delayed and performed in the background. Therefore, lookup result of the first request of an unknown URL is always "uncategorized". In SSLi / Forward-proxy deployments, the lapse of proper URL categorization can result in intercepting requests that should be bypassed or allowing requests that should be dropped.

Web category lookup enforcement resolves the category of unknown (first request) URLs by pausing the data plane connection. When the result is known and the URL is categorized, the connection is resumed.

To enable web category lookup enforcement through the ACOS CLI, enter `require-web-category` under the following templates as applicable:

- policy template for URL filtering
- client-ssl template for web-reputation-based SSLi bypass

Similarly, Web category lookup enforcement can be utilized for Web Reputation. When the web-reputation is set, the reputations score will be checked. If the score is 0, means that the reputation score can not be resolved from server or local library. When it fails and if the option `require-web-category` is set, then ACOS will send the query to the BrightCloud to get the reputation score. And the connection will be in pending status at this time until it receives the response, and then continue to check if the rules are matched or not. Otherwise, if the reputation score is not retrieved from the local cache and the option `require-web-reputation` is not set, the by default it continues to intercept process directly.

Implementing URL Filtering

SSLi when deployed as an Explicit Proxy or a Transparent Proxy, utilize a policy template that binds a web-category list to destination rules. Web Category Lookup Enforcement is enabled for these features by adding the `require-web-category` option to the policy template.

The following example enables Web Category Lookup Enforcement for all actions defined under the RED policy template.


```
ACOS(config)# slb template policy RED
ACOS(config-policy)# forward-policy
ACOS(config-policy-forward-policy)# require-web-category
ACOS(config-policy-forward-policy)#
```

Implementing SSLi Bypass

The client-ssl template includes **forward-proxy-bypass require-web-category** - Enables web-category based SSLi bypass policies under that template.

The following example enables Web Category Lookup Enforcement for web-category based SSLi bypass policies under the BLUE client-ssl template.

```
ACOS(config)# slb template client-ssl BLUE
ACOS(config-client ssl)# forward-proxy-bypass web-category financial-
services
ACOS(config-client ssl)# forward-proxy-bypass web-category health-and-
medicine
ACOS(config-client ssl)# forward-proxy-bypass exception-web-category games
ACOS(config-client ssl)# forward-proxy-bypass require-web-category
ACOS(config-client ssl)#
```

Web Reputation Bypass

Web Reputation refers to the URL reputation scores, which helps the SSLi to decide whether to intercept or bypass the URL. This improves the security and limits the inappropriate and malicious URLs to pass through your network. As per the standards, the Web Reputation score range from (1-100) with tiers split into the following levels and one customized option:

- Trustworthy (81-100)
- Low Risk (61-80)
- Moderate Risk (41-60)
- Suspicious (21-40)
- Malicious (1-20)
- <1-100>

Web Reputation feature can be accessed through the existing Web Category database and cloud lookup. For more information on:

- Web category installation and configuration, refer [Installing Web Category License](#).
- Web reputation CLI commands, refer *Command Line Interface Reference*.

The following topics are covered:

Configuring the Forward Policy	418
Configuring the SSLi Bypass Filtering	419
Viewing the Statistics	420

Configuring the Forward Policy

Before you begin to use the web reputation feature, you must perform the following steps:

1. Configure the `reputation-scope` under the `web-category`.
2. Bind the above setting to the forward policy template destination.
3. Configure the web reputation feature under `forward-policy` `source` rules.

This setting links the destination and matching rules for an `slb template policy` through a `web-reputation-scope`.

NOTE: The `reputation-scope` can be greater-than or less-than the score level or customized score. Additionally, only one entry can be set in one scope for each greater-than or less-than setting.

The following example enables web-reputation based forward policy template:

```
ACOS(config)# slb template policy BLUE
ACOS(config-policy)# forward-policy
ACOS(config-policy-forward-policy)# require-web-category
ACOS(config-policy-forward-policy)# action A1
ACOS(config-policy-forward-policy-action)# source S1
ACOS(config-policy-forward-policy-source)# destination web-reputation-
scope scope_1 action A1 url priority 120
```

The following example defines reputation-scope list in the web-category configuration node and this can be used in forward policy template:

```
ACOS(config)# web-category
ACOS(config-web-category)# reputation-scope trustworthy
ACOS(config-web-category-reputation-scope)# greater-than trustworthy
```

Configuring the SSLi Bypass Filtering

After configuring the forward-policy, you can enable the Web Reputation for SSLi bypass decision-making by using the client-SSL template. You can also view the web-reputation statistics hits for the client-SSL template:

- **forward-proxy-bypass web-reputation** - When the web-reputation score is greater than or equal to the setting level or customized score, the request would be bypassed.
- **forward-proxy-bypass exception-web-reputation** - When the web-reputation score is less than or equal to the setting level or customized score, the request would be intercepted.

The following example enables exception-web-reputation based SSLi bypass policies under the client-SSL template for the financial-services category. If the reputation score is less than 40, it will be intercepted or else bypassed.

```
ACOS(config)# slb template client-ssl BLUE
ACOS(config-client ssl)# forward-proxy-bypass web-category financial-
services
ACOS(config-client ssl)# forward-proxy-bypass exception-web-reputation
suspicious
```

The following example enables both the web-reputation and exception-web-reputation based SSLi bypass policies under the client-SSL template.

```
ACOS(config)# slb template client-ssl BLUE
ACOS(config-client ssl)# forward-proxy-bypass web-reputation low-risk
ACOS(config-client ssl)# forward-proxy-bypass exception-web-reputation
moderate-risk
```

Viewing the Statistics

- The following example shows the web-reputation statistics for the client-ssl template:

```
ACOS(config)# show slb template client-ssl BLUE
Category hits:
  Other Categories                                0
Reputation hits:
  Trustworthy(81-100)                            2
  Low-risk(61-80)                               1
  Moderate-risk(41-60)                          1
  Suspicious(21-40)                             1
  Malicious(1-20)                               1
```

- The following command shows the URLs bypassed by the web reputation feature:

```
ACOS(config)# show web-reputation bypassed-urls
Score      URL
79         www.77file.com
81         www.testing.com
81         a10networks.com
...
```

- The following command shows the URLs intercepted by the web reputation feature:

```
ACOS(config)# show web-reputation intercepted-urls
Score      URL
10         17ebook.com
40         gerry90160.a10-tplab.com
54         earn4files.com
```

- The following commands show the web reputation scores of the URLs:

```
ACOS(config)# show web-reputation url-reputation www.youtube.com
trustworthy(81)
ACOS(config)# show web-reputation url-reputation www.google.com
trustworthy(81)
ACOS(config)# show web-reputation url-reputation www.abc.com
trustworthy(96)
```

```
ACOS(config)# show web-reputation url-reputation www.17ebook.com  
malicious(10)
```

URL Filtering

This section provides guidelines for the implementation of URL Filtering configurations. URL Filtering can be implemented either by web category or SNI matching.

The following topics are covered:

Overview	423
CLI Configuration	424
GUI Configuration	425
Consolidated Configuration	426

Overview

Forward policy actions follow after the decision has been made in the by the Client-SSL template whether to bypass or intercept. In other words, after ACOS processes the incoming traffic as provisioned the Client-SSL template, then it processes the incoming traffic as provisioned by the forward policy.

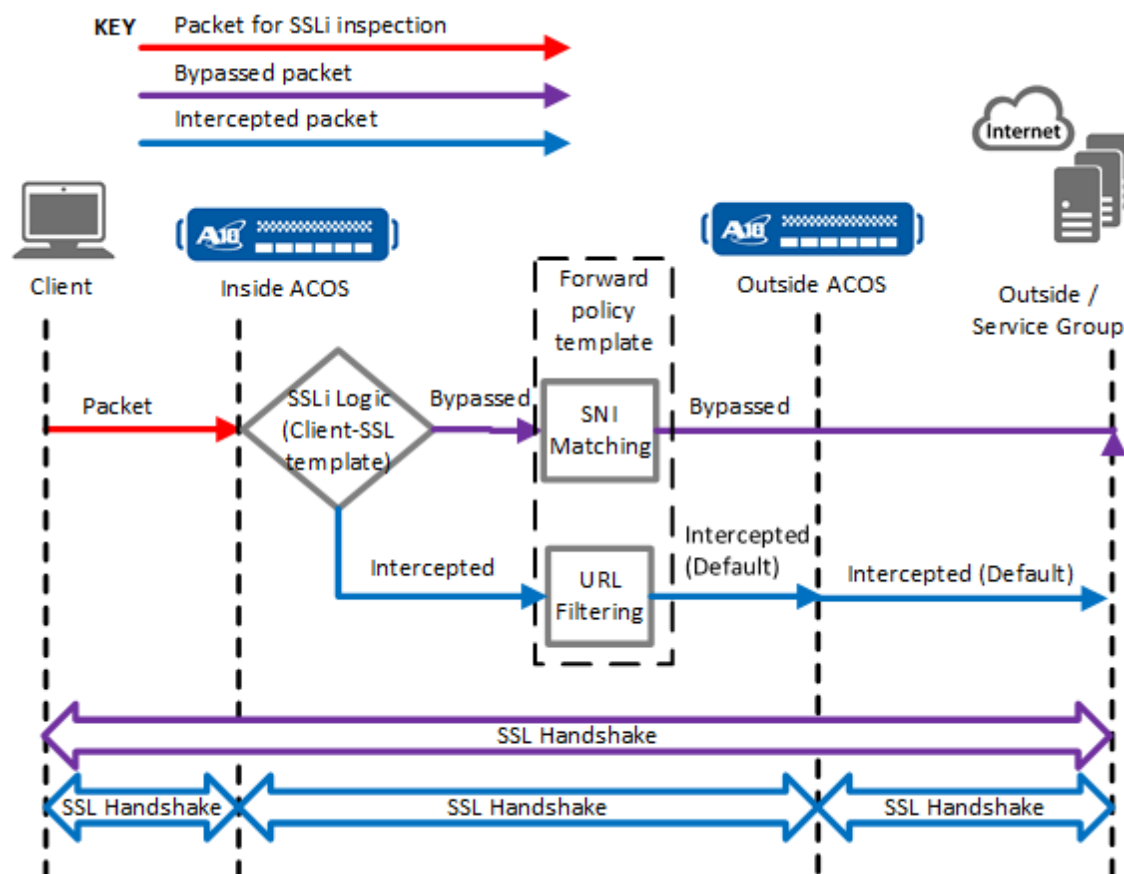
The SSLi forward policy handles the traffic of bypassed (non-decrypted) sessions differently than the traffic of intercepted (decrypted) sessions. This difference is illustrated in [Figure 37](#).

In a bypassed connection, by default ACOS examines the server name identification (SNI) field to determine a course of action for the traffic of that connection.

In a intersected connection, by default ACOS looks at the client's request HTTP header to determine a course of action.

While these actions work by default for an SSLi configuration, options are available to provide different ways of handling bypassed and intercepted SSLi packets by using the `ssli-url-filtering` CLI command from the forward-policy configuration mode in an SLB template policy that is applied to a SLB client-SSL template. The specific options for `ssli-url-filtering` are available under the `forward-policy` command in the *Command Line Interface Reference*.

Figure 37 : Transparent Proxy with SSLi SNI Matching and URL Filtering Default Packet Flow Sequence



CLI Configuration

This section describes how to add transparent HTTP proxy services to the SSLi.

In this example, we create a server load balancing template policy `ExamplePolicy`, followed by the `forward-policy` sub-command and configure `ssli-url-filtering` to allow transparent SSLi proxy traffic not containing SNI extension information to be forwarded, rather than being dropped (default action).

```
ACOS(config)# slb template policy ExamplePolicy
ACOS(config-policy)# forward-policy
ACOS(config-policy-forward-policy)# ssli-url-filtering no-sni-allow
```


Other actions that are configurable include disabling SNI inspection on bypassed traffic, enabling SNI matching for intercepted transparent proxy SSLi traffic and disabling HTTP header inspection for intercepted transparent proxy SSLi traffic (see `ssli-url-filtering` in the Command Line Interface Reference Guide).

Known Limitations:

- From the forward-policy configuration, `no-client-conn-reuse` is not supported in a server load balancing template policy consisting of a HTTPS virtual port and a wildcard VIP. The commands are permitted, but it will be ignored for this specific case.
- From the forward-policy configuration, `drop-message` and `drop-redirect-url` are not supported in the case where the ACOS device acts as a transparent proxy with a SSLi connection due to the fact that the drop commands are http level messages, but with SNI matching, the device is inspecting at the SSL handshake level.

GUI Configuration

This section describes the steps to configure SSL Insight URL filtering options using the GUI.

1. Navigate to **Security >> Forward Proxy**.
2. Click on the **Templates** tab.
3. Click **+ Create** and click on **Policy**.
4. In the **Add Policy Template** page, enter a policy name in the **Name** field.

NOTE:	It does not matter if the Action Policies tab or Source Policies tab has been selected.

5. In SSLi URL Filtering, click on the check box for the SSLi URL Filtering options you wish to be active.
 - Bypassed SNI Disable
 - Intercepted SNI Enable

- Intercepted HTTP Disable
- NO SNI Allow

6. Click **Add Template**.

Consolidated Configuration

The following example deployment illustrates configurations for SSLi bypass in the Client-SSL template and URL filtering and SNI matching in the forwarding policy.

In this example, a web-category category-list drops requests from clients trying to connect to sites classified as various types of security risks. The failsafe-disable option is disabled so that when an SSL handshake transaction fails, the traffic inspection is not bypassed. Because of privacy rules, this configuration does not decrypt and inspect the financial transactions and medical and health categories.

For further information on configuration of the forward-policy, see the [“Explicit and Transparent Proxy”](#) section.

```
Current active partition: ssli_in
ACOS[ssli_in]#show run
!Current configuration: 1546 bytes
!Configuration last updated at 21:21:06 PST Fri Mar 10 2017
!Configuration last saved at 12:57:23 PST Thu Mar 9 2017
!
active-partition ssli_in
!
!
access-list 190 remark ssli_in
!
access-list 190 permit ip any any vlan 850
!
access-list 191 remark block_quic
!
access-list 191 deny udp any any eq 80
!
access-list 191 deny udp any any eq 443
!
access-list 191 permit ip any any
!
```

```
class-list Block_domains ac
  contains sslitest
!
web-category
  category-list Url_filter_cat
    malware-sites
    phishing-and-other-fraud
    proxy-avoid-and-anonymizers
    spyware-and-adware
    bot-nets
    confirmed-spam-sources
    spam-urls
    unconfirmed-spam-sources
!
slb template cipher cl_cipher_template
  SSL3_RSA_DES_192_CBC3_SHA
  TLS1_RSA_AES_128_SHA
  TLS1_RSA_AES_256_SHA
  TLS1_ECDHE_RSA_AES_128_SHA
  TLS1_ECDHE_RSA_AES_256_SHA
  TLS1_ECDHE_RSA_AES_128_SHA256
!
slb server fw1 30.91.11.104

  port 0 tcp
    health-check-disable
    _0_tcp_port
  port 0 udp
    health-check-disable
    _0_udp_port
  port 80 tcp
    health-check-disable
    _80_tcp_port
  port 8080 tcp
    health-check-disable
    user-tag Security,ssli_signaling
!
slb service-group SG_SSLi_HTTP tcp
```

```
member fw1 80
!
slb service-group SG_SSLi_TCP tcp

member fw1 0
!
slb service-group SG_SSLi_UDP udp

member fw1 0
!
slb service-group SG_SSLi_Xlated tcp

member fw1 8080
!
slb template client-ssl cl_ssl
template cipher cl_cipher_template
forward-proxy-ca-certificate Cert123.pem key key123
forward-proxy-ocsp-disable
forward-proxy-crl-disable
forward-proxy-cert-expiry hours 168
forward-proxy-enable
forward-proxy-failsafe-disable
forward-proxy-bypass web-category financial-services
forward-proxy-bypass web-category health-and-medicine
!
slb template http insertHeaders
non-http-bypass service-group SG_SSLi_Xlated
!
slb template policy Url_filter_pl

forward-policy
action Drop
drop
log
action Permit
forward-to-internet SG_SSLi_Xlated
action permi
source Any
match-any
destination class-list Block_domains action Drop url priority 20
```

```
        destination web-category-list Url_filter_cat action Drop url
priority 10
        destination any action Permit
!
slb virtual-server SSLi_in_ingress 0.0.0.0 acl 190
    port 0 tcp
        service-group SG_SSLi_TCP
        no-dest-nat
    port 0 udp
        service-group SG_SSLi_UDP
        no-dest-nat
    port 0 others
        service-group SG_SSLi_UDP
        no-dest-nat
    port 80 http
        service-group SG_SSLi_Xlated
        template policy Url_filter_pl
        no-dest-nat port-translation
    port 443 https
        service-group SG_SSLi_Xlated
        template policy Url_filter_pl
        template http insertHeaders
        template client-ssl cl_ssl
        no-dest-nat port-translation
!
end
!Current config commit point for partition 1 is 0 & config mode is
classical-mode
ACOS[ssli_in]#
```

TLS Fingerprinting

TLS fingerprinting is a technology used to detect and mitigate TLS handshake attacks. This technique associates an application and/or TLS library with parameters derived from a TLS Client Hello using a curated fingerprint database.

TLS negotiations are usually transmitted in clear text. Following the three-way handshake, the client will send a Client Hello message. Client Hello messages are generated by many programs using various packages and programmatic approaches. As a result, this packet will always be the same from one application to the next. The fingerprint itself is based on the unique way a client and server establish a secure connection.

ACOS supports TLS fingerprinting with [JA3](#) and [JA4+](#) methods that can be implemented to:

- Insert the JA3 or JA4+ fingerprint as an HTTP header and forward it to the backend HTTP server.
- Reject SSL/TLS connections by class-list containing JA3 or JA4+ fingerprints of the known malicious clients.
- Reject SSL/TLS connections if the JA3 or JA4+ fingerprints amount of the client address exceeds a certain value.

ACOS creates and stores JA3 or JA4+ fingerprint records in the partition biased table, which is shared by all the client-SSL templates in the same L3V partition.

Limitation

ACOS does not synchronize JA3 or JA4+ records for high availability.

The following topics are covered:

JA3 Fingerprinting Overview	431
JA4+ Fingerprinting Overview	432
CLI Configuration	433
GUI Configuration	435
Show Commands	436

JA3 Fingerprinting Overview

The JA3 method utilizes the following fields from the Client Hello message and hashes the result using MD5 to produce a 32-character fingerprint.

- SSL/TLS Version
- Ciphers Accepted
- Extensions List
- Elliptic Curves
- Elliptic Curve Formats

JA3 method follows the below workflow or process:

1. First it gathers the decimal values of the bytes for the above fields.
2. Then it concatenates those values together in order, using a “,” to delimit each field and a “-” to delimit each value in each field.

Example: 769,47-53-5-10-49161-49162-49171-49172-50-56-19-4,0-10-11,23-24-25,0

If there are no SSL Extensions in the Client Hello, the fields are left empty.

Example: 769,4-5-10-9-100-98-3-6-19-18-99,,,

3. Lastly, it hashes the strings to MD5 to produce an easily consumable and shareable 32 character fingerprint.

Example:

769,47-53-5-10-49161-49162-49171-49172-50-56-19-4,0-10-11,23-24-25,0

→ ada70206e40642a3e4461f35503241d5

769,4-5-10-9-100-98-3-6-19-18-99,,,

→ de350869b8c85de67a350c8d186f11e6

The JA3 fingerprint will look identical to any other application that establishes a TLS connection. The fingerprint itself is based on the unique way a client and server establish a secure connection.

JA4+ Fingerprinting Overview

JA4+ is a successor to the JA3 TLS fingerprinting method, which uses both human and machine-readable fingerprinting techniques.

JA4+ examines the TLS Client Hello packet and builds a fingerprint for the client based on various attributes within the packet, as shown in [Figure 38](#) and described in the [Table 16](#).

The JA4 fingerprinting looks like: `t13i3111h2_e8f1e7e78f70_b26ce05bbdd6`.

Figure 38 : JA4+ TLS Client Fingerprint Attributes

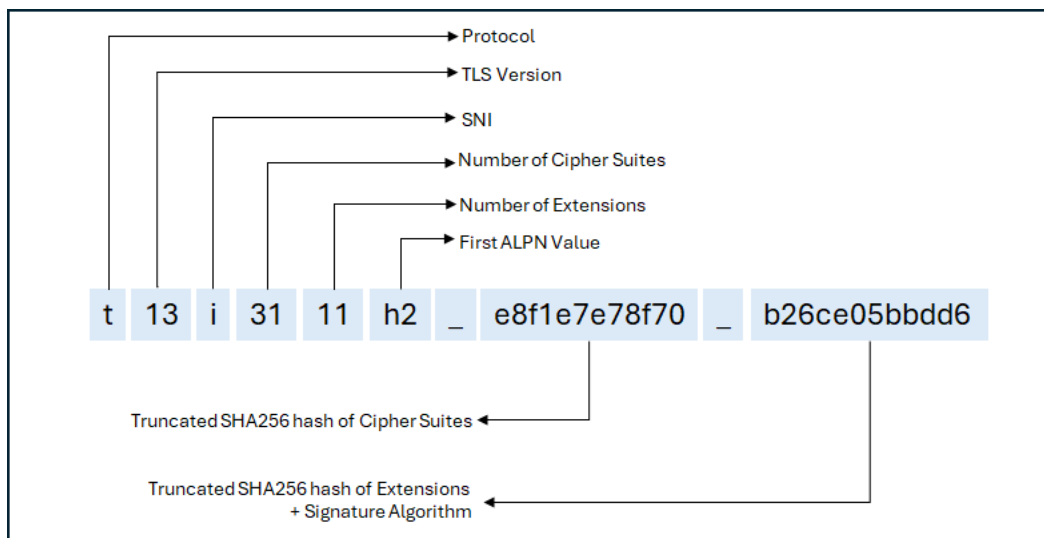


Table 16 : JA4+ Attributes Explanation

Attribute	Description
Protocol	Indicates whether the hello packet is for QUIC, DTLS, or normal TLS/TCP protocol. For example, - QUIC="q" - DTLS="d" - Normal TLS="t"
TLS Version	Indicates the TLS version. For example, 1.2 or 1.3.
SNI	Indicates whether the SNI extension exists or not. For example, SNI="d" means the destination of the connection is a domain.

Table 16 : JA4+ Attributes Explanation

Attribute	Description
	no SNI="i" means the destination is an IP address.
Number of Ciphers	Indicates two-character count of ciphers. For example, if there are 8 ciphers the value will be 08. If the count is more than 99, they don't count.
Number of Extensions	Indicates two-character count of extensions. Same as ciphers.
ALPN Extension Value	Indicates the first and last characters of the first Application-Layer Protocol Negotiation (ALPN) extension value. For example, if it is the http/1.1 protocol, the "h1" character is displayed.
Cipher hash	Indicates the 12-character truncated SHA256 hash of the list of cipher sorted in hex order, first 12 characters.
Extension hash	Indicates the 12-character truncated SHA256 hash of the list of extensions sorted in hex order and followed by the list of signatures.

CLI Configuration

This topic describes how to enable the JA4 fingerprinting features or options for TLS client connections under client-SSL template using CLI.

Similarly, you can configure the JA3 fingerprinting features by replacing `ja4` with `ja3`.

Enable JA4+ Feature

To enable the JA4+ feature, use the following command:

```
ACOS(config)#slb template client-ssl SSLI
ACOS(config-client ssl)#ja4-enable
ACOS(config-client ssl)#exit
```

Insert JA4+ as a HTTP Header

This feature computes the JA4 hash and inserts that hash into a custom HTTP

header and forwards it to the backend HTTP server.

NOTE: Since the server is HTTP, the traffic must be HTTPS, and SSLi or SSL offload must be configured, otherwise the header cannot be inserted.

To insert the JA4 hash into the request as a HTTP header, use the following command:

```
ACOS(config)#slb template client-ssl SSLI
ACOS(config-client ssl)#ja4-enable
ACOS(config-client ssl)#ja4-insert-http-header <string, length> {X-JA4}
ACOS(config-client ssl)#exit
```

Reject TLS Connections by Class-List

This feature checks the class-list containing the JA4 fingerprints of the known malicious clients. If JA4 of the current connection matches any member in the class-list, SSL rejects this connection.

To reject TLS connections by class-list, use the following commands:

- Configure a class-list (type string-case-insensitive) in the global configuration mode.

```
ACOS(config)#class-list reject_class string-case-insensitive
ACOS(config-class list)#str F436B9416F37D134CADD04886327D3E8
```

- Bind the defined class-list in the client-SSL template.

```
ACOS(config)#slb template client-ssl SSLI
ACOS(config-client ssl)#ja4-enable
ACOS(config-client ssl)#ja4-reject-class-list reject_class
ACOS(config-client ssl)#exit
```

Reject TLS Connections by Amount

This feature configures the maximum JA4 amount of client address allowed in the client-SSL template. You can also set the TTL value to retain each JA4 record. If the JA4 amount of client address exceeds this number, SSL rejects this connection.

To configure JA4+ fingerprinting amount of each client address, use the following commands:

```
ACOS(config)#slb template client-ssl SSLI
```

```
ACOS(config-client ssl)#ja4-enable
ACOS(config-client ssl)#ja4-reject-max-number-per-host 2 ttl 30
ACOS(config-client ssl)#exit
```

GUI Configuration

This topic describes how to enable the JA3 fingerprinting feature for TLS client connections under client-SSL template using GUI.

NOTE: The JA4+ fingerprinting feature is not supported on ACOS GUI.

Enable JA3 Feature

To enable JA3 feature, navigate to:

- **ADC > Templates > SSL > Client SSL > Enable JA3** option.
- **Security > SSLi > Templates > Client SSL > Enable JA3** option.

Insert JA4+ as a HTTP Header

To configure this feature, navigate to:

- **ADC > Templates > SSL > Client SSL > JA3 Insert HTTP Header** option.
- **Security > SSLi > Templates > Client SSL > JA3 Insert HTTP Header** option.

Reject TLS Connections by Class-List

To configure this feature, navigate to:

- **ADC > Templates > SSL > Client SSL > JA3 Reject Class List** option.
- **Security > SSLi > Templates > Client SSL > JA3 Reject Class List** option.

Reject TLS Connections by Amount

To configure this feature, navigate to:

- **ADC > Templates > SSL > Client SSL > JA3 Reject Maximum Number Per Host and JA3 TTL** options.
- **Security > SSLi > Templates > Client SSL > JA3 Reject Maximum Number Per Host and JA3 TTL** options

View JA3 amount

To view the current number of JA3 amount, navigate to:

- **ADC > Statistics > L7 > SSL > SSL JA3**
- **Security > SSLi > Reports > SSL JA3**

Show Commands

This section describes the show commands for the JA3 and JA4+ fingerprinting.

- To view the current amount of the JA4+ fingerprints of each address, use the following command:

```
ACOS(config)#show slb ssl-ja4
Host          Numbers of JA4
-----
10.251.3.2    1
```

- To view the current amount of the JA3 fingerprints of each address, use the following command:

```
ACOS(config)#show slb ssl-ja3
Host          Numbers of JA3
-----
10.251.3.1    1
```

For the multi-PU platforms, the statistics display the aggregated counters; PU1 + PU2.

ICAP Services

This section provides information on configuring Internet Content Adaptation Protocol (ICAP) in a static-port SSLi deployment.

The following topics are covered:

Overview	438
Topology Example	439
CLI Configuration	441
GUI Configuration	443
Show Commands	445
ICAP Configuration Options	445

Overview

ICAP provides security services to HTTP and HTTPS sessions. On traffic from the client to the web server, ICAP typically serves to provide data loss prevention (DLP). Whereas, on traffic from the Web server to the client, ICAP typically provides anti-virus (AV) services.

ICAP services are frequently deployed in conjunction with forward proxy, such as SSLi to intercept and inspect traffic as the man-in-the-middle.

Limitation

- The SSLi virtual port feature described does not support ICAP.
- ICAP with proxy chaining is not supported on the same ACOS device.

ICAP Template Options

The following REQMOD template options can be configured:

- `allowed-http-methods` - List of allowed HTTP methods

Note: This option is not supported when a template is specified in the `forward-http-connect-to-icap reqmod-icap` command.

- `fail-close` - Mark the virtual port down when the template service group is down
- `include-protocol-in-uri` - Include the protocol and port in the HTTP URI sent to the ICAP server
- `min-payload-size` - Set the minimum payload size sent to the ICAP server

Note: This option is not supported when a template is specified in the `forward-http-connect-to-icap reqmod-icap` command.

- `preview` - The number of bytes that ACOS forwards to the ICAP server at the beginning of a transaction.

Note: This option is not supported when a template is specified in the `forward-http-connect-to-icap reqmod-icap` command.

- `service-group` - The names of the ICAP service groups
- `service-url` - The URLs of the ICAP servers
- `template` - ACOS logging, server-ssl, and tcp-proxy templates applied to this ICAP transactions

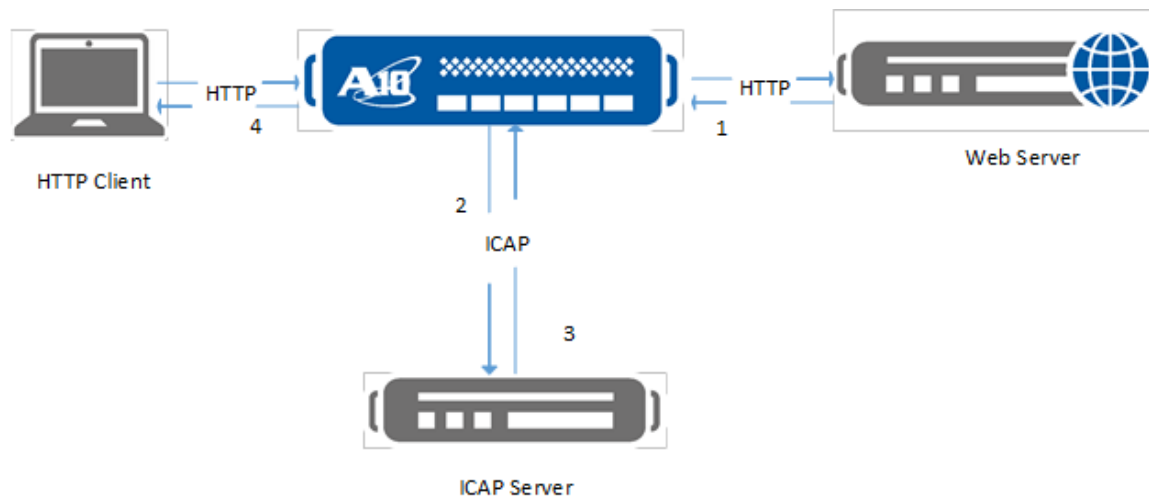
The following RESPMOD template options can be configured:

- `fail-close` - Mark the virtual port down when the template service group is down
- `include-protocol-in-uri` - Include the protocol and port in the HTTP URI sent to the ICAP server
- `min-payload-size` - Set the minimum payload size sent to the ICAP server
- `service-group` - The names of the ICAP service groups
- `service-url` - The URLs of the ICAP servers
- `template` - ACOS logging, server-ssl, and tcp-proxy templates applied to this ICAP transactions

Topology Example

[Figure 39](#) below shows a sample ICAP topology. The numbers in the diagram show the messaging steps described in the following section.

Figure 39 : ICAP REQMOD Message Exchange



When the ACOS device is configured as an ICAP client with Request Modification Process (REQMOD) capability and is also configured as a forward proxy for an HTTP client, the ICAP message exchange process follows these steps:

1. The web client sends an HTTP GET request to the Web server.
2. The ACOS device intercepts the request, processes the HTTP header, and forwards it to the ICAP server in an ICAP REQMOD message to the ICAP server.
3. The ICAP server sends a REQMOD response to the ACOS device.
4. The ICAP REQMOD response and the actions taken by the ACOS device can be one or more of the following:

- ICAP REQMOD response has Status Code 200 and contains an HTTP request.

The ACOS device sends the HTTP request contained in the ICAP response to the web server (instead of the original intercepted HTTP request).

- ICAP REQMOD response has Status Code 204.

The ACOS device sends the original intercepted HTTP request to the web server.

- ICAP REQMOD response has Status Code 100.

The ACOS device the ACOS device needs to send more data to the ICAP server.

- ICAP REQMOD response has Status Code 200 contains an HTTP response.

The ACOS device does not send an HTTP request to the web server. Instead, it sends this HTTP response back to client.

- ICAP REQMOD response has any other Status Code.

The ACOS device treats the ICAP response as if it were Status Code 204.

CLI Configuration

This section refers to the inside and outside ACOS devices in the SSLi configuration. Equivalent configurations can be provisioned on a single ACOS device split into an inside and outside partitions.

The inside partition performs decryption and is often called the decryption partition, while the outside partition performs re-encryption and is often called the re-encryption partition.

NOTE: Although this example shows ICAP configured on the inside ACOS device virtual port 443, it can alternatively be configured on the outside ACOS device on virtual port 8080 that receives decrypted traffic.

Inside_SSLi

1. First, configure the IP address of the ICAP server and create an ICAP service group to provide a path to the ICAP server. This example assumes that the ICAP server is listening over port 1344.

```
ACOS-Inside(config)# slb server ICAP_server_1 10.1.260.11
ACOS-Inside(config-real server)# port 1344 tcp
ACOS-Inside(config)# slb service-group SG_ICAP tcp
ACOS-Inside(config-slb svc group)# member ICAP_server_1 1344
```

2. Create the ICAP REQMOD template. Include the ICAP service group and the URL of the ICAP REQMOD service:

The `template reqmod-icap` command provisions the ICAP server for ICAP REQMOD messaging, and the `template respmod-icap` command provisions the ICAP server for ICAP RESPMOD messaging.

```
ACOS-Inside(config)# slb template reqmod-icap REQMOD_abcd
ACOS-Inside(config-reqmod-icap)# service-group SG_ICAP
ACOS-Inside(config-reqmod-icap)# service-url
icap://dlpserver:1344/reqmod
```

3. Optionally, the REQMOD connection can be secured by enabling SSL with an SSL-server template, such as is shown in the following commands:

```
ACOS-Inside(config)# slb template reqmod-icapREQMOD_abcd
ACOS-Inside(config-reqmod-icap)# service-group SG_ICAP
ACOS-Inside(config-reqmod-icap)# service-url
icap://dlpserver:11344/reqmod
ACOS-Inside(config-reqmod-icap)# template server-ssl ssl
```

4. Create the ICAP RESPMOD template. Include the ICAP service group and the URL of the ICAP RESPMOD service:

```
ACOS-Inside(config)# slb template respmod-icap RESPMOD_abcd
ACOS-Inside(config-reqmod-icap)# service-group SG_ICAP
ACOS-Inside(config-reqmod-icap)# service-url
icap://dlpserver:1344/respmod
```

5. Optionally, the RESPMOD connection can be secured by enabling SSL with an SSL-server template, such as is shown in the following commands:

```
ACOS-Inside(config)# slb template reqmod-icap RESPMOD_abcd
ACOS-Inside(config-reqmod-icap)# service-group SG_ICAP
ACOS-Inside(config-reqmod-icap)# service-url
icap://dlpserver:11344/respmod
ACOS-Inside(config-reqmod-icap)# template server-ssl ssl
```

6. Bind the ICAP templates to the HTTPS virtual port of the wildcard VIP configured in the [Two-Device Static-HTTPS-Port SSLi Configuration](#).

```
ACOS-Inside(config)# slb virtual-server Decrypt_VIP 0.0.0.0 acl 100
ACOS-Inside(config-slb vserver)# port 443 https
ACOS-Inside(config-slb vserver-vport)# template reqmod-icap REQMOD_abcd
ACOS-Inside(config-slb vserver-vport)# template respmod-icap RESPMOD_
abcd
```

```
ACOS-Inside(config-slbf vserver-vport)# service-group FW1_Inspect_SG
ACOS-Inside(config-slbf vserver-vport)# no-dest-nat port-translation
```

NOTE: The order of packet processing for HTTP Layer 7 virtual ports is described in the “Usage Guidelines” section of the `port` command (virtual server configuration mode/level) in the *Command Line Interface Reference Guide*.

7. When you bind an ICAP template to the HTTP or HTTPS port of a virtual server, you are configuring the ACOS device to operate as an ICAP client. This enables the ACOS device to forward decrypted intercepted traffic to the ICAP servers specified in the template.

Outside_SSLi

The following example shows ICAP configured on the outside ACOS device.

The ICAP templates are bound to virtual port 8080 because that is the port that receives decrypted SSL traffic.

```
ACOS-Outside(config)# slbf virtual-server Outside_VIP 0.0.0.0 acl 101
ACOS-Outside(config-slbf vserver)# port 8080 http
ACOS-Outside(config-slbf vserver-vport)# no-dest-nat port-translation
ACOS-Outside(config-slbf vserver-vport)# service-group DG_SSL_SG
ACOS-Outside(config-slbf vserver-vport)# template server-ssl SSLInsight_
ServerSide
ACOS-Outside(config-slbf vserver-vport)# template reqmod-icap REQMOD_abcd
ACOS-Outside(config-slbf vserver-vport)# template respmod-icap RESPMOD_abcd
ACOS-Outside(config-slbf vserver-vport)# exit
```

GUI Configuration

Configure the RESPMOD and REQMOD templates.

1. Navigate to **ADC >> Templates >> L7 Protocols**
2. To begin the creation of the RESPMOD template, click the **+ Create** button and select **RESPMOD**.

3. When the **Create RESPMOD Template** pop-up window appears, the only required field is the **Name** of the template. In this example we configure the following fields:
 - a. The previously configured service group, SG_ICAP provides a path over which ACOS can connect to the RESPMOD and REQMOD servers. Select **SG_ICAP** for the **Service Group** field.
 - b. The **URL** of the RESPMOD server is entered as `service-url
icap://dlpserver:1344/respmod.`
4. Click the **Create** button to complete the creation of the RESPMOD template.
5. To begin the creation of the REQMOD template, click the **+ Create** button and select **REQMOD**.
6. When the **Create REQMOD Template** pop-up window appears, the only required field is the **Name** of the template. In this example we configure the following fields:
 - a. The previously configured service group, SG_ICAP provides a path over which ACOS can connect to the RESPMOD and REQMOD servers. Select **SG_ICAP** for the **Service Group** field.
 - b. The **URL** of the REQMOD server is entered as `service-url
icap://dlpserver:1344/reqmod.`
7. Click the **Create** button to complete the creation of the REQMOD template.

For a static-port SSLi configuration in which there is an inside virtual server and an outside virtual server in separate partitions or configured on separate ACOS devices, the following steps bind the RESPMOD and REQMOD templates to the inside VIP to enable ICAP RESPMOD and REQMOD services.

Bind the RESPMOD and REQMOD templates to the inside SSLi VIP.

1. Navigate to **Security >> SSLi >> Services**.
2. Assuming SSLi is already configured, click the **Edit** button of the inside VIP.
3. When the **Update SSLi Service** pop-up window appears, click the **Edit** button of the **https 443** virtual port.
4. When the **Update SSLi Service Port** pop-up window appears, click **More Options...**

5. Notice that the client-ssl template that you previously configured on the inside SSLi virtual server appears.
6. In the **Templates** field, select **reqmod-icap** from the drop-down list and then click the **+Add** button.
7. A new row should appear for the **reqmod-icap** template above the client-ssl row. For the **Name** of the reqmod-icap template, select **REQMOD_abcd** which was created above. Click **Apply** to bind the template to the port.
8. To bind the RESPMOD_abcd template to the port, select **respmod-icap**, and click **+Add**.
9. Select **RESPMOD_abcd** (also created above) and click **Apply** to bind the template to the port.

Show Commands

Use the following show commands to view the ICAP traffic or statistics:

- The `show slb icap` command displays statistics that includes both blocked and not blocked traffic.
- The `show slb icap-http` command displays the statistics specific to ICAP blocked traffic. When traffic is blocked by the ICAP server, it sends the HTTP response to ACOS.

ICAP Configuration Options

The following topics are covered:

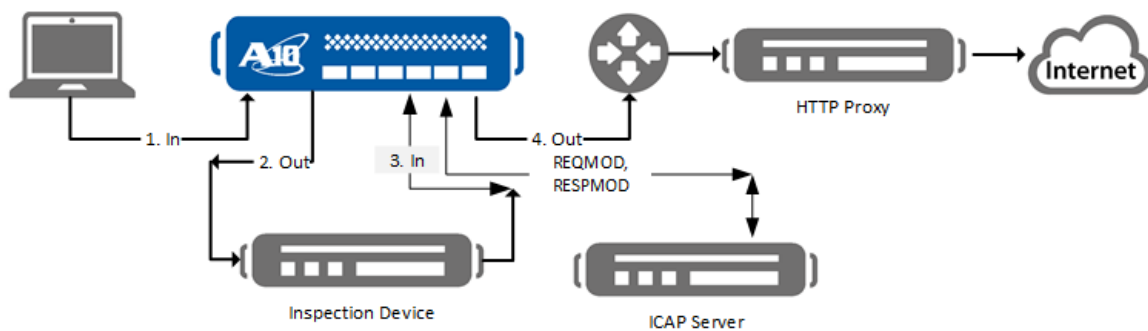
Including Protocol and Port in HTTP URI	446
Pre-Filtering ICAP Traffic	448
Configuring ACOS Logging	450

Including Protocol and Port in HTTP URI

When a connection request is forwarded through HTTPS transparent proxy (such as ACOS SSLi), ICAP forwards the entire URL (including URL scheme and FQDN) of the site requested.

In the scenario where there is a web proxy with authentication, you can configure the web proxy to relay the user information, and would configure ICAP on the outside ACOS device. See [Figure 40](#)

Figure 40 : ICAP Services in a Proxy Chain Topology



For configuration information, see [CLI Configuration](#) and traffic workFlow:

1. After HTTP header processing is done, ACOS checks the allowed methods and the minimum payload size (if a payload exists). If both checks are passed, ACOS proceeds to the next step.
 - a. The allowed HTTP methods are specified by the `allowed-http-methods` command under `template reqmod-icap`.
 - b. The minimum payload length is specified by the `min-payload-size` command under `template reqmod-icap`.
2. When copying the request, if the `include-protocol-in-uri` command is configured, the server URL is converted to an absolute URI with the protocol, host and port number in the URI. The user-defined X- headers described in “ICAP Extensions, draft-stecher-icap-subid-00.txt” are used for this purpose.
3. If secure ICAP is configured by the `template server-ssl` command, the TCP SSL callback routines are used. But, if the `template server-ssl` command is not enabled, the regular ICAP handshake proceeds.

4. The ICAP packet is built and sent to the ICAP server.
5. When the ICAP server responds, if the handshake is SSL, ACOS decrypts and calls the ICAP processing code.
6. ACOS logs the ICAP transaction information.

CLI Configuration

1. To provision the outside VIP to relay the original port and protocol that was changed during decryption functions, the ICAP templates are configured with the **include-protocol-in-uri** command.

```
ACOS(config)# slb template reqmod-icap REQMOD_abcd
ACOS(config-reqmod-icap)# include-protocol-in-uri
ACOS(config)# slb template respmod-icap RESPMOD_abcd
ACOS(config-reqmod-icap)# include-protocol-in-uri
ACOS-Outside(config)# slb virtual-server Outside_VIP 0.0.0.0 acl 101
ACOS-Outside(config-slb vsrver)# port 8080 http
ACOS-Outside(config-slb vsrver-vport)# no-dest-nat port-translation
ACOS-Outside(config-slb vsrver-vport)# service-group DG_SSL_SG
ACOS-Outside(config-slb vsrver-vport)# template server-ssl SSLInsight_
ServerSide
ACOS-Outside(config-slb vsrver-vport)# template reqmod-icap REQMOD_
abcd
ACOS-Outside(config-slb vsrver-vport)# template respmod-icap RESPMOD_
abcd
ACOS-Outside(config-slb vsrver-vport)# exit
```

2. To use the **include-protocol-in-uri** for ICAP on the outside ACOS device (or re-encrypt partition), you also need to have the X-Protocol-Port header injected on the inside ACOS device (or decrypt partition) via HTTP template.

```
ACOS-Inside(config)# slb template http insert_port
ACOS-Inside(config-http)# request-header-insert "X-Protocol-Port: https
443"
```

3. Apply the HTTP template under the virtual port 443 https of the inside ACOS device.

```
ACOS-Inside(config)# slb virtual-server Decrypt_VIP 0.0.0.0 acl 100
ACOS-Inside(config-slb vsrver)# port 443 https
ACOS-Inside(config-slb vsrver-vport)# service-group FW1_Inspect_SG
```

```
ACOS-Inside(config-slb vserver-vport)# no-dest-nat port-translation
ACOS-Inside(config-slb vserver-vport)# template client-ssl SSLInsight_
ClientSide
ACOS-Inside(config-slb vserver-vport)# template http insert_port
```

Pre-Filtering ICAP Traffic

In some scenarios, you can control the traffic forwarded to ICAP and filter traffic to bypasses ICAP.

- Allowed HTTP methods

The **allowed-http-methods** command is a REQMOD template option. It specifies the HTTP traffic methods that are forwarded to ICAP servers. By default, all methods are forwarded.

- Minimum payload size

The **min-payload-size** command is a REQMOD and RESPMOD template option. It specifies the smallest payload size that is forwarded to ICAP servers. By default, payloads that are smaller than 4096 bytes bypass ICAP.

- Disable ICAP Templates

The **disable-reqmod-icap** and **disable-respmod-icap** commands are options in the forward-policy 'destination' rule module. When an HTTP request matches the destination rule, it disables the REQMOD and/or RESPMOD ICAP templates bound to the virtual port. See [Disabling ICAP Templates](#).

Disabling ICAP Templates

This section describes how to configure the 'disable ICAP templates' pre-filtering option for ICAP templates. This feature is available as an additional action in the advanced mode of forward-policy template.

For more information on the advanced mode of forward-policy template, see *Application Delivery Controller Guide*.

CLI Configuration

1. Configure the ICAP server and SLB templates ICAP as described in the [CLI Configuration](#).
2. Enable the advance match feature in the forward-policy module and configure the forward actions.

NOTE: Enabling the advanced match feature will deactivate all the other kinds of destination rules.

The **TEST** forward-policy is enabled with the advance mode and **forward-to-internet** is the forwarding action to the **HTTP** service group.

```
ACOS(config)# slb template policy TEST
ACOS(config-policy)# forward-policy
ACOS(config-policy-forward-policy)# enable-adv-match
ACOS(config-policy-forward-policy)# action FWD
ACOS(config-policy-forward-policy-action)# forward-to-internet NEXT_4
snat FP_SSII
ACOS(config-policy-forward-policy-action)# log
```

3. Configure the advance match conditions and/or bind the class-list to the destination rule. Additionally, configure the forwarding action.

The **adv-match 1000** destination rule matches the content-length range of the HTTP request. If the length larger than 1M bytes, the REQMOD and RESPMOD ICAP templates bound to the virtual port are disabled.

```
ACOS(config-policy-forward-policy)# source rule-src-dst
ACOS(config-policy-forward-policy-source)# match-any
ACOS(config-policy-forward-policy-source)# destination adv-match 1000
ACOS(config-policy-forward-policy-source-...)# match-http-content-
length-range 1048576 to 0
ACOS(config-policy-forward-policy-source-...)# disable-reqmod-icap
ACOS(config-policy-forward-policy-source-...)# disable-respmod-icap
ACOS(config-policy-forward-policy-source-...)# action FWD
```

NOTE: Each destination rule must contain at least one forwarding 'action' (to-internet, to-service-group, to-proxy, or drop) or a 'dual-stack-action.'

4. Bind the SLB template policy to the HTTP virtual port.

```
ACOS(config)# slb virtual-server VS1 192.168.0.121
ACOS(config-slb vserver)# port 3128 http
ACOS(config-slb vserver-vport)# template policy TEST
ACOS(config-slb vserver-vport)# template reqmod-icap REQMOD_abcd
ACOS(config-slb vserver-vport)# template respmod-icap RESPMOD_abcd
```

Limitation

- You cannot configure a rule to disable only the ICAP templates.

Configuring ACOS Logging

The following steps provision ACOS logging in the ICAP templates, `RESPMOD_abcd` and `REQMOD_abcd`:

1. Create the logging template.

```
ACOS-Inside(config)# slb template logging log-template
ACOS-Inside(config-logging)# local-logging 1
```

2. Bind the logging template to the ICAP template.

```
ACOS-Inside(config)# slb template reqmod-icap REQMOD_abcd
ACOS-Inside(config-reqmod-icap)# template logging log-template

ACOS-Inside(config)# slb template respmod-icap RESPMOD_abcd
ACOS-Inside(config-respmod-icap)# template logging log-template
```

3. Configure the ICAP service URL. You have two choices.

- Use TCP port 1344 for a non-secure connection.

```
ACOS-Inside(config)# slb template reqmod-icap REQMOD_abcd
ACOS-Inside(config-respmod-icap)# service-url
icap://dlpserver:1344/reqmod
```

- Use TCP 11344 for a secure ICAP connection.

```
ACOS-Inside(config)# slb template reqmod-icap Secure_ICAP_Req
ACOS-Inside(config-reqmod-icap)# service-url
icap://dlpserver:11344/reqmod
```

Log Example

The following two logs provide an of an ICAP transaction between an ACOS TH5430 and a RESPMOD server. Web logging is described in detail in the “Web Logging for HTTP and RAM Caching” section of the Application Delivery and Server Load Balancing Guide.

```
CEF:1|A10|TH5430S|4.1.0|ES|Feb 01 2016 08:18:42|RESPONSE|2|src=40.36.1.176
spt=55906 dst=40.36.108.108 Status:200 user:(null) req="POST
https://clients1.google.com:443/tbproxy/af/query?client=Google%20Chrome
HTTP/1.1 " 0 msg="RESPMOD"

CEF:1|A10|TH5430S|4.1.0|ES|Feb 01 2016 08:18:42|REQUEST|2|src=40.36.1.176
spt=55906 dst=40.36.108.108 Sent user:(null) req="POST
https://clients1.google.com:443/tbproxy/af/query?client=Google%20Chrome
HTTP/1.1 " 0 msg="RESPMOD"
```

Single Virtual Port Deployment

This section describes how to deploy SSLi using a single virtual port instead of the conventional deployment that involves two VIPs or virtual ports. The single-port approach simplifies the configuration and reduces interface utilization on the device. This deployment is beneficial in scenarios where you need to manipulate HTTP headers within HTTPS traffic exclusively, without the need for additional traffic inspection.

Supported Features

- All basic SSLi functionalities that are available in the client-SSL template.
- SSLi in L2 and L3 mode for both static and dynamic ports.
- Explicit and Transparent proxy with proxy chaining.
- Software TLS1.3, QAT, and old N5 SSL modules.

Limitation

The new N5 SSL module is not supported.

The following topics are covered:

Key Considerations	452
Basic SSLi Deployment	453
Explicit Proxy Deployment	458

Key Considerations

This section describes the key considerations for deploying SSLi using a single virtual port.

- ACOS utilizes only the client-SSL configuration to process and forward the traffic. The server-SSL configuration is ignored (except `forward-proxy-enable` setting).

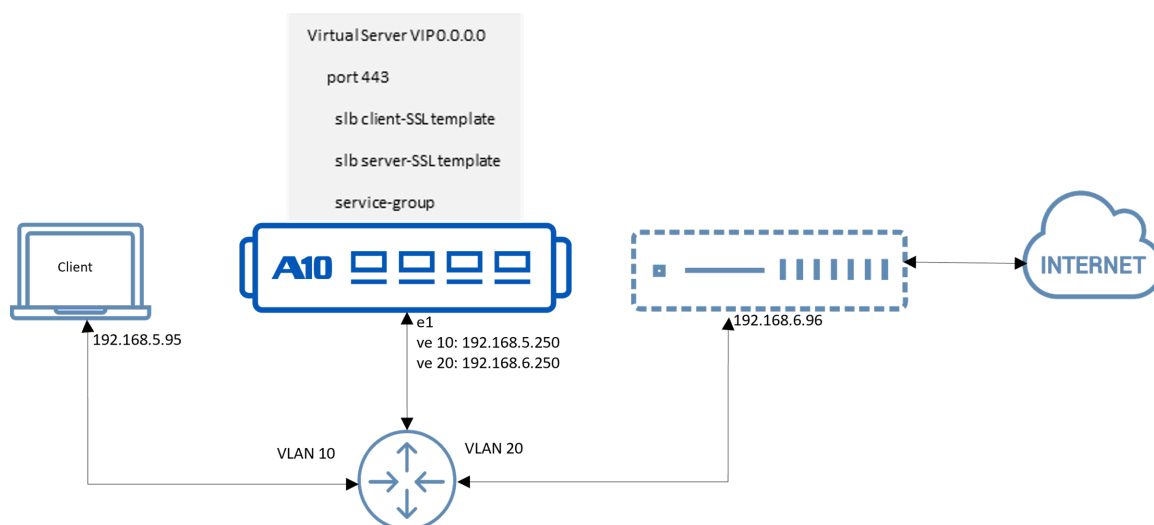
- When using aFlex scripts, the aFlex commands `SSL::enable serverside` and `SSL::template serverside <name>` cannot be used in an aFlex script that is bound to the same virtual port.
- All show commands available for SSLi apply to a single virtual port deployment. For more information, see *Command Line Interface Reference*.

Basic SSLi Deployment

In this deployment, the SSLi solution can be configured with a single ACOS device on only one virtual port.

[Figure 41](#) show the topology of basic SSLi deployment with single virtual port.

Figure 41 : A Single Port Deployment for SSLi



The following section describes the traffic flows:

- The traffic from the client network is sent to the ACOS device on the e1 interface (ve 10).
- After decrypting the incoming traffic from the client, the ACOS does not send the decrypted traffic to the server immediately. Instead, the decrypted traffic is processed internally to apply security functionalities provided by ACOS, such as URL filtering and other traffic filtering features.

- The configuration in the client SSL template is used to learn the server's certificate, and to connect the server for traffic forwarding. ACOS ignores the server-SSL template configuration when only one service-group is used.
- The ACOS device re-encrypts the traffic and forwards it to the external gateway on interface e2 (ve 20).

CLI Configuration

This section describes the basic configuration guidelines and examples for deploying SSLi using a single virtual port.

1. Configure the network interface, VLANs, and IP Addresses configuration.

```
ACOS(config)# interface management
ACOS(config-if:management)# ip address 192.168.1.10 255.255.255.0
ACOS(config-if:management)# ip default-gateway 192.168.1.1
ACOS(config-if:management)# exit

ACOS(config)# interface ethernet 1
ACOS(config-if:ethernet:1)# enable
ACOS(config-if:ethernet:1)# ip allow-promiscuous-vip
ACOS(config-if:ethernet:1)# exit

ACOS(config)# vlan 10
ACOS(config-vlan:10)# tagged ethernet 1
ACOS(config-vlan:10)# router-interface ve 10
ACOS(config-vlan:10)# exit

ACOS(config) #vlan 20
ACOS(config-vlan:20)# tagged ethernet 1
ACOS(config-vlan:20)# router-interface ve 20
ACOS(config-vlan:20)# exit

ACOS(config)# interface ve 10
ACOS(config-if:ve10)# ip address 192.168.5.250 255.255.255.0
ACOS(config-if:ve10)# ip allow-promiscuous-vip
ACOS(config-if:ve10)# exit
```

```
ACOS(config)# interface ve 20
ACOS(config-if:ve10)# ip address 192.168.6.250 255.255.255.0
ACOS(config-if:ve20)# exit
```

2. Configure a default route to the private network. Optionally, configure the source-NAT pool of IP addresses.

```
ACOS(config)# ip route 172.20.16.0 /24 192.168.1.1
ACOS(config)# ip nat pool pool_ssli 192.168.100.100 192.168.100.100
netmask /24
```

3. Configure the client-SSL template called **cl_ssl**, and server-SSL template called **sr_ssl**. Enable the forward proxy services on both templates. Additionally, configure the required SSLi features in the templates.

```
ACOS(config)# slb template client-ssl cl_ssl
ACOS(config-client ssl)# forward-proxy-ca-certificate Cert123.pem key
key123
ACOS(config-client ssl)# forward-proxy-enable
ACOS(config-client ssl)# session-cache-size 128
ACOS(config-client ssl)# session-cache-timeout 2000
ACOS(config-client ssl)# exit

ACOS(config)# slb template server-ssl sr_ssl
ACOS(config-server ssl)# forward-proxy-enable
ACOS(config-server ssl)# exit
```

4. Configure a security device as a real server called **GW** and its ports.

```
ACOS(config)# slb server GW 192.168.6.96
ACOS(config-real server)# port 0 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit
ACOS(config-real server)# port 443 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit
```

5. Configure at least one service-group called **sg_TCP_443** with one or more members.

```
ACOS(config)# slb service-group sg_TCP_443 tcp
ACOS(config-slb svc group)# member sgl 443
```

```
ACOS(config-slb svc group-member:443) exit
```

6. Configure at least one virtual server **SSLi_inside** with virtual port **443**. Bind the service-group, client-SSL, and server-SSL templates to the same virtual port.

```
ACOS(config)# slb virtual-server SSLi_inside 0.0.0.0
ACOS(config-slb vserver)# port 443 https
ACOS(config-slb vserver-vport)# source-nat pool pool_ssli
ACOS(config-slb vserver-vport)# service-group sg_TCP_443
ACOS(config-slb vserver-vport)# template client-ssl cl_ssl
ACOS(config-slb vserver-vport)# template server-ssl sr_ssl
ACOS(config-slb vserver-vport)# no-dest-nat port-translation
ACOS(config-slb vserver-vport)# exit
```

Consolidated Configuration

This section displays the consolidated configuration for basic SSLi deployed in single virtual port.

```
ACOS#show running-config
!Current configuration: 2333 bytes
!Configuration last updated at 17:03:06 PDT Fri May 19 2017
!Configuration last saved at 14:15:38 PDT Wed May 17 2017
!64-bit Advanced Core OS (ACOS) version 6.0.3, build 28 (Nov-12-2023,04:15)
!
vlan 10
  tagged ethernet 1
  router-interface ve 10
!
vlan 20
  tagged ethernet 1
  router-interface ve 20
!
interface management
  ip address 192.168.1.10 255.255.255.0
  ip default-gateway 192.168.1.1
!
interface ethernet 1
  enable
```

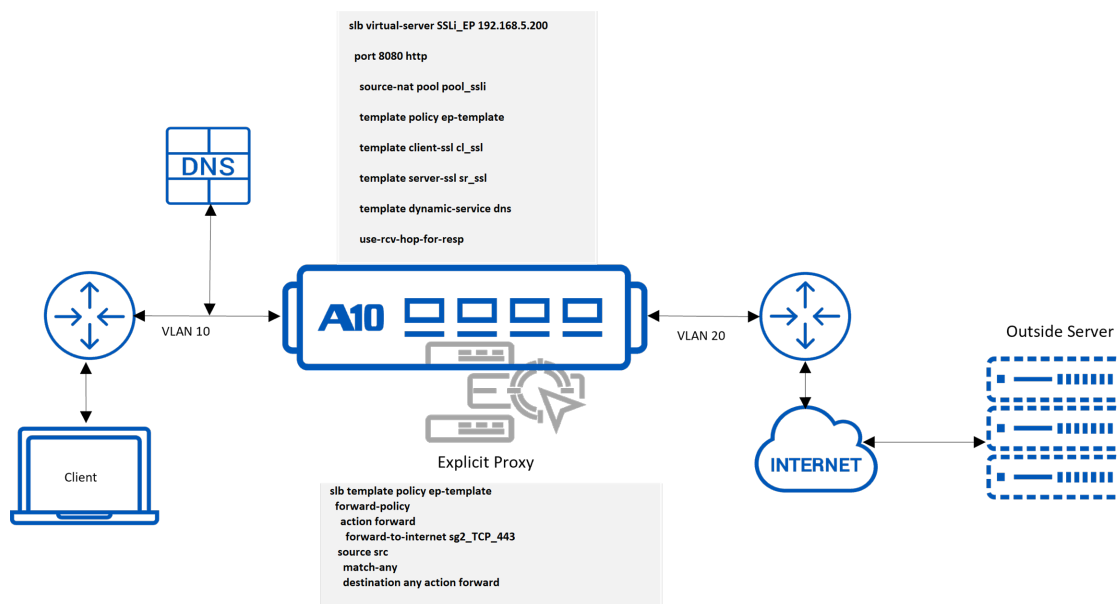


```
ip allow-promiscuous-vip
!
interface ve 10
  ip address 192.168.5.250 255.255.255.0
  ip allow-promiscuous-vip
!
interface ve 20
  ip address 192.168.6.250 255.255.255.0
!
ip route 172.20.16.0 /24 192.168.1.1
!
ip nat pool pool_ssli 192.168.100.100 192.168.100.100 netmask /24
!
slb template client-ssl cl_ssl
  forward-proxy-ca-certificate Cert123.pem key key123
  forward-proxy-enable
  session-cache-size 128
  session-cache-timeout 2000
!
slb template server-ssl sr_ssl
  forward-proxy-enable
!
slb server GW 192.168.6.96
  port 0 tcp
  health-check-disable
  port 443 tcp
  health-check-disable
!
slb service-group sg_TCP_443 tcp
  member sg1 443
!
slb virtual-server SSLi_inside 0.0.0.0
  port 443 https
  service-group sg_TCP_443
  source-nat pool pool_ssli
  template client-ssl cl_ssl
  template server-ssl sr_ssl
  no-dest-nat port-translation
!
```

Explicit Proxy Deployment

In this deployment example, the SSLi and explicit proxy can simultaneously be configured on the same virtual port. See [Figure 42](#).

Figure 42 : SSLi and Explicit Proxy Deployment on Single Virtual Port



The traffic flow is similar to the [Basic SSLi Deployment](#), with the exception of two distinct service-groups configured in the explicit/transparent proxy, each with the same members. One service-group is bound to the SLB template policy, while the other is bound to the virtual port. This results in two service-group members and two SSL contexts, one for certificate learning and the other for forwarding traffic.

This implementation allows separate configurations for certificate learning and traffic forwarding. It helps to mitigate potential conflicts in configurations, such as those arising from identical client-SSL and server-SSL settings (e.g., using the same certificate or SSL versions).

CLI Configuration

This section describes the explicit proxy configuration guidelines and examples for deploying SSLi on a single virtual port.

1. Configure the network interface, VLANs, and IP Addresses configuration.

```
ACOS(config)# interface management
ACOS(config-if:management)# ip address 192.168.1.10 255.255.255.0
ACOS(config-if:management)# ip default-gateway 192.168.1.1
ACOS(config-if:management)# exit

ACOS(config)# interface ethernet 1
ACOS(config-if:ethernet:1)# enable
ACOS(config-if:ethernet:1)# exit

ACOS(config)# vlan 10
ACOS(config-vlan:10)# tagged ethernet 1
ACOS(config-vlan:10)# router-interface ve 10
ACOS(config-vlan:10)# exit

ACOS(config) #vlan 20
ACOS(config-vlan:20)# tagged ethernet 1
ACOS(config-vlan:20)# router-interface ve 20
ACOS(config-vlan:20)# exit

ACOS(config)# interface ve 10
ACOS(config-if:ve10)# ip address 192.168.5.250 255.255.255.0
ACOS(config-if:ve10)# exit
ACOS(config)# interface ve 20
ACOS(config-if:ve10)# ip address 192.168.6.250 255.255.255.0
ACOS(config-if:ve20)# exit
```

2. Configure a default route to the private network. Additionally, the source-NAT pool of IP addresses required by the **forward-to-internet** action.

```
ACOS(config)# ip route 172.20.16.0 /24 192.168.1.1
ACOS(config)# ip nat pool pool_ssli 192.168.100.100 192.168.100.100
netmask /24
```

3. Configure the client-SSL template called **cl_ssl**, and server-SSL template called **sr_ssl**. Enable the forward proxy services on both templates. Additionally, implement the required SSLi features in the templates.

```
ACOS(config)# slb template client-ssl cl_ssl
ACOS(config-client ssl)# forward-proxy-ca-certificate Cert123.pem key
key123
ACOS(config-client ssl)# forward-proxy-enable
ACOS(config-client ssl)# session-cache-size 128
ACOS(config-client ssl)# session-cache-timeout 2000
ACOS(config-client ssl)# exit

ACOS(config)# slb template server-ssl sr_ssl
ACOS(config-server ssl)# forward-proxy-enable
ACOS(config-server ssl)# exit
```

4. Configure a dynamic service template called **d1** that contain a default DNS server.

```
ACOS_decrypt(config)# slb template dynamic-service d1
ACOS_decrypt(config-dynamic-service)# dns server 192.168.1.110
```

5. Configure a security device as a real server called **GW** and its ports.

```
ACOS(config)# slb server GW 192.168.6.96
ACOS(config-real server)# port 0 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit
ACOS(config-real server)# port 443 tcp
ACOS(config-real server-node port)# health-check-disable
ACOS(config-real server-node port)# exit
```

6. Configure two service-groups called **sg1_TCP_443** and **sg2_TCP_443 tcp** with same members.

```
ACOS(config)# slb service-group sg1_TCP_443 tcp
ACOS(config-slb svc group)# member sg1 443
ACOS(config-slb svc group)# exit
ACOS(config)# slb service-group sg2_TCP_443 tcp
ACOS(config-slb svc group)# member sg1 443
ACOS(config-slb svc group)# exit
```

7. Define the policy template called **ep-template** for the explicit proxy. Enable the **forward-to-internet** option and bind the service group **sg2_TCP_443 tcp**.

```
ACOS(config)# slb template policy ep-template
ACOS(config-policy)# forward-policy
ACOS(config-policy-forward-policy)# action forward
ACOS(config-policy-forward-policy-action)# forward-to-internet sg2_TCP_
443
ACOS(config-policy-forward-policy-action)# exit
ACOS(config-policy-forward-policy)# action GAMBLE
ACOS(config-policy-forward-policy-action)# drop
ACOS(config-policy-forward-policy-action)# drop-redirect-url
http://192.168.1.115
ACOS(config-policy-forward-policy-action)# exit
ACOS(config-policy-forward-policy)# action GAME
ACOS(config-policy-forward-policy-action)# drop
ACOS(config-policy-forward-policy-action)# drop-redirect-url
https://www.apple.com
ACOS(config-policy-forward-policy-action)# exit
```

8. Configure at least one virtual server **SSLi_EP** with virtual port **8080**. Bind the policy, client-SSL, server-SSL templates.

```
ACOS(config)# slb virtual-server SSLi_EP 192.168.5.200
ACOS(config-slb vserver)# port 8080 http
ACOS(config-slb vserver-vport)# source-nat pool pool_ssli
ACOS(config-slb vserver-vport)# template policy ep-template
ACOS(config-slb vserver-vport)# template client-ssl cl_ssl
ACOS(config-slb vserver-vport)# template server-ssl sr_ssl
ACOS(config-slb vserver-vport)# template dynamic-service d1
ACOS(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS(config-slb vserver-vport)# exit
```

NOTE:

- By default, in SSLi Single Virtual Port Deployment, when ACOS connects to the server, the TLS profile(e.g. TLS version) follows client-ssl template.
- If you want to use different TLS profiles on the client and server sides, you can bind a different service group to “non-ssl-bypass service-group” than “template policy”.

This applies server-ssl template to the server-side connection.

```
slb service-group sg1_TCP_443 tcp
    member gw1 443    <<< same member as sg2_TCP_443 tcp but
different service group
slb service-group sg2_TCP_443 tcp
    member gw1 443
slb template server-ssl sr_ssl
    forward-proxy-enable
    version 34 33
slb template client-ssl cl_ssl
    forward-proxy-ca-certificate Cert123.pem key key123
    forward-proxy-enable
    version 33 33
    non-ssl-bypass service-group sg1_TCP_443 tcp    <<< bind a
service group sg1_TCP_443 tcp to “non-ssl-bypass service-
group”

slb template policy ep-template
    forward-policy
        action forward
            forward-to-internet sg2_TCP_443
        source any
        match-any
        destination any action forward
```

Consolidated Configuration

This section displays the consolidated configuration for explicit proxy and SSLi deployed in single virtual port.

```
ACOS#show running-config
!Current configuration: 2333 bytes
!Configuration last updated at 17:03:06 PDT Fri May 19 2017
!Configuration last saved at 14:15:38 PDT Wed May 17 2017
!64-bit Advanced Core OS (ACOS) version 6.0.3, build 28 (Nov-12-2023,04:15)
!
vlan 10
    tagged ethernet 1
    router-interface ve 10
!
vlan 20
    tagged ethernet 1
    router-interface ve 20
!
interface management
    ip address 192.168.1.10 255.255.255.0
    ip default-gateway 192.168.1.1
!
interface ethernet 1
    enable
!
interface ve 10
    ip address 192.168.5.250 255.255.255.0
!
interface ve 20
    ip address 192.168.6.250 255.255.255.0
!
ip route 172.20.16.0 /24 192.168.1.1
!
ip nat pool pool_ssli 192.168.100.100 192.168.100.100 netmask /24
!
slb template server-ssl sr_ssl
    forward-proxy-enable
!
```

```
slb server GW 192.168.6.96
    port 0 tcp
        health-check-disable
    port 443 tcp
        health-check-disable
!
slb service-group sg1_TCP_443 tcp
    member GW 443
!
slb service-group sg2_TCP_443 tcp
    member GW 443
!
slb template client-ssl cl_ssl
    forward-proxy-ca-certificate Cert123.pem key key123
    forward-proxy-enable
!
slb template policy ep-template
    forward-policy
        action forward
            forward-to-internet sg2_TCP_443
        source any
        match-any
        destination any action forward
!
slb template dynamic-service d1
    dns server 192.168.1.110
!
slb virtual-server SSLi_EP 192.168.5.200
    port 8080 http
    template policy ep-template
    use-rcv-hop-for-resp
    template dynamic-service d1
    template client-ssl cl_ssl
    template server-ssl sr_ssl
!
!
end
```


SSLi with IPv6 Deployment

ACOS supports SSLi IPv6 deployment in a single ACOS device with two partitions. Two partitions are required for SSLi in this deployment, one to decrypt SSL traffic and the second to encrypt SSL traffic.

Although A10 Networks supports a number of different types of SSLi deployments, with each deployment supporting different SSLi features, the overall steps for configuring SSLi for each deployment are the same.

NOTE: If you are new to SSLi, it is recommended that you first understand the IPv4 static port deployment for both GUI and CLI discussed in [Two ACOS Devices, Each With Single Partition Deployment](#).

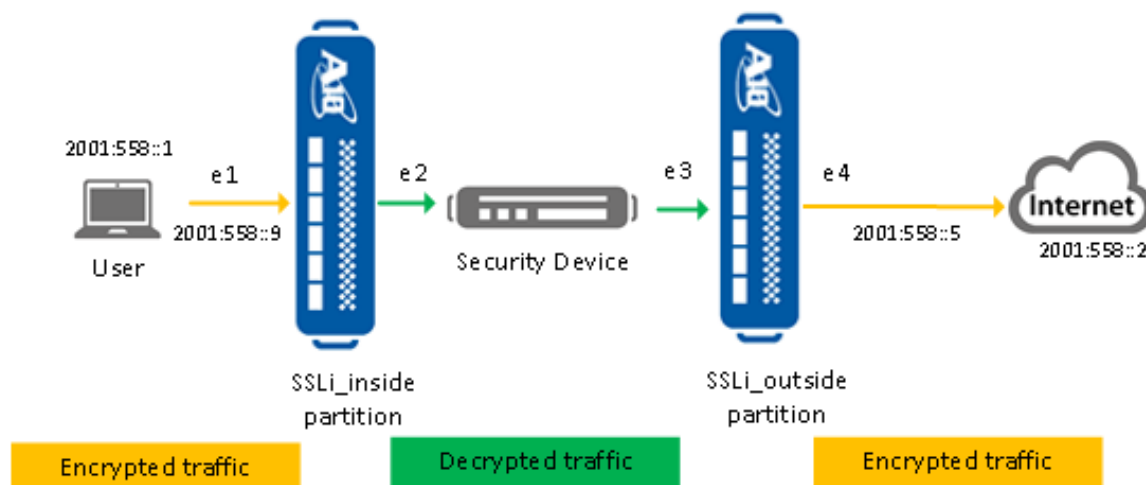
The following topics are covered:

Overview	466
Prerequisites	467
CLI Configuration	467

Overview

In the sample deployment as shown in [Figure 43](#), the client device is connected to the SSLi solution, which is then connected to the external gateway. The SSLi solution consists of a single ACOS device and a single security device. The ACOS device is connected to the client with a partition called **SSLi_inside**. The ACOS device is also connected to the external gateway with a partition called **SSLi_outside**.

Figure 43 : Sample Topology for SSLi for IPV6 Deployment



The following steps provide an overview of the SSLi process:

1. The client sets up an SSLi connection with **SSLi_inside** and sends an encrypted request.
2. **SSLi_inside** selects a traffic inspection device, decrypts the request, and sends the request over a TCP connection to the traffic inspection device.
3. The traffic inspection device inspects the request data.
4. **SSLi_outside** encrypts the request and sends it to the outside server.
5. The server sends the encrypted reply.
6. **SSLi_outside** decrypts the reply and sends it back to the same traffic inspection device.

7. If the reply traffic is allowed by the traffic inspection device, the reply is forwarded to **SSLi_inside**.
8. **SSLi_inside** encrypts the reply and sends it to the client.

Prerequisites

To deploy the SSLi solution on a single ACOS device with two partitions for IPv6 addressing, the following are the prerequisites:

- A10 Networks Advanced Core Operating System (ACOS®) 4.1.4-P3 or higher.
- Supported A10 Thunder or vThunder device(s).
- Security appliance or ICAP-based (RFC3507) antivirus or DLP solution.
- A self-signed certificate or a certification authority (CA) certificate with a known private key.

NOTE: If not already provisioned, push an internal PKI CA root certificate to all the client machines.

- The ACOS device supports both CLI and GUI for configuration. Change the default management port IP address for GUI or CLI access.
- Two partitions are required for SSLi for IPv6 addressing, one to decrypt SSL traffic and the second to encrypt SSL traffic. Make sure that you are on the correct partition when creating configurations.
- In a single device solution, use the command `system ve-mac-scheme system-mac` to avoid MAC address duplication.

CLI Configuration

Perform the following steps:

1. Follow the prerequisites discussed in [Overview](#).
2. To avoid a duplicate MAC address because of the VLAN that is shared, add the

global command of **system ve-mac-scheme system-mac** in the shared partition:

```
ACOS(config)# system ve-mac-scheme system-mac
```

3. Create the SSLi_inside and SSLi_outside partitions by running the following commands:

```
ACOS(config)# partition SSLi_outside id 1 application-type adc
ACOS(config-partition: SSLi_outside)# exit
ACOS(config)# active-partition SSLi_outside
ACOS[SSLi_outside](config)#
ACOS[SSLi_outside](config)# active-partition shared
ACOS(config)# partition SSLi_inside id 2 application-type adc
ACOS[SSLi_inside](config)# exit
```

4. Specify the DNS addressing by running the following command:

```
ACOS(config)# ip dns primary 10.5.3.1
```

5. Specify the management address and external gateway by running the following commands:

```
ACOS(config)# interface management
ACOS(config-if:management)#ip address 10.6.23.65 255.255.255.0
ACOS(config-if:management)#ip default-gateway 10.6.22.1
ACOS(config-if:management)# exit
```

ACOS_decrypt Configuration

The following topics are covered:

Step 1. Configuring the Network VLANs	469
Step 2. Configuring the Network IP Addresses	469
Step 3. Configuring the SSLi Services	470
Step 4. Configuring the SSLi Service Groups	470
Step 5. Configuring the Virtual Server	471
Consolidated Configuration for SSLi_Inside	473

Step 1. Configuring the Network VLANs

1. Configure the default VLAN. Bind ethernet ports 1 and 2 to the VLAN. Also, bind a virtual interface ve to the VLAN. In this example, a default VLAN of 850 is configured.

```
SSLi_inside(config)# vlan 850
SSLi_inside(config-vlan:850)# untagged ethernet 1 to 2
SSLi_inside(config-vlan:850)# router-interface ve 850
SSLi_inside(config-vlan:850)# exit-module
```

2. Enable the ethernet interfaces 1 and 2 that are associated with the VLAN:

```
SSLi_inside(config)# interface ethernet 1
SSLi_inside(config-if:ethernet:1)# enable
SSLi_inside(config-if:ethernet:1)# cpu-process
SSLi_inside(config-if:ethernet:1)# exit-module

SSLi_inside(config)# interface ethernet 2
SSLi_inside(config-if:ethernet:2)# enable
SSLi_inside(config-if:ethernet:2)# cpu-process
SSLi_inside(config-if:ethernet:2)# exit-module
```

3. Verify the operational state of the interfaces by running the show interfaces command.

```
SSLi_inside(config)# show interfaces brief
```

Step 2. Configuring the Network IP Addresses

Associate the IPv6 address with interface ve 850. Also, specify the IP routes.

```
SSLi_inside(config)# interface ve 850
SSLi_inside(config-if:ve850)# ip allow-promiscuous-vip
SSLi_inside(config-if:ve850)# ipv6 address 2001:558:3dc:1::9/127
SSLi_inside(config-if:ve850)# exit-module
SSLi_inside(config)# ipv6 route ::/0 2001:558:3dc:1::8
```

Step 3. Configuring the SSLi Services

1. Create a client SSL template **cl_ssl_ipv6** with forward-proxy enable configured. This configuration enables the **SSLi_inside** device to proxy for the remote SSL servers and bring up SSL sessions with the clients. Also, configure the correct service group for non-SSLi traffic.

```
SSLi_inside(config)# slb template client-ssl cl_ssl_ipv6
SSLi_inside(config-client ssl)# forward-proxy-ca-certificate
Cert123.pem key key123
SSLi_inside(config-client ssl)# forward-proxy-enable
SSLi_inside(config-client ssl)# exit-module
```

2. Configure a real server called **fw1_ipv6** with the IP address **2001:558:3dc:1::8**. Bind **fw1_ipv6** interface to TCP port 8080 so that **SSLi_inside** forwards decrypted SSLi over VLAN 850 to the security device. All other UDP and TCP traffic is forwarded by using the wildcard ports **port 0 tcp** and **port 0 udp**.

```
SSLi_inside(config)# slb server fw1_ipv6 2001:558:3dc:1::8

SSLi_inside(config-real server)# port 8080 tcp
SSLi_inside(config-real server-node port)# health-check-disable
SSLi_inside(config-real server-node port)# exit

SSLi_inside(config-real server)# port 0 tcp
SSLi_inside(config-real server-node port)# health-check-disable
SSLi_inside(config-real server-node port)# exit

SSLi_inside(config-real server)# port 0 udp
SSLi_inside(config-real server-node port)# health-check-disable
SSLi_inside(config-real server-node port)# exit
```

Step 4. Configuring the SSLi Service Groups

1. Configuring the SSLi service groups enable you to manage how the different types of traffic coming from the clients is handled by **SSLi_inside**.
2. Create a service group named **sg_ssli_ipv6_intercept** for decrypted SSL traffic.

```
SSLi_inside(config)# slb service-group sg_ssli_ipv6_intercept tcp
SSLi_inside(config-slb svc group)# member fw1_ipv6 8080
SSLi_inside(config-slb svc group)# exit-module
```

3. For the non-HTTPS traffic that is to be bypassed, configure three other service groups called **sg_ssli_ipv6_tcp**, **sg_ssli_ipv6_others**, and **sg_ssli_ipv6_udp**.

```
SSLi_inside(config)# slb service-group sg_ssli_ipv6_tcp tcp
SSLi_inside(config-slb svc group)# member fw1_ipv6 0
SSLi_inside(config-slb svc group)# exit-module

SSLi_inside(config)# slb service-group sg_ssli_ipv6_others udp
SSLi_inside(config-slb svc group)# member fw1_ipv6 0
SSLi_inside(config-slb svc group)# exit-module

SSLi_inside(config)# slb service-group sg_ssli_ipv6_udp udp
SSLi_inside(config-slb svc group)# member fw1_ipv6 0
SSLi_inside(config-slb svc group)# exit-module
```

Step 5. Configuring the Virtual Server

1. Configure the access list to permit all IPv6 traffic on VLAN 850 on ethernet 1. You must bind this ACL to the virtual server that you are going to create in the next step.

```
SSLi_inside(config)# ipv6 access-list ipv6-decrypt
SSLi_inside(config-access-list:ipv6-decrypt)# permit ipv6 any any vlan
850 ethernet 1
SSLi_inside(config-access-list:ipv6-decrypt)# exit-module
```

2. Create a virtual server called **ssli_ipv6_decryption** and associate it to the wildcard outbound VIP to intercept traffic from clients. The following virtual ports are configured on this VIP:
 - 443 (HTTPS)—Intercepts SSL-encrypted traffic from the clients. Port 443 on the wildcard outbound VIP is bound to a service group called **sg_ssli_ipv6_intercept** that contains the path through the security device to the **SSLi_outside** device. Consider the following information:
 - The destination NAT is disabled, and **SSLi_inside** does not change the source or destination IP addresses of the traffic.
 - Port translation is enabled and required because the ACOS device must change the destination protocol port from 443 to the port number on which the

security device listens for traffic.

- The client-SSL template **ssl cl_ssl_ipv6** is bound to the virtual port 443 HTTPS.
- 0 (TCP), 0 (UDP), and 0 (Others) —Intercepts the client traffic that is not HTTPS in the following ways: The TCP port intercepts all other TCP traffic from clients. The TCP wildcard port is bound to a TCP service group called **sg_ssli_ipv6_tcp** that contains the path through the security device to the **SSLi_outside** device.
- The UDP port intercepts all other UDP traffic from clients. The UDP wildcard port is bound to a UDP service group called **sg_ssli_ipv6_udp** that contains the path through the security device to the **SSLi_outside** device.
- The Others port intercepts the client traffic types that are not listed. The Others port is for IP traffic not included by the TCP and UDP all-ports sections. The Others wildcard port is bound to a UDP service group called **sg_ssli_ipv6_others** that contains the path through the security device to the **SSLi_outside** device.
- The destination NAT and port translation are disabled for the aforementioned ports.

NOTE: If you replace a certificate and key in a client-SSL or server-SSL template, you must unbind the template from the virtual ports that use it and then rebind the template to the virtual ports.

```
SSLi_inside(config)# slb virtual-server ssli_ipv6_decryption :: ipv6-acl
ipv6-decrypt
SSLi_inside(config-slb vserver)# port 0 tcp
SSLi_inside(config-slb vserver-vport)# service-group sg_ssli_ipv6_tcp
SSLi_inside(config-slb vserver-vport)# no-dest-nat
SSLi_inside(config-slb vserver-vport)# exit

SSLi_inside(config-slb vserver)# port 0 udp
SSLi_inside(config-slb vserver-vport)# service-group sg_ssli_ipv6_udp
SSLi_inside(config-slb vserver-vport)# no-dest-nat
SSLi_inside(config-slb vserver-vport)# exit

SSLi_inside(config-slb vserver)# port 0 others
SSLi_inside(config-slb vserver-vport)# sg_ssli_ipv6_others
SSLi_inside(config-slb vserver-vport)# no-dest-nat
SSLi_inside(config-slb vserver-vport)# exit
```



```
SSLi_inside(config-slb vserver)# port 443 https
SSLi_inside(config-slb vserver-vport)# sg_ssli_ipv6_intercept
SSLi_inside(config-slb vserver-vport)# template client-ssl cl_ssl_ipv6
SSLi_inside(config-slb vserver-vport)# no-dest-nat port-translation
SSLi_inside(config-slb vserver-vport)# exit-module
```

Consolidated Configuration for SSLi_Inside

```
active-partition ssli_in
!
vlan 850
    untagged ethernet 1 to 2
    router-interface ve 850
    name ssli_in_ingress_egress
    user-tag Security,ssli_in_ingress_egress
    exit-module
!
ipv6 access-list ipv6-decrypt
    permit ipv6 any any vlan 850 ethernet 1
    exit-module
!
interface ethernet 1
    name ssli_in_ingress
    enable
    cpu-process
    user-tag Security,ssli_in_ingress
    exit-module
!
interface ethernet 2
    name ssli_in_egress
    enable
    cpu-process
    user-tag Security,ssli_in_egress
    exit-module
!
interface ve 850
    name ssli_in_ingress_egress
    user-tag Security,ssli_in_ingress_egress
    ip address 10.177.253.13 255.255.255.240
    ip allow-promiscuous-vip
    ipv6 address 2001:558:3dc:1::9/127
```

```
exit-module
!
!
ipv6 route ::/0 2001:558:3dc:1::8
!
!
slb server fw1_ipv6 2001:558:3dc:1::8
    port 0 tcp
        health-check-disable
    exit-module
    port 0 udp
        health-check-disable
    exit-module
    port 8080 tcp
        health-check-disable
    exit-module
exit-module
!
!
slb service-group sg_ssli_ipv6_intercept tcp
    member fw1_ipv6 8080
    exit-module
exit-module
!
slb service-group sg_ssli_ipv6_others udp
    member fw1_ipv6 0
    exit-module
exit-module
!
slb service-group sg_ssli_ipv6_tcp tcp
    member fw1_ipv6 0
    exit-module
exit-module
!
slb service-group sg_ssli_ipv6_udp udp
    member fw1_ipv6 0
    exit-module
exit-module
!
!
```

```
slb template client-ssl cl_ssl_ipv6
  forward-proxy-ca-certificate Cert123.pem key key123
  forward-proxy-enable
  non-ssl-bypass service-group sg_ssli_ipv6_tcp
  exit-module
!
!
slb virtual-server ssli_ipv6_decryption :: ipv6-acl ipv6-decrypt
  user-tag Security,ipv6
  port 0 tcp
    service-group sg_ssli_ipv6_tcp
    no-dest-nat
    user-tag Security,ipv6_port_0tcp
    exit-module
  port 0 udp
    service-group sg_ssli_ipv6_udp
    user-tag Security,ipv6_port_0udp
    exit-module
  port 0 others
    service-group sg_ssli_ipv6_others
    no-dest-nat
    user-tag Security,ipv6_port_0others
    exit-module
  port 443 https
    service-group sg_ssli_ipv6_intercept
    template client-ssl cl_ssl_ipv6
    no-dest-nat port-translation
    user-tag Security,ipv6_port_443https
    exit-module
  exit-module
!
end
!Current configuration: 7779 bytes
!Configuration last updated at 06:29:47 UTC Thu Aug 16 2018
!Configuration last saved at 09:12:05 UTC Wed Sep 5 2018
!
```

ACOS_encrypt Configuration

The following topics are covered:

Step 1. Configuring the Network VLANs	476
Step 2. Configuring the Network IP Addresses	476
Step 3. Configuring the SSLi Services	477
Step 4. Configuring the SSLi Service Groups	477
Step 5. Configuring the Virtual Server	478
Consolidated Configuration	479

Step 1. Configuring the Network VLANs

```
SSLi_outside(config)# vlan 860
SSLi_outside(config-vlan:860)# untagged ethernet 3 to 4
SSLi_outside(config-vlan:860)# router-interface ve 860
SSLi_outside(config-vlan:860)# exit-module

SSLi_outside(config)# interface ethernet 3
SSLi_outside(config-if:ethernet:3)# enable
SSLi_outside(config-if:ethernet:3)# cpu-process
SSLi_outside(config-if:ethernet:3)# exit-module

SSLi_outside(config)# interface ethernet 4
SSLi_outside(config-if:ethernet:4)# enable
SSLi_outside(config-if:ethernet:4)# cpu-process
SSLi_outside(config-if:ethernet:4)# exit-module
```

Verify the operational state of the interfaces by running the show interfaces command.

```
SSLi_outside(config)# show interfaces brief
```

Step 2. Configuring the Network IP Addresses

```
SSLi_outside(config)# interface ve 860
SSLi_outside(config-if:ve860)# ip allow-promiscuous-vip
SSLi_outside(config-if:ve860)# ipv6 address 2001:558:3dc:1::5/125
SSLi_outside(config-if:ve860)# exit-module
SSLi_outside(config)# ipv6 route ::/0 2001:558:3dc:1::2
```

```
SSLi_outside(config)# exit-module
```

Step 3. Configuring the SSLi Services

Create a real server called **fw2_ipv6** on **SSLi_outside**. Configure the ports.

```
SSLi_outside(config)# slb server fw2_ipv6 2001:558:3dc:1::2
SSLi_outside(config-real server)# health-check-disable
SSLi_outside(config-real server)# port 0 tcp
SSLi_outside(config-real server-node port)# health-check-disable
SSLi_outside(config-real server-node port)# exit

SSLi_outside(config-real server)# port 0 udp
SSLi_outside(config-real server-node port)# health-check-disable
SSLi_outside(config-real server-node port)# exit

SSLi_outside(config-real server)# port 443 tcp
SSLi_outside(config-real server-node port)# health-check-disable
SSLi_outside(config-real server-node port)# exit-module
```

Step 4. Configuring the SSLi Service Groups

1. Create a service group called **sg_ssli_ipv6_443** and provide a path for the intercepted HTTPS traffic by binding the service group to ports **443** of the real server **fw2_ipv6**.

```
SSLi_outside(config)# slb service-group sg_ssli_ipv6_443 tcp
SSLi_outside(config-slb svc group)# member fw2_ipv6 443
SSLi_outside(config-slb svc group)# exit-module
```

2. Create the other service groups to handle the other kinds of traffic.

```
SSLi_outside(config)# slb service-group sg_ssli_ipv6_tcp tcp
SSLi_outside(config-slb svc group)# member fw2_ipv6 0
SSLi_outside(config-slb svc group)# exit-module

SSLi_outside(config)# slb service-group sg_ssli_ipv6_others udp
SSLi_outside(config-slb svc group)# member fw2_ipv6 0
SSLi_outside(config-slb svc group)# exit-module

SSLi_outside(config)# slb service-group sg_ssli_ipv6_udp udp
```

```
SSLi_outside(config-slb svc group)# member fw2_ipv6 0
SSLi_outside(config-slb svc group)# exit-module
```

Step 5. Configuring the Virtual Server

1. Create the access lists.

```
SSLi_outside(config)#ipv6 access-list ipv6-permit
SSLi_outside(config-access-list:ipv6-permit)#permit ipv6 any any vlan
860 ethernet 3
SSLi_outside(config-access-list:ipv6-permit)# exit-module
```

2. Create the virtual server for IPv6 traffic. Associate the virtual server **ssli_ipv6_encrypt** with the **ipv6-permit** ACL thst permists all traffic on VLAN 860 on ethernet 3.

```
SSLi_outside(config)# slb virtual-server ssli_ipv6_encrypt :: ipv6-acl
ipv6-permit
SSLi_outside(config-slb vsrver)# port 0 tcp
SSLi_outside(config-slb vsrver-vport)# service-group sg_ssli_ipv6_tcp
SSLi_outside(config-slb vsrver-vport)# use-rcv-hop-for-resp
SSLi_outside(config-slb vsrver-vport)# no-dest-nat
SSLi_outside(config-slb vsrver-vport)# exit

SSLi_outside(config-slb vsrver)# port 0 udp
SSLi_outside(config-slb vsrver-vport)# service-group sg_ssli_ipv6_udp
SSLi_outside(config-slb vsrver-vport)# use-rcv-hop-for-resp
SSLi_outside(config-slb vsrver-vport)# no-dest-nat
SSLi_outside(config-slb vsrver-vport)# exit

SSLi_outside(config-slb vsrver)# port 0 others
SSLi_outside(config-slb vsrver-vport)# service-group sg_ssli_ipv6_
others
SSLi_outside(config-slb vsrver-vport)# use-rcv-hop-for-resp
SSLi_outside(config-slb vsrver-vport)# no-dest-nat
SSLi_outside(config-slb vsrver-vport)# exit

SSLi_outside(config-slb vsrver)# port 443 tcp
SSLi_outside(config-slb vsrver-vport)# service-group sg_ssli_ipv6_443
SSLi_outside(config-slb vsrver-vport)# use-rcv-hop-for-resp
SSLi_outside(config-slb vsrver-vport)# no-dest-nat
```

```
SSLi_outside(config-slb vserver-vport)# exit

SSLi_outside(config-slb vserver)# port 8080 http
SSLi_outside(config-slb vserver-vport)# service-group sg_ssli_ipv6_
encrypt
SSLi_outside(config-slb vserver-vport)# use-rcv-hop-for-resp
SSLi_outside(config-slb vserver-vport)# no-dest-nat port-translation
SSLi_outside(config-slb vserver-vport)# exit-module
```

Consolidated Configuration

```
active-partition ssli_out
!
!
vlan 860
  untagged ethernet 3 to 4
  router-interface ve 860
  name ssli_out_ingress_egress
  user-tag Security,ssli_out_ingress_egress
  exit-module
!
ipv6 access-list ipv6-permit
  permit ipv6 any any vlan 860 ethernet 3
  exit-module
!
interface ethernet 3
  name ssli_out_ingress
  enable
  cpu-process
  user-tag Security,ssli_out_ingress
  exit-module
!
interface ethernet 4
  name ssli_out_egress
  enable
  cpu-process
  user-tag Security,ssli_out_egress
  exit-module
!
interface ve 860
  name ssli_out_ingress_egress
```

```
user-tag Security,ssli_out_ingress_egress
ip allow-promiscuous-vip
ipv6 address 2001:558:3dc:1::5/125
exit-module
!
!
ipv6 route ::/0 2001:558:3dc:1::2
!
!
slb server fw2_ipv6 2001:558:3dc:1::2
health-check-disable
port 0 tcp
health-check-disable
exit-module
port 0 udp
health-check-disable
exit-module
port 443 tcp
health-check-disable
exit-module
exit-module
!
!
slb service-group sg_ssli_ipv6_encrypt tcp
member fw2_ipv6 443
exit-module
exit-module
!
slb service-group sg_ssli_ipv6_others udp
member fw2_ipv6 0
exit-module
exit-module
!
slb service-group sg_ssli_ipv6_tcp tcp
member fw2_ipv6 0
exit-module
exit-module
!
slb service-group sg_ssli_ipv6_udp udp
member fw2_ipv6 0
```



```
    exit-module
  exit-module
!
!
slb virtual-server ssli_ipv6_encrypt :: ipv6-acl ipv6-permit
  user-tag Security,ssli_out
  port 0 tcp
    service-group sg_ssli_ipv6_tcp
    use-rcv-hop-for-resp
    no-dest-nat
    user-tag Security,ssli_out_port_0tcp
    exit-module
  port 0 udp
    service-group sg_ssli_ipv6_udp
    use-rcv-hop-for-resp
    no-dest-nat
    user-tag Security,ssli_out_port_0udp
    exit-module
  port 0 others
    service-group sg_ssli_ipv6_others
    use-rcv-hop-for-resp
    no-dest-nat
    user-tag Security,ssli_out_port_0others
    exit-module
  port 443 tcp
    service-group sg_ssli_ipv6_443
    use-rcv-hop-for-resp
    no-dest-nat
    user-tag Security,ssli_out_port_443tcp
    exit-module
  port 8080 http
    service-group sg_ssli_ipv6_encrypt
    use-rcv-hop-for-resp
    no-dest-nat port-translation
    user-tag Security,ssli_out_decrypted_port_44380http
    exit-module
!
end
```

GUI Configuration

The procedures for creating the configuration for both SSLi_Inside and SSLi_Outside for single ACOS device with dual partitions for IPv6 is very similar to the procedure in [Two ACOS Devices, Each With Single Partition Deployment](#)

Follow the steps and make appropriate replacements by consulting the consolidated configurations discussed in the above section.

SSLi in VRRP-A Deployment

This section helps you understand SSL Insight in a VRRP-A deployment.

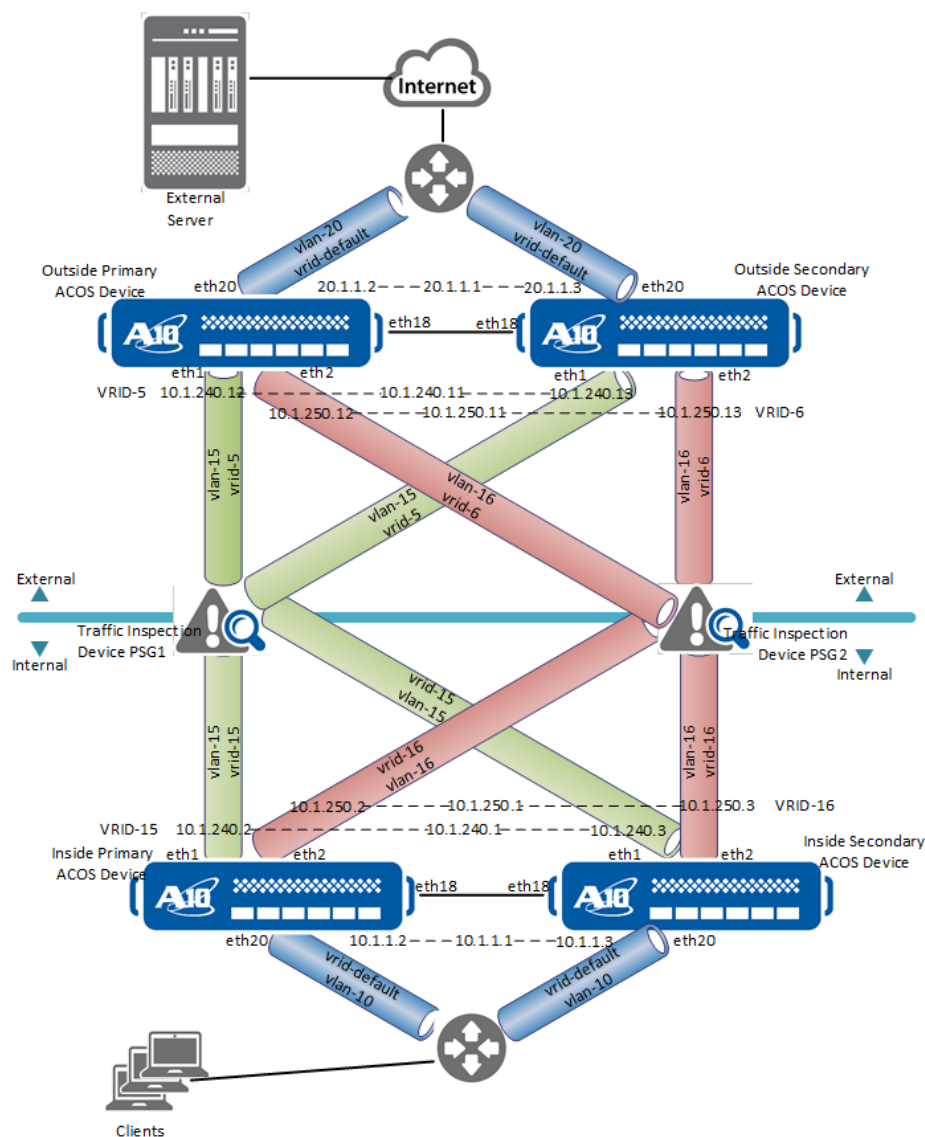
The following topics are covered:

- [Deployment Example](#) 484
- [ACOS_decrypt CLI Configuration](#) 485
- [ACOS_encrypt CLI Configuration](#) 495

Deployment Example

The following sections describe the configuration steps needed to create an example SSL Insight VRRP-A deployment. [Figure 44](#) is the topology of this example.

Figure 44 : SSL Insight Topology Example



ACOS_decrypt CLI Configuration

Inside Primary ACOS device Configuration

Hostname Configuration

```
ACOS(config)# hostname ACOS-Inside-Primary
```

Layer 2/3 Configuration

Enter the following commands to configure the VLANs:

```
ACOS-Inside-Primary(config)# vlan 10
ACOS-Inside-Primary(config-vlan:10)# untagged ethernet 20
ACOS-Inside-Primary(config-vlan:10)# router-interface ve 10
ACOS-Inside-Primary(config-vlan:10)# exit
ACOS-Inside-Primary(config)# vlan 15
ACOS-Inside-Primary(config-vlan:15)# untagged ethernet 1
ACOS-Inside-Primary(config-vlan:15)# router-interface ve 15
ACOS-Inside-Primary(config-vlan:15)# exit
ACOS-Inside-Primary(config)# vlan 16
ACOS-Inside-Primary(config-vlan:16)# untagged ethernet 2
ACOS-Inside-Primary(config-vlan:16)# router-interface ve 16
ACOS-Inside-Primary(config-vlan:16)# exit
ACOS-Inside-Primary(config)# vlan 99
ACOS-Inside-Primary(config-vlan:99)# untagged ethernet 18
ACOS-Inside-Primary(config-vlan:99)# router-interface ve 99
ACOS-Inside-Primary(config-vlan:99)# exit
```

The following commands assign IP addresses to the VEs (router interfaces) that are configured on the VLANs. Since VE 10 is connected to the clients, promiscuous VIP mode is enabled on this VE. The other VEs do not use promiscuous VIP mode in this deployment.

```
ACOS-Inside-Primary(config)# interface ve 10
ACOS-Inside-Primary(config-if:ve10)# ip address 10.1.1.2/24
ACOS-Inside-Primary(config-if:ve10)# ip allow-promiscuous-vip
ACOS-Inside-Primary(config-if:ve10)# exit
```

```

ACOS-Inside-Primary(config)# interface ve 15
ACOS-Inside-Primary(config-if:ve15)# ip address 10.1.240.2/24
ACOS-Inside-Primary(config-if:ve15)# exit
ACOS-Inside-Primary(config)# interface ve 16
ACOS-Inside-Primary(config-if:ve16)# ip address 10.1.250.2/24
ACOS-Inside-Primary(config-if:ve16)# exit
ACOS-Inside-Primary(config)# interface ve 99
ACOS-Inside-Primary(config-if:ve99)# ip address 55.1.1.1/24
ACOS-Inside-Primary(config-if:ve99)# exit

```

The following commands configure static routes to the network on the side of the outside ACOS devices that connects to the Internet. The next-hop IP address of each route is the floating IP address of a VRID on the outside ACOS devices. Specifically, these are the floating IP addresses that belong to the VRIDs for the VLANs that contain the security devices.

```

ACOS-Inside-Primary(config)# ip route 20.1.1.0 /24 10.1.240.11
ACOS-Inside-Primary(config)# ip route 20.1.1.0 /24 10.1.250.11

```

SSL Configuration

The following commands import the root CA-signed certificate used by the content servers, and the certificate's private key:

```

ACOS-Inside-Primary(config)# import cert ca.cert.pem scp:
Address or name of remote host []?192.168.1.111
User name []?admin
Password []?*****
File name [/]?ca-cert.pem
ACOS-Inside-Primary(config)# import key private-key ca.key.pem scp:
Address or name of remote host []?192.168.1.111
User name []?admin
Password []?*****
File name [/]?ca-certkey.pem

```

The following commands configure the client-SSL template:

```

ACOS-Inside-Primary(config)# slb template client-ssl SSLInsight_
ClientSide
ACOS-Inside-Primary(config-client SSL template)# forward-proxy-enable

```

```
ACOS-Inside-Primary(config-client SSL template)# forward-proxy-ca-
certificate Cert123.pem key key123
ACOS-Inside-Primary(config-client SSL template)# exit
```

Path Configuration

The following commands configure the paths through the security devices:

```
ACOS-Inside-Primary(config)# slb server PSG1_Path 10.1.240.11
ACOS-Inside-Primary(config-real server)# port 0 tcp
ACOS-Inside-Primary(config-real server-node port)# health-check-disable
ACOS-Inside-Primary(config-real server-node port)# exit
ACOS-Inside-Primary(config-real server)# port 0 udp
ACOS-Inside-Primary(config-real server-node port)# health-check-disable
ACOS-Inside-Primary(config-real server-node port)# exit
ACOS-Inside-Primary(config-real server)# port 8080 tcp
ACOS-Inside-Primary(config-real server-node port)# health-check-disable
ACOS-Inside-Primary(config-real server-node port)# exit
ACOS-Inside-Primary(config-real server)# exit
ACOS-Inside-Primary(config)# slb server PSG2_Path 10.1.250.11
ACOS-Inside-Primary(config-real server)# port 0 tcp
ACOS-Inside-Primary(config-real server-node port)# health-check-disable
ACOS-Inside-Primary(config-real server-node port)# exit
ACOS-Inside-Primary(config-real server)# port 0 udp
ACOS-Inside-Primary(config-real server-node port)# health-check-disable
ACOS-Inside-Primary(config-real server-node port)# exit
ACOS-Inside-Primary(config-real server)# port 8080 tcp
ACOS-Inside-Primary(config-real server-node port)# health-check-disable
ACOS-Inside-Primary(config-real server-node port)# exit
ACOS-Inside-Primary(config-real server)# exit

ACOS-Inside-Primary(config)# slb service-group LB_Paths_UDP udp
ACOS-Inside-Primary(config-slb svc group)# member PSG1_Path 0
ACOS-Inside-Primary(config-slb svc group)# member PSG2_Path 0
ACOS-Inside-Primary(config-slb svc group)# exit
ACOS-Inside-Primary(config)# slb service-group LB_Paths_TCP tcp
ACOS-Inside-Primary(config-slb svc group)# member PSG1_Path 0
ACOS-Inside-Primary(config-slb svc group)# member PSG2_Path 0
ACOS-Inside-Primary(config-slb svc group)# exit
ACOS-Inside-Primary(config)# slb service-group SSL tcp
```

```
ACOS-Inside-Primary(config-slb svc group)# member PSG1_Path 8080
ACOS-Inside-Primary(config-slb svc group)# member PSG2_Path 8080
ACOS-Inside-Primary(config-slb svc group)# exit
```

The following commands configure the wildcard VIP to intercept all outbound traffic that originates from the inside network:

```
ACOS-Inside-Primary(config)# access-list 100 permit ip any any vlan 10
ACOS-Inside-Primary(config)# slb virtual-server outbound_wildcard
0.0.0.0 acl 100
ACOS-Inside-Primary(config-slb vserver)# port 0 tcp
ACOS-Inside-Primary(config-slb vserver-vport)# name Inside1_in_to_out
ACOS-Inside-Primary(config-slb vserver-vport)# service-group LB_Paths_
TCP
ACOS-Inside-Primary(config-slb vserver-vport)# no-dest-nat
ACOS-Inside-Primary(config-slb vserver-vport)# exit
ACOS-Inside-Primary(config-slb vserver)# port 0 udp
ACOS-Inside-Primary(config-slb vserver-vport)# name Inside1_in_to_out_
UDP
ACOS-Inside-Primary(config-slb vserver-vport)# service-group LB_Paths_
UDP
ACOS-Inside-Primary(config-slb vserver-vport)# no-dest-nat
ACOS-Inside-Primary(config-slb vserver-vport)# exit
ACOS-Inside-Primary(config-slb vserver)# port 443 https
ACOS-Inside-Primary(config-slb vserver-vport)# name Inside1_in_to_out_
443
ACOS-Inside-Primary(config-slb vserver-vport)# service-group SSL
ACOS-Inside-Primary(config-slb vserver-vport)# template client-ssl
SSLInsight_ClientSide
ACOS-Inside-Primary(config-slb vserver-vport)# no-dest-nat port-
translation
ACOS-Inside-Primary(config-slb vserver-vport)# exit
ACOS-Inside-Primary(config-slb vserver)# exit
```

VRRP-A Configuration

The following commands specify the VRRP-A device ID for this ACOS device, add the ACOS device to VRRP-A set 1, and enable VRRP-A on the device:

```
ACOS-Inside-Primary(config)# vrrp-a common
```



```
ACOS-Inside-Primary(config-common)# device-id 1  
ACOS-Inside-Primary(config-common)# set-id 1  
ACOS-Inside-Primary(config-common)# enable  
ACOS-Inside-Primary(config-common)# exit
```

The following commands configure the VRID for the inside ACOS devices' interface with the client network:

```
ACOS-Inside-Primary(config)# vrrp-a vrid 0  
ACOS-Inside-Primary(config-vrid:0)# floating-ip 10.1.1.1  
ACOS-Inside-Primary(config-vrid:0)# blade-parameters  
ACOS-Inside-Primary(config-vrid:0-blade-parameters)# priority 200  
ACOS-Inside-Primary(config-vrid:0-blade-parameters)# tracking-options  
ACOS-Inside-Primary(config-vrid:0-blade-parameters-track...)# interface  
ethernet 1 priority-cost 60  
ACOS-Inside-Primary(config-vrid:0-blade-parameters-track...)# interface  
ethernet 2 priority-cost 60  
ACOS-Inside-Primary(config-vrid:0-blade-parameters-track...)# interface  
ethernet 20 priority-cost 60  
ACOS-Inside-Primary(config-vrid:0-blade-parameters-track...)# exit  
ACOS-Inside-Primary(config-vrid:0-blade-parameters)# exit  
ACOS-Inside-Primary(config-vrid:0)# exit
```

The following commands configure the VRID for the VLAN that contains the first security device (PSG1):

```
ACOS-Inside-Primary(config)# vrrp-a vrid 15  
ACOS-Inside-Primary(config-vrid:15)# floating-ip 10.1.240.1  
ACOS-Inside-Primary(config-vrid:15)# blade-parameters  
ACOS-Inside-Primary(config-vrid:15-blade-parameters)# priority 200  
ACOS-Inside-Primary(config-vrid:15-blade-parameters)# tracking-options  
ACOS-Inside-Primary(config-vrid:15-blade-parameters-track...)# interface  
ethernet 1 priority-cost 60  
ACOS-Inside-Primary(config-vrid:15-blade-parameters-track...)# interface  
ethernet 2 priority-cost 60  
ACOS-Inside-Primary(config-vrid:15-blade-parameters-track...)# interface  
ethernet 20 priority-cost 60  
ACOS-Inside-Primary(config-vrid:15-blade-parameters-track...)# exit  
ACOS-Inside-Primary(config-vrid:15-blade-parameters)# exit  
ACOS-Inside-Primary(config-vrid:15)# exit
```

The following commands configure the VRID for the VLAN that contains the second security device (PSG2):

```
ACOS-Inside-Primary(config)# vrrp-a vrid 16
ACOS-Inside-Primary(config-vrid:16)# floating-ip 10.1.250.1
ACOS-Inside-Primary(config-vrid:16)# blade-parameters
ACOS-Inside-Primary(config-vrid:16-blade-parameters)# priority 200
ACOS-Inside-Primary(config-vrid:16-blade-parameters)# tracking-options
ACOS-Inside-Primary(config-vrid:16-blade-parameters-track...)# interface
ethernet 1 priority-cost 60
ACOS-Inside-Primary(config-vrid:16-blade-parameters-track...)# interface
ethernet 2 priority-cost 60
ACOS-Inside-Primary(config-vrid:16-blade-parameters-track...)# interface
ethernet 20 priority-cost 60
ACOS-Inside-Primary(config-vrid:16-blade-parameters-track...)# exit
ACOS-Inside-Primary(config-vrid:16-blade-parameters)# exit
ACOS-Inside-Primary(config-vrid:16)# exit
```

The following command configures the VRRP-S interface that connects this ACOS device to its VRRP-A peer:

```
ACOS-Inside-Primary(config)# vrrp-a interface ethernet 18
ACOS-Inside-Primary(config-ethernet:18)# vlan 99
```

Inside Secondary ACOS device Configuration

The configuration on the inside secondary ACOS device is the same as the configuration on the inside primary ACOS device, except for the following device-specific parameters:

- **Hostname** – The hostname is configured with a unique value to make it simpler to identify the device.
- **VRRP-A device ID** – The ID must be unique in the set of ACOS devices that are backed up by VRRP-A (the VRRP-A set).
- **Interface IP addresses** – The VLAN IDs are the same on both ACOS devices, but the router interface on each VLAN has a unique IP address. The IP address is unique on each ACOS device.

- Priority values of the VRIDs – To specify the ACOS device's default VRRP-A role (active or backup), each VRID on this ACOS device is configured with a lower priority value than the same VRID on the inside primary ACOS device.

Hostname Configuration

```
ACOS(config)# hostname ACOS-Inside-Secondary
```

Layer 2/3 Configuration

```
ACOS-Inside-Secondary(config)# vlan 10
ACOS-Inside-Secondary(config-vlan:10)# untagged ethernet 20
ACOS-Inside-Secondary(config-vlan:10)# router-interface ve 10
ACOS-Inside-Secondary(config-vlan:10)# exit
ACOS-Inside-Secondary(config)# vlan 15
ACOS-Inside-Secondary(config-vlan:15)# untagged ethernet 1
ACOS-Inside-Secondary(config-vlan:15)# router-interface ve 15
ACOS-Inside-Secondary(config-vlan:15)# exit
ACOS-Inside-Secondary(config)# vlan 16
ACOS-Inside-Secondary(config-vlan:16)# untagged ethernet 2
ACOS-Inside-Secondary(config-vlan:16)# router-interface ve 16
ACOS-Inside-Secondary(config-vlan:16)# exit
ACOS-Inside-Secondary(config)# vlan 99
ACOS-Inside-Secondary(config-vlan:99)# untagged ethernet 18
ACOS-Inside-Secondary(config-vlan:99)# router-interface ve 99
ACOS-Inside-Secondary(config-vlan:99)# exit

ACOS-Inside-Secondary(config)# interface ve 10
ACOS-Inside-Secondary(config-if:ve10)# ip address 10.1.1.3 255.255.255.0
ACOS-Inside-Secondary(config-if:ve10)# ip allow-promiscuous-vip
ACOS-Inside-Secondary(config-if:ve10)# exit
ACOS-Inside-Secondary(config)# interface ve 15
ACOS-Inside-Secondary(config-if:ve15)# ip address 10.1.240.3
255.255.255.0
ACOS-Inside-Secondary(config-if:ve15)# exit
ACOS-Inside-Secondary(config)# interface ve 16
ACOS-Inside-Secondary(config-if:ve16)# ip address 10.1.250.3
255.255.255.0
ACOS-Inside-Secondary(config-if:ve16)# exit
ACOS-Inside-Secondary(config)# interface ve 99
ACOS-Inside-Secondary(config-if:ve99)# ip address 55.1.1.2 255.255.255.0
ACOS-Inside-Secondary(config-if:ve99)# exit
```

```
ACOS-Inside-Secondary(config)# ip route 20.1.1.0 /24 10.1.240.11
ACOS-Inside-Secondary(config)# ip route 20.1.1.0 /24 10.1.250.11
```

SSL Configuration

```
ACOS-Inside-Primary(config)# import cert ca.cert.pem scp:
Address or name of remote host []?192.168.1.111
User name []?admin
Password []?*****
File name [/]?ca-cert.pem
ACOS-Inside-Primary(config)# import key private-key ca.key.pem scp:
Address or name of remote host []?192.168.1.111
User name []?admin
Password []?*****
File name [/]?ca-certkey.pem
ACOS-Inside-Secondary(config)# slb template client-ssl SSLInsight_
ClientSide
ACOS-Inside-Secondary(config-client SSL template)# forward-proxy-enable
ACOS-Inside-Secondary(config-client SSL template)# forward-proxy-ca-
certificate Cert123.pem key key123
ACOS-Inside-Secondary(config-client SSL template)# exit
```

Path Configuration

```
ACOS-Inside-Secondary(config)# slb server PSG1_Path 10.1.240.11
ACOS-Inside-Secondary(config-real server)# port 0 tcp
ACOS-Inside-Secondary(config-real server-node port)# health-check-
disable
ACOS-Inside-Secondary(config-real server-node port)# exit
ACOS-Inside-Secondary(config-real server)# port 0 udp
ACOS-Inside-Secondary(config-real server-node port)# health-check-
disable
ACOS-Inside-Secondary(config-real server-node port)# exit
ACOS-Inside-Secondary(config-real server)# port 8080 tcp
ACOS-Inside-Secondary(config-real server-node port)# health-check-
disable
ACOS-Inside-Secondary(config-real server-node port)# exit
ACOS-Inside-Secondary(config-real server)# exit
ACOS-Inside-Secondary(config)# slb server PSG2_Path 10.1.250.11
ACOS-Inside-Secondary(config-real server)# port 0 tcp
ACOS-Inside-Secondary(config-real server-node port)# health-check-
disable
```

```
ACOS-Inside-Secondary(config-real server-node port)# exit
ACOS-Inside-Secondary(config-real server)# port 0 udp
ACOS-Inside-Secondary(config-real server-node port)# health-check-
disable
ACOS-Inside-Secondary(config-real server-node port)# exit
ACOS-Inside-Secondary(config-real server)# port 8080 tcp
ACOS-Inside-Secondary(config-real server-node port)# health-check-
disable
ACOS-Inside-Secondary(config-real server-node port)# exit
ACOS-Inside-Secondary(config-real server)# exit

ACOS-Inside-Secondary(config)# slb service-group LB_Paths_UDP udp
ACOS-Inside-Secondary(config-slb svc group)# member PSG1_Path 0
ACOS-Inside-Secondary(config-slb svc group)# member PSG2_Path 0
ACOS-Inside-Secondary(config-slb svc group)# exit
ACOS-Inside-Secondary(config)# slb service-group LB_Paths_TCP tcp
ACOS-Inside-Secondary(config-slb svc group)# member PSG1_Path 0
ACOS-Inside-Secondary(config-slb svc group)# member PSG2_Path:0
ACOS-Inside-Secondary(config-slb svc group)# exit
ACOS-Inside-Secondary(config)# slb service-group SSL tcp
ACOS-Inside-Secondary(config-slb svc group)# member PSG1_Path 8080
ACOS-Inside-Secondary(config-slb svc group)# member PSG2_Path 8080
ACOS-Inside-Secondary(config-slb svc group)# exit

ACOS-Inside-Secondary(config)# access-list 100 permit ip any any vlan 10
ACOS-Inside-Secondary(config)# slb virtual-server outbound_wildcard
0.0.0.0 acl 100
ACOS-Inside-Secondary(config-slb vserver)# port 0 tcp
ACOS-Inside-Secondary(config-slb vserver-vport)# name Inside1_in_to_out
ACOS-Inside-Secondary(config-slb vserver-vport)# service-group LB_Paths_
TCP
ACOS-Inside-Secondary(config-slb vserver-vport)# no-dest-nat
ACOS-Inside-Secondary(config-slb vserver-vport)# exit
ACOS-Inside-Secondary(config-slb vserver)# port 0 udp
ACOS-Inside-Secondary(config-slb vserver-vport)# name Inside1_in_to_out_
UDP
ACOS-Inside-Secondary(config-slb vserver-vport)# service-group LB_Paths_
UDP
ACOS-Inside-Secondary(config-slb vserver-vport)# no-dest-nat
```

```
ACOS-Inside-Secondary(config-slb vserver-vport)# exit
ACOS-Inside-Secondary(config-slb vserver)# port 443 https
ACOS-Inside-Secondary(config-slb vserver-vport)# name Inside1_in_to_out_
443
ACOS-Inside-Secondary(config-slb vserver-vport)# service-group SSL
ACOS-Inside-Secondary(config-slb vserver-vport)# template client-ssl
SSLInsight_ClientSide
ACOS-Inside-Secondary(config-slb vserver-vport)# no-dest-nat port-
translation
ACOS-Inside-Secondary(config-slb vserver-vport)# exit
```

VRRP-A Configuration

```
ACOS-Inside-Secondary(config)# vrrp-a common
ACOS-Inside-Secondary(config-common)# device-id 2
ACOS-Inside-Secondary(config-common)# set-id 1
ACOS-Inside-Secondary(config-common)# enable
ACOS-Inside-Secondary(config-common)# exit
ACOS-Inside-Secondary(config)# vrrp-a vrid 0
ACOS-Inside-Secondary(config-vrid:0)# floating-ip 10.1.1.1
ACOS-Inside-Secondary(config-vrid:0)# blade-parameters
ACOS-Inside-Secondary(config-vrid:0-blade-parameters)# priority 180
ACOS-Inside-Secondary(config-vrid:0-blade-parameters)# tracking-options
ACOS-Inside-Secondary(config-vrid:0-blade-parameters-track...)#
interface ethernet 1 priority-cost 60
ACOS-Inside-Secondary(config-vrid:0-blade-parameters-track...)#
interface ethernet 2 priority-cost 60
ACOS-Inside-Secondary(config-vrid:0-blade-parameters-track...)#
interface ethernet 20 priority-cost 60
ACOS-Inside-Secondary(config-vrid:0-blade-parameters-track...)# exit
ACOS-Inside-Secondary(config-vrid:0-blade-parameters)# exit
ACOS-Inside-Secondary(config-vrid:0)# exit
ACOS-Inside-Secondary(config)# vrrp-a vrid 15
ACOS-Inside-Secondary(config-vrid:15)# floating-ip 10.1.240.1
ACOS-Inside-Secondary(config-vrid:15)# blade-parameters
ACOS-Inside-Secondary(config-vrid:15-blade-parameters)# priority 180
ACOS-Inside-Secondary(config-vrid:15-blade-parameters)# tracking-options
ACOS-Inside-Secondary(config-vrid:15-blade-parameters-track...)#
interface ethernet 1 priority-cost 60
```

```
ACOS-Inside-Secondary(config-vrid:15-blade-parameters-track...)#  
interface ethernet 2 priority-cost 60  
ACOS-Inside-Secondary(config-vrid:15-blade-parameters-track...)#  
interface ethernet 20 priority-cost 60  
ACOS-Inside-Secondary(config-vrid:15-blade-parameters-track...)# exit  
ACOS-Inside-Secondary(config-vrid:15-blade-parameters)# exit  
ACOS-Inside-Secondary(config-vrid:15)# exit  
ACOS-Inside-Secondary(config)# vrrp-a vrid 16  
ACOS-Inside-Secondary(config-vrid:16)# floating-ip 10.1.250.1  
ACOS-Inside-Secondary(config-vrid:16)# blade-parameters  
ACOS-Inside-Secondary(config-vrid:16-blade-parameters)# priority 180  
ACOS-Inside-Secondary(config-vrid:16-blade-parameters)# tracking-options  
ACOS-Inside-Secondary(config-vrid:16-blade-parameters-track...)#  
interface ethernet 1 priority-cost 60  
ACOS-Inside-Secondary(config-vrid:16-blade-parameters-track...)#  
interface ethernet 2 priority-cost 60  
ACOS-Inside-Secondary(config-vrid:16-blade-parameters-track...)#  
interface ethernet 20 priority-cost 60  
ACOS-Inside-Secondary(config-vrid:16-blade-parameters-track...)# exit  
ACOS-Inside-Secondary(config-vrid:16-blade-parameters)# exit  
ACOS-Inside-Secondary(config-vrid:16)# exit  
ACOS-Inside-Secondary(config)# vrrp-a interface ethernet 18  
ACOS-Inside-Secondary(config-ethernet:18)# vlan 99
```

ACOS_encrypt CLI Configuration

Outside Primary ACOS device Configuration

The following commands access the configuration level of the CLI and change the hostname:

```
ACOS>enable  
Password:*****  
ACOS# config  
ACOS(config)# hostname ACOS-Outside-Primary
```

Layer 2/3 Configuration

The following commands configure the VLANs:

```
ACOS-Outside-Primary(config)# vlan 15
ACOS-Outside-Primary(config-vlan:15)# untagged ethernet 1
ACOS-Outside-Primary(config-vlan:15)# router-interface ve 15
ACOS-Inside-Secondary(config-vlan:15)# exit
ACOS-Outside-Primary(config)# vlan 16
ACOS-Outside-Primary(config-vlan:16)# untagged ethernet 2
ACOS-Outside-Primary(config-vlan:16)# router-interface ve 16
ACOS-Inside-Secondary(config-vlan:16)# exit
ACOS-Outside-Primary(config)# vlan 20
ACOS-Outside-Primary(config-vlan:20)# untagged ethernet 20
ACOS-Outside-Primary(config-vlan:20)# router-interface ve 20
ACOS-Inside-Secondary(config-vlan:20)# exit
ACOS-Outside-Primary(config)# vlan 99
ACOS-Outside-Primary(config-vlan:99)# untagged ethernet 18
ACOS-Outside-Primary(config-vlan:99)# router-interface ve 99
```

The following commands assign IP addresses to the VEs (router interfaces) that are configured on the VLANs.

```
ACOS-Outside-Primary(config-vlan:99)# interface ve 15
ACOS-Outside-Primary(config-if:ve15)# ip address 10.1.240.12
255.255.255.0
ACOS-Outside-Primary(config-if:ve15)# ip allow-promiscuous-vip
ACOS-Inside-Secondary(config-vlan:15)# exit
ACOS-Outside-Primary(config)# interface ve 16
ACOS-Outside-Primary(config-if:ve16)# ip address 10.1.250.12
255.255.255.0
ACOS-Outside-Primary(config-if:ve16)# ip allow-promiscuous-vip
ACOS-Inside-Secondary(config-vlan:16)# exit
ACOS-Outside-Primary(config)# interface ve 20
ACOS-Outside-Primary(config-if:ve20)# ip address 20.1.1.2 255.255.255.0
ACOS-Inside-Secondary(config-vlan:20)# exit
ACOS-Outside-Primary(config)# interface ve 99
ACOS-Outside-Primary(config-if:ve99)# ip address 99.1.1.1 255.255.255.0
ACOS-Outside-Primary(config-if:ve99)# exit
```


Promiscuous VIP mode is enabled on the VEs that are in the VLANs that contain the security devices. The other VEs do not use promiscuous VIP mode in this deployment.

The following commands configure static routes to the network on the client side of the inside ACOS devices. The next-hop IP address of each route is the floating IP address of a VRID on the inside ACOS devices. Specifically, these are the floating IP addresses that belong to the VRIDs for the VLANs that contain the security devices.

```
ACOS-Outside-Primary(config)# ip route 10.1.1.0 /24 10.1.240.1
ACOS-Outside-Primary(config)# ip route 10.1.1.0 /24 10.1.250.1
```

SSL Configuration

The following commands configure the server-SSL template:

```
ACOS-Outside-Primary(config)# slb template server-ssl SSLInsight_
ServerSide
ACOS-Outside-Primary(config-server SSL template)# forward-proxy-enable
ACOS-Outside-Primary(config-server SSL template)# exit
```

Path Configuration

The following commands configure the paths through the security devices to the router on the client network:

```
ACOS-Outside-Primary(config)# slb server server-gateway 20.1.1.253
ACOS-Outside-Primary(config-real server)# port 0 tcp
ACOS-Outside-Primary(config-real server-node port)# health-check-disable
ACOS-Outside-Primary(config-real server-node port)# exit
ACOS-Outside-Primary(config-real server)# port 0 udp
ACOS-Outside-Primary(config-real server-node port)# health-check-disable
ACOS-Outside-Primary(config-real server-node port)# exit
ACOS-Outside-Primary(config-real server)# port 443 tcp
ACOS-Outside-Primary(config-real server-node port)# health-check-disable
ACOS-Outside-Primary(config-real server-node port)# exit
ACOS-Outside-Primary(config-real server)# exit

ACOS-Outside-Primary(config)# slb service-group SG_TCP tcp
ACOS-Outside-Primary(config-slb svc group)# member server-gateway 0
ACOS-Outside-Primary(config-slb svc group)# exit
ACOS-Outside-Primary(config)# slb service-group SG_UDP udp
```

```
ACOS-Outside-Primary(config-slb svc group)# member server-gateway 0
ACOS-Outside-Primary(config-slb svc group)# exit
ACOS-Outside-Primary(config)# slb service-group SG_443 tcp
ACOS-Outside-Primary(config-slb svc group)# member server-gateway 443
ACOS-Outside-Primary(config-slb svc group)# exit
```

The following commands configure the wildcard VIP to intercept all outbound traffic that originates from the inside network:

```
ACOS-Outside-Primary(config)# access-list 100 permit ip any any vlan 15
ACOS-Outside-Primary(config)# access-list 100 permit ip any any vlan 16
ACOS-Outside-Primary(config)# slb virtual-server outside_in_to_out
0.0.0.0 acl 100
ACOS-Outside-Primary(config-slb vserver)# port 0 tcp
ACOS-Outside-Primary(config-slb vserver-vport)# service-group SG_TCP
ACOS-Outside-Primary(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS-Outside-Primary(config-slb vserver-vport)# no-dest-nat
ACOS-Outside-Primary(config-slb vserver-vport)# exit
ACOS-Outside-Primary(config-slb vserver)# port 0 udp
ACOS-Outside-Primary(config-slb vserver-vport)# service-group SG_UDP
ACOS-Outside-Primary(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS-Outside-Primary(config-slb vserver-vport)# no-dest-nat
ACOS-Outside-Primary(config-slb vserver-vport)# exit
ACOS-Outside-Primary(config-slb vserver)# port 8080 http
ACOS-Outside-Primary(config-slb vserver-vport)# name ReverseProxy_
Wildcard
ACOS-Outside-Primary(config-slb vserver-vport)# service-group SG_443
ACOS-Outside-Primary(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS-Outside-Primary(config-slb vserver-vport)# template server-ssl
outside-intercept
ACOS-Outside-Primary(config-slb vserver-vport)# exit
ACOS-Outside-Primary(config-slb vserver)# exit
```

VRRP-A Configuration

The following commands specify the VRRP-A device ID for this ACOS device, add the ACOS device to VRRP-A set 2, and enable VRRP-A on the device:

```
ACOS-Outside-Primary(config)# vrrp-a common
ACOS-Outside-Primary(config-common)# device-id 3
ACOS-Outside-Primary(config-common)# set-id 2
```

```
ACOS-Outside-Primary(config-common)# enable  
ACOS-Outside-Primary(config-common)# exit  
ACOS-Outside-Primary(config)#
```

The following commands configure the VRID for the interface with the client network:

```
ACOS-Outside-Primary(config)# vrrp-a vrid 0  
ACOS-Outside-Primary(config-vrid:0)# floating-ip 20.1.1.1  
ACOS-Outside-Primary(config-vrid:0)# blade-parameters  
ACOS-Outside-Primary(config-vrid:0-blade-parameters)# priority 200  
ACOS-Outside-Primary(config-vrid:0-blade-parameters)# tracking-options  
ACOS-Outside-Primary(config-vrid:0-blade-parameters-track...)# interface  
ethernet 1 priority-cost 60  
ACOS-Outside-Primary(config-vrid:0-blade-parameters-track...)# interface  
ethernet 2 priority-cost 60  
ACOS-Outside-Primary(config-vrid:0-blade-parameters-track...)# interface  
ethernet 20 priority-cost 60  
ACOS-Outside-Primary(config-vrid:0-blade-parameters-track...)#exit  
ACOS-Outside-Primary(config-vrid:0-blade-parameters)# exit  
ACOS-Outside-Primary(config-vrid:0)# exit
```

The following commands configure the VRID for the VLAN that contains the first security device (PSG1):

```
ACOS-Outside-Primary(config)# vrrp-a vrid 5  
ACOS-Outside-Primary(config-vrid:5)# floating-ip 10.1.240.11  
ACOS-Outside-Primary(config-vrid:5)# blade-parameters  
ACOS-Outside-Primary(config-vrid:5-blade-parameters)# priority 200  
ACOS-Outside-Primary(config-vrid:5-blade-parameters)# tracking-options  
ACOS-Outside-Primary(config-vrid:5-blade-parameters-track...)# interface  
ethernet 1 priority-cost 60  
ACOS-Outside-Primary(config-vrid:5-blade-parameters-track...)# interface  
ethernet 2 priority-cost 60  
ACOS-Outside-Primary(config-vrid:5-blade-parameters-track...)# interface  
ethernet 20 priority-cost 60  
ACOS-Outside-Primary(config-vrid:5-blade-parameters-track...)#exit  
ACOS-Outside-Primary(config-vrid:5-blade-parameters)# exit  
ACOS-Outside-Primary(config-vrid:5)# exit
```

The following commands configure the VRID for the VLAN that contains the second security device (PSG2):

```
ACOS-Outside-Primary(config)# vrrp-a vrid 6
ACOS-Outside-Primary(config-vrid:6)# floating-ip 10.1.250.11
ACOS-Outside-Primary(config-vrid:6)# blade-parameters
ACOS-Outside-Primary(config-vrid:6-blade-parameters)# priority 200
ACOS-Outside-Primary(config-vrid:6-blade-parameters)# tracking-options
ACOS-Outside-Primary(config-vrid:6-blade-parameters-track...)# interface
ethernet 1 priority-cost 60
ACOS-Outside-Primary(config-vrid:6-blade-parameters-track...)# interface
ethernet 2 priority-cost 60
ACOS-Outside-Primary(config-vrid:6-blade-parameters-track...)# interface
ethernet 20 priority-cost 60
ACOS-Outside-Primary(config-vrid:6-blade-parameters-track...)# exit
ACOS-Outside-Primary(config-vrid:6-blade-parameters)# exit
ACOS-Outside-Primary(config-vrid:6)# exit
```

The following command configures the VRRP-A interface that connects this ACOS device to its VRRP-A peer:

```
ACOS-Inside-Primary(config)# vrrp-a interface ethernet 18
ACOS-Inside-Primary(config-ethernet:18)# vlan 99
```

Outside Secondary ACOS device Configuration

The configuration on the outside secondary ACOS device is the same as the configuration on the outside primary ACOS device, with the exception of the following device-specific parameters:

- Hostname
- VRRP-A device ID
- Interface IP addresses
- Priority values of the VRIDs

Hostname Configuration

```
ACOS(config)# hostname ACOS-Outside-Secondary
```

Layer 2/3 Configuration

The following commands configure the VLANs:

```
ACOS-Outside-Secondary(config)# vlan 15
ACOS-Outside-Secondary(config-vlan:15)# untagged ethernet 1
ACOS-Outside-Secondary(config-vlan:15)# router-interface ve 15
ACOS-Outside-Secondary(config-vlan:15)# exit
ACOS-Outside-Secondary(config)# vlan 16
ACOS-Outside-Secondary(config-vlan:16)# untagged ethernet 2
ACOS-Outside-Secondary(config-vlan:16)# router-interface ve 16
ACOS-Outside-Secondary(config-vlan:16)# exit
ACOS-Outside-Secondary(config)# vlan 20
ACOS-Outside-Secondary(config-vlan:20)# untagged ethernet 20
ACOS-Outside-Secondary(config-vlan:20)# router-interface ve 20
ACOS-Outside-Secondary(config-vlan:20)# exit
ACOS-Outside-Secondary(config)# vlan 99
ACOS-Outside-Secondary(config-vlan:99)# untagged ethernet 18
ACOS-Outside-Secondary(config-vlan:99)# router-interface ve 99
ACOS-Outside-Secondary(config-vlan:99)# exit
ACOS-Outside-Secondary(config)# interface ve 15
ACOS-Outside-Secondary(config-if:ve15)# ip address 10.1.240.13
255.255.255.0
ACOS-Outside-Secondary(config-if:ve15)# ip allow-promiscuous-vip
ACOS-Outside-Secondary(config-vlan:15)# exit
ACOS-Outside-Secondary(config)# interface ve 16
ACOS-Outside-Secondary(config-if:ve16)# ip address 10.1.250.13
255.255.255.0
ACOS-Outside-Secondary(config-if:ve16)# ip allow-promiscuous-vip
ACOS-Outside-Secondary(config-vlan:16)# exit
ACOS-Outside-Secondary(config)# interface ve 20
ACOS-Outside-Secondary(config-if:ve20)# ip address 20.1.1.3
255.255.255.0
ACOS-Outside-Secondary(config-vlan:20)# exit
ACOS-Outside-Secondary(config)# interface ve 99
ACOS-Outside-Secondary(config-if:ve99)# ip address 99.1.1.2
255.255.255.0
ACOS-Outside-Secondary(config-if:ve99)# exit
ACOS-Outside-Secondary(config)# ip route 10.1.1.0 /24 10.1.240.1
ACOS-Outside-Secondary(config)# ip route 10.1.1.0 /24 10.1.250.1
```

SSLi Configuration

```
ACOS-Outside-Secondary(config)# slb template server-ssl SSLInsight_
ServerSide
ACOS-Outside-Secondary(config-server SSL template)# forward-proxy-enable
ACOS-Outside-Secondary(config-server SSL template)# exit
```

Path Configuration

```
ACOS-Outside-Secondary(config-client ssl)# slb server server-gateway
20.1.1.253
ACOS-Outside-Secondary(config-real server)# port 0 tcp
ACOS-Outside-Secondary(config-real server-node port)# health-check-
disable
ACOS-Outside-Secondary(config-real server-node port)# exit
ACOS-Outside-Secondary(config-real server)# port 0 udp
ACOS-Outside-Secondary(config-real server-node port)# health-check-
disable
ACOS-Outside-Secondary(config-real server-node port)# exit
ACOS-Outside-Secondary(config-real server)# port 443 tcp
ACOS-Outside-Secondary(config-real server-node port)# health-check-
disable
ACOS-Outside-Secondary(config-real server-node port)# exit
ACOS-Outside-Secondary(config-real server)# exit

ACOS-Outside-Secondary(config)# slb service-group SG_TCP tcp
ACOS-Outside-Secondary(config-slb svc group)# member server-gateway 0
ACOS-Outside-Secondary(config-slb svc group)# exit
ACOS-Outside-Secondary(config)# slb service-group SG_UDP UDP
ACOS-Outside-Secondary(config-slb svc group)# member server-gateway 0
ACOS-Outside-Secondary(config-slb svc group)# exit
ACOS-Outside-Secondary(config)# slb service-group SG_443 tcp
ACOS-Outside-Secondary(config-slb svc group)# member server-gateway 443
ACOS-Outside-Secondary(config-slb svc group)# exit

ACOS-Outside-Secondary(config)# access-list 100 permit ip any any vlan
15
ACOS-Outside-Secondary(config)# access-list 100 permit ip any any vlan
16
ACOS-Outside-Secondary(config)# slb virtual-server outside_in_to_out
0.0.0.0 acl 100
ACOS-Outside-Secondary(config-slb vserver)# port 0 tcp
```

```

ACOS-Outside-Secondary(config-slb vserver-vport)# service-group SG_TCP
ACOS-Outside-Secondary(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS-Outside-Secondary(config-slb vserver-vport)# no-dest-nat
ACOS-Outside-Secondary(config-slb vserver-vport)# exit
ACOS-Outside-Secondary(config-slb vserver)# port 0 udp
ACOS-Outside-Secondary(config-slb vserver-vport)# service-group SG_UDP
ACOS-Outside-Secondary(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS-Outside-Secondary(config-slb vserver-vport)# no-dest-nat
ACOS-Outside-Secondary(config-slb vserver-vport)# exit
ACOS-Outside-Secondary(config-slb vserver)# port 8080 http
ACOS-Outside-Secondary(config-slb vserver-vport)# name ReverseProxy_
Wildcard
ACOS-Outside-Secondary(config-slb vserver-vport)# service-group SG_443
ACOS-Outside-Secondary(config-slb vserver-vport)# use-rcv-hop-for-resp
ACOS-Outside-Secondary(config-slb vserver-vport)# template server-ssl
outside-intercept
ACOS-Outside-Secondary(config-slb vserver-vport)# exit
ACOS-Outside-Secondary(config-slb vserver)# exit

```

VRRP-A Configuration

```

ACOS-Outside-Secondary(config)# vrrp-a common
ACOS-Outside-Secondary(config-common)# device-id 4
ACOS-Outside-Secondary(config-common)# set-id 2
ACOS-Outside-Secondary(config-common)# enable
ACOS-Outside-Secondary(config-common)# exit
ACOS-Outside-Secondary(config)# vrrp-a vrid 0
ACOS-Outside-Secondary(config-vrid:0)# floating-ip 20.1.1.1
ACOS-Outside-Secondary(config-vrid:0)# blade-parameters
ACOS-Outside-Secondary(config-vrid:0-blade-parameters)# priority 180
ACOS-Outside-Secondary(config-vrid:0-blade-parameters)# tracking-options
ACOS-Outside-Secondary(config-vrid:0-blade-parameters-track...)#
interface ethernet 1 priority-cost 60
ACOS-Outside-Secondary(config-vrid:0-blade-parameters-track...)#
interface ethernet 2 priority-cost 60
ACOS-Outside-Secondary(config-vrid:0-blade-parameters-track...)#
interface ethernet 20 priority-cost 60
ACOS-Outside-Secondary(config-vrid:0-blade-parameters-track...)# exit
ACOS-Outside-Secondary(config-vrid:0-blade-parameters)# exit
ACOS-Outside-Secondary(config-vrid:0)# exit
ACOS-Outside-Secondary(config)# vrrp-a vrid 5

```

```
ACOS-Outside-Secondary(config-vrid:5)# floating-ip 10.1.240.11
ACOS-Outside-Secondary(config-vrid:5)# blade-parameters
ACOS-Outside-Secondary(config-vrid:5-blade-parameters)# priority 180
ACOS-Outside-Secondary(config-vrid:5-blade-parameters)# tracking-options
ACOS-Outside-Secondary(config-vrid:5-blade-parameters-track...)#
interface ethernet 1 priority-cost 60
ACOS-Outside-Secondary(config-vrid:5-blade-parameters-track...)#
interface ethernet 2 priority-cost 60
ACOS-Outside-Secondary(config-vrid:5-blade-parameters-track...)#
interface ethernet 20 priority-cost 60
ACOS-Outside-Secondary(config-vrid:5-blade-parameters-track...)# exit
ACOS-Outside-Secondary(config-vrid:5-blade-parameters)# exit
ACOS-Outside-Secondary(config-vrid:5)# exit
ACOS-Outside-Secondary(config)# vrrp-a vrid 6
ACOS-Outside-Secondary(config-vrid:6)# floating-ip 10.1.250.11
ACOS-Outside-Secondary(config-vrid:6)# blade-parameters
ACOS-Outside-Secondary(config-vrid:6-blade-parameters)# priority 180
ACOS-Outside-Secondary(config-vrid:6-blade-parameters)# tracking-options
ACOS-Outside-Secondary(config-vrid:6-blade-parameters-track...)#
interface ethernet 1 priority-cost 60
ACOS-Outside-Secondary(config-vrid:6-blade-parameters-track...)#
interface ethernet 2 priority-cost 60
ACOS-Outside-Secondary(config-vrid:6-blade-parameters-track...)#
interface ethernet 20 priority-cost 60
ACOS-Outside-Secondary(config-vrid:6-blade-parameters-track...)# exit
ACOS-Outside-Secondary(config-vrid:6-blade-parameters)# exit
ACOS-Outside-Secondary(config-vrid:6)# exit

ACOS-Inside-Primary(config)# vrrp-a interface ethernet 18
ACOS-Inside-Primary(config-ethernet:18)# vlan 99
```


SSLi in Multi-PU Deployment

This section describes the features and configurations specific to SSLi implementation in multi-PU platforms.

For the common multi-PU topology, configuration, and implementation details, see *Application Delivery Controller Guide*.

The following topics are covered:

- [Supported Features](#) 506
- [Key Considerations](#) 506
- [Limitations](#) 507
- [CLI Configuration](#) 507
- [Show Commands](#) 519

Supported Features

Supported SSLi Features on Multi-PU

- All the software SSL modules, TLS versions, and TLS offloading with HTTP/1.1 and HTTP/2.
- Certificate pinning, fetching, and forging.
- PKI certificates enrolment and renewal for SCEP and CMPv2.
- Web category bypass for URL classification lookup.
- TLS fingerprinting with JA3 for TLS handshake.
- Forward proxy and DNS dynamic services.

Supported Deployment and Topology

- All the SSLi basic deployments and topologies.
- Explicit and Transparent Proxy with Proxy chaining.
- SSLi in L3 mode for both static and dynamic ports.

Key Considerations

- The SSL session ID will be unique for each PU. For PU1 TLS connections, even session IDs are generated. Similarly, for PU2 TLS connections, odd session IDs are generated.
- For certificate management, the PKI, certificate, and key are generated as a part of enrollment or renewal process are configured on the PU1. This configuration is then replicated to the PU2 appropriately.
- For certificate forging, the certificates are auto generated and allows the SSLi to perform traffic inspection. The PU ID is added to the serial number of the forged certificates. This ensures the client's forged certificates are not conflicted and the PU1 and PU2 can independently perform traffic inspection. See [Certificate Fetching and Forging Configuration](#).

Limitations

- ADC and CGN cannot co-exist in the box.
- SSLi in L2 and virtual wire mode deployment are not supported.
- Enrollment and renewal of SCEP certificate with IPv6 address is not supported.
- Enrollment and renewal of ACME certificate is not supported.
- aXAPI of the SSL module will take more time to process and display the statistics of both the PUs as compared to the single PU.
- GUI and Harmony Controller support is not available in this release.

CLI Configuration

The SSLi multi-PU deployment can be with two L3v partitions or one shared partition and one L3v partition. This section describes the general CLI configuration guidelines and configuration examples for implementing SSLi in multi-PU platforms.

1. Configure the application type using the `chassis-application-type adc` command.
2. Enable the odd-even IP NAT globally using the `odd-even-nat-enable` command for seamless traffic distribution across both PUs.
3. Configure IP NAT pools using the `ip nat pool` or `ipv6 nat pool` commands to source NAT client traffic.
4. Configure client-side VLAN (or interface ethernet) with traffic distribution mode SIP using the `traffic-distribution-mode` command.
5. Configure server-side VLAN (or interface ethernet) with traffic distribution mode DIP using the `traffic-distribution-mode` command.
6. Create SLB client SSL and SLB server SSL templates.
7. Configure the forward proxy services or policies and bind them to the SLB client SSL template.

If using certificate forge with source NAT pool configuration, enable port splitting between PU1 and PU2 using the **system chassis-port-split** command. For more information, see [Certificate Fetching and Forging Configuration](#).

8. Create SLB servers, SLB service groups, SLB dynamic services, and SLB virtual servers and ports.
9. Bind the appropriate templates to the virtual ports.
10. Verify the configuration and view the show commands for traffic distribution.

The following topics are covered:

Basic SSLi Configuration	508
CLI Configuration	508
Explicit and Transparent Proxy Configuration	513
CLI Configuration	513
Certificate Fetching and Forging Configuration	518
CLI Configuration	518

Basic SSLi Configuration

This section provides the configuration example for SSLi in L3 mode deployment in multi-PU platforms.

For more information on all the SSLi deployments, see [Deployments](#).

CLI Configuration

Common Configuration

```
ACOS(config)# multi-ctrl-cpu 10
ACOS(config)# chassis-application-type adc
ACOS(config)# multi-config enable
ACOS(config)# system chassis-port-split enable
ACOS(config)# terminal idle-timeout 0
ACOS(config)# partition ssl_i_in id 15
ACOS(config)# partition ssl_i_out id 16
```

```
ACOS(config)# interface management
ACOS(config-if:management)# ip address 10.65.18.137 255.255.255.0
ACOS(config-if:management)# ip default-gateway 10.65.18.1
ACOS(config-if:management)# exit

ACOS(config)# ip route 10.10.10.0 /24 10.31.8.34
ACOS(config)# ip route 20.20.20.0 /24 10.31.9.34
ACOS(config)# ip route 172.20.0.0 /16 10.65.18.1
ACOS(config)# ipv6 route 2001:100::/64 2000:100:200::2
ACOS(config)# ipv6 route 2406:202:20::/64 2602:100:200::2
ACOS(config)# ipv6 route 2020:100::/64 2602:100:200::2
ACOS(config)# ip nat pool snat1 10.31.9.48 10.31.9.63 netmask /24
ACOS(config)# ip nat pool snat2 10.31.9.64 10.31.9.79 netmask /24
ACOS(config)# ipv6 nat pool snat6 2602:100:200::50 2602:100:200::80
netmask 64

ACOS(config)# slb common
ACOS(config-common)# odd-even-nat-enable
ACOS(config-common)# exit
```

ACOS Decrypt (Inside) Configuration

```
ACOS[ssli_in](config)# active-partition ssli_in
ACOS[ssli_in](config)# access-list 101 4 permit ip any any
ACOS[ssli_in](config)# vlan 100
ACOS[ssli_in](config-vlan:100)# tagged ethernet 1
ACOS[ssli_in](config-vlan:100)# router-interface ve 100
ACOS[ssli_in](config-vlan:100)# exit

ACOS[ssli_in](config)# vlan 200
ACOS[ssli_in](config-vlan:200)# tagged ethernet 1
ACOS[ssli_in](config-vlan:200)# router-interface ve 200
ACOS[ssli_in](config-vlan:200)# exit

ACOS[ssli_in](config)# interface ethernet 6
ACOS[ssli_in](config-if:ethernet:6)# enable
ACOS[ssli_in](config-if:ethernet:6)# traffic-distribution-mode dip
ACOS[ssli_in](config-if:ethernet:6)# ip address 192.168.10.1 255.255.255.0
ACOS[ssli_in](config-if:ethernet:6)# ip allow-promiscuous-vip
```

```
ACOS[ssli_in] (config-if:ethernet:6) # exit

ACOS[ssli_in] (config) # interface ve 100
ACOS[ssli_in] (config-if:ethernet:100) # ip address 192.168.100.1
255.255.255.0
ACOS[ssli_in] (config-if:ethernet:100) # ip allow-promiscuous-vip
ACOS[ssli_in] (config-if:ethernet:100) # exit

ACOS[ssli_in] (config) # interface ve 200
ACOS[ssli_in] (config-if:ethernet:200) # ip address 192.168.200.2
255.255.255.0
ACOS[ssli_in] (config-if:ethernet:200) # ip allow-promiscuous-vip
ACOS[ssli_in] (config-if:ethernet:200) # exit

ACOS[ssli_in] (config) # ip route 10.10.10.0 /24 192.168.20.2
ACOS[ssli_in] (config) # ip route 20.20.20.0 /24 192.168.10.2
ACOS[ssli_in] (config) # ip route 192.168.30.0 /24 192.168.10.2

ACOS[ssli_in] (config) # slb server outside-ax 192.168.10.2
ACOS[ssli_in] (config-real server) # port 8080 tcp
ACOS[ssli_in] (config-real server-node port) # health-check-disable
ACOS[ssli_in] (config-real server-node port) # exit

ACOS[ssli_in] (config) # slb service-group outside-ax-grp tcp
ACOS[ssli_in] (config-slb svc group) # member outside-ax 8080
ACOS[ssli_in] (config-slb svc group) # exit

ACOS[ssli_in] (config) # slb template client-ssl inside
ACOS[ssli_in] (config-client ssl) # forward-proxy-ca-certificate rsa-4k key
rsa-4k
ACOS[ssli_in] (config-client ssl) # forward-proxy-enable
ACOS[ssli_in] (config-client ssl) # exit

ACOS[ssli_in] (config) # slb virtual-server inside 0.0.0.0 acl 101
ACOS[ssli_in] (config-slb vserver) # port 443 https
ACOS[ssli_in] (config-slb vserver-vport) # source-nat auto
ACOS[ssli_in] (config-slb vserver-vport) # service-group outside-ax-grp
ACOS[ssli_in] (config-slb vserver-vport) # template client-ssl inside
ACOS[ssli_in] (config-slb vserver-vport) # no-dest-nat port-translation
ACOS[ssli_in] (config-slb vserver-vport) # exit
```

```
ACOS[ssli_in](config-slb vserver)# exit
```

ACOS Encrypt (Outside) Configuration

```
ACOS[ssli_out](config)# active-partition ssli_out
ACOS[ssli_out](config)# access-list 100 4 permit ip any any
ACOS[ssli_out](config)# vlan 101
ACOS[ssli_out](config-vlan:101)# untagged ethernet 10
ACOS[ssli_out](config-vlan:101)# router-interface ve 101
ACOS[ssli_out](config-vlan:101)# exit

ACOS[ssli_out](config)# vlan 301
ACOS[ssli_out](config-vlan:301)# tagged ethernet 5
ACOS[ssli_out](config-vlan:301)# router-interface ve 301
ACOS[ssli_out](config-vlan:301)# exit

ACOS[ssli_out](config)# interface ethernet 7
ACOS[ssli_out](config-if:ethernet:7)# enable
ACOS[ssli_out](config-if:ethernet:7)# traffic-distribution-mode sip
ACOS[ssli_out](config-if:ethernet:7)# ip address 192.168.10.2
255.255.255.0
ACOS[ssli_out](config-if:ethernet:7)# ip allow-promiscuous-vip
ACOS[ssli_out](config-if:ethernet:7)# exit

ACOS[ssli_out](config)# interface ve 101
ACOS[ssli_out](config-if:ethernet:101)# ip address 20.20.20.148
255.255.255.0
ACOS[ssli_out](config-if:ethernet:101)# ip allow-promiscuous-vip
ACOS[ssli_out](config-if:ethernet:101)# exit

ACOS[ssli_out](config)# interface ve 301
ACOS[ssli_out](config-if:ethernet:301)# ip address 30.30.30.148
255.255.255.0
ACOS[ssli_out](config-if:ethernet:301)# ip allow-promiscuous-vip
ACOS[ssli_out](config-if:ethernet:301)# exit

ACOS[ssli_out](config)# ip route 192.168.20.0 /24 192.168.10.1
ACOS[ssli_out](config)# ip route 192.168.100.0 /24 192.168.10.1
```

```
ACOS[ssli_out](config)# slb template server-ssl outside
ACOS[ssli_out](config)# slb server backend-srv 30.30.30.10
ACOS[ssli_out](config-real server)# port 0 tcp
ACOS[ssli_out](config-real server-node port)# health-check-disable
ACOS[ssli_out](config-real server-node port)# exit

ACOS[ssli_out](config-real server)# port 443 tcp
ACOS[ssli_out](config-real server)# port 8080 tcp
ACOS[ssli_out](config-real server-node port)# health-check-disable
ACOS[ssli_out](config-real server-node port)# exit
ACOS[ssli_out](config-real server)# exit

ACOS[ssli_out](config)# slb server s1 20.20.20.20
ACOS[ssli_out](config-real server)# port 443 tcp
ACOS[ssli_out](config-real server)# exit

ACOS[ssli_out](config)# slb service-group backend-srv-grp tcp
ACOS[ssli_out](config-slb svc group)# member backend-srv 443
ACOS[ssli_out](config-slb svc group)# exit

ACOS[ssli_out](config)# slb service-group sg tcp
ACOS[ssli_out](config-slb svc group)# member backend-srv 0
ACOS[ssli_out](config-slb svc group)# exit

ACOS[ssli_out](config)# slb virtual-server outside 0.0.0.0 acl 100
ACOS[ssli_out](config-slb vserver)# port 0 tcp
ACOS[ssli_out](config-slb vserver-vport)# source-nat auto
ACOS[ssli_out](config-slb vserver-vport)# service-group sg
ACOS[ssli_out](config-slb vserver-vport)# no-dest-nat
ACOS[ssli_out](config-slb vserver-vport)# exit

ACOS[ssli_out](config-slb vserver)# port 8080 http
ACOS[ssli_out](config-slb vserver-vport)# source-nat auto
ACOS[ssli_out](config-slb vserver-vport)# service-group backend-srv-grp
ACOS[ssli_out](config-slb vserver-vport)# template server-ssl outside
ACOS[ssli_out](config-slb vserver-vport)# no-dest-nat port-translation
ACOS[ssli_out](config-slb vserver-vport)# exit
ACOS[ssli_out](config-slb vserver)# exit
```


Explicit and Transparent Proxy Configuration

This section provides the configuration example for explicit and transparent proxy mode deployment in multi-PU platforms.

For more information on the explicit, transparent proxy, and proxy chaining deployments, see [Explicit and Transparent Proxy](#).

CLI Configuration

ACOS Decrypt (Inside) Configuration

```
ACOS_encrypt(config)# active-partition ssli_in
ACOS_decrypt(config)# chassis-application-type adc
ACOS_decrypt(config)# multi-config enable
ACOS_decrypt(config)# system chassis-port-split enable
ACOS_decrypt(config)# terminal idle-timeout 0
ACOS_decrypt(config)# ip dns primary 8.8.8.8

ACOS_decrypt(config)# vlan 30
ACOS_decrypt(config-vlan:30)# tagged ethernet 3
ACOS_decrypt(config-vlan:30)# router-interface ve 30
ACOS_decrypt(config-vlan:30)# name client
ACOS_decrypt(config-vlan:30)# traffic-distribution-mode sip
ACOS_decrypt(config-vlan:30)# exit

ACOS_decrypt(config)# interface management
ACOS_decrypt(config-if:management)# ip address 10.65.19.119 255.255.255.0
ACOS_decrypt(config-if:management)# ip default-gateway 10.65.19.1
ACOS_decrypt(config-if:management)# exit

ACOS_decrypt(config)# interface ethernet 1
ACOS_decrypt(config)# interface ethernet 2
ACOS_decrypt(config)# interface ethernet 3
ACOS_decrypt(config-if:ethernet:3)# speed-forced-40g
ACOS_decrypt(config-if:ethernet:3)# enable
ACOS_decrypt(config-if:ethernet:3)# ip allow-promiscuous-vip
ACOS_decrypt(config-if:ethernet:3)# ip nat inside
ACOS_decrypt(config-if:ethernet:3)# exit
```

```
ACOS_decrypt(config)# interface ethernet 4
ACOS_decrypt(config-if:ethernet:4)# speed-forced-40g
ACOS_decrypt(config-if:ethernet:4)# enable
ACOS_decrypt(config-if:ethernet:4)# exit

ACOS_decrypt(config)# interface ethernet 5
ACOS_decrypt(config)# interface ethernet 6
ACOS_decrypt(config-if:ethernet:6)# enable
ACOS_decrypt(config-if:ethernet:6)# traffic-distribution-mode dip
ACOS_decrypt(config-if:ethernet:6)# ip address 9.5.2.7 255.255.0.0
ACOS_decrypt(config-if:ethernet:6)# exit

ACOS_decrypt(config)# interface ethernet 7
ACOS_decrypt(config-if:ethernet:7)# enable
ACOS_decrypt(config-if:ethernet:7)# exit

ACOS_decrypt(config)# interface ve 30
ACOS_decrypt(config-if:ethernet:30)# ip address 30.30.30.1 255.255.255.0
ACOS_decrypt(config-if:ethernet:30)# ip allow-promiscuous-vip
ACOS_decrypt(config-if:ethernet:30)# ip route 0.0.0.0 /0 9.5.2.8
ACOS_decrypt(config-if:ethernet:30)# exit

ACOS_decrypt(config)# ip nat pool SSLI_NAT 9.5.2.123 9.5.2.124 netmask /24
gateway 9.5.2.8

ACOS_decrypt(config)# slb common
ACOS_decrypt(config-common)# odd-even-nat-enable
ACOS_decrypt(config-common)# exit

ACOS_decrypt(config)# slb server s1 9.5.2.8
ACOS_decrypt(config-real server)# health-check-disable
ACOS_decrypt(config-real server)# port 80 tcp
ACOS_decrypt(config-real server-node port)# health-check-disable
ACOS_decrypt(config-real server-node port)# exit
ACOS_decrypt(config-real server)# exit
```

```
ACOS_decrypt(config)# slb service-group sg-80 tcp
ACOS_decrypt(config-slb svc group)# health-check-disable
ACOS_decrypt(config-slb svc group)# member sl 80
ACOS_decrypt(config-slb svc group)# exit

ACOS_decrypt(config)# slb template client-ssl c1
ACOS_decrypt(config-client ssl)# forward-proxy-ca-certificate a.crt key
a.key
ACOS_decrypt(config-client ssl)# forward-proxy-enable
ACOS_decrypt(config-client ssl)# certificate rsa key rsa
ACOS_decrypt(config-client ssl)# exit

ACOS_decrypt(config)# slb template policy policy_ep_tp
ACOS_decrypt(config-policy)# forward-policy
ACOS_decrypt(config-policy-forward-policy)# action A1
ACOS_decrypt(config-policy-forward-policy-action)# forward-to-internet sg-
80 snat SSLI_NAT
ACOS_decrypt(config-policy-forward-policy-action)# log
ACOS_decrypt(config-policy-forward-policy-action)# exit
ACOS_decrypt(config-policy-forward-policy)# source ANY
ACOS_decrypt(config-policy-forward-policy-source)# match-any
ACOS_decrypt(config-policy-forward-policy-source)# destination any action
A1
ACOS_decrypt(config-policy-forward-policy-source)# exit
ACOS_decrypt(config-policy-forward-policy)# exit
ACOS_decrypt(config-policy)# exit

ACOS_decrypt(config)# slb template dynamic-service DNS
ACOS_decrypt(config-dynamic-service)# dns server 8.8.8.8
ACOS_decrypt(config-dynamic-service)# exit

ACOS_decrypt(config)# slb virtual-server vs_ep 30.30.30.123
ACOS_decrypt(config-slb vserver)# port 3128 http
ACOS_decrypt(config-slb vserver-vport)# template policy policy_ep_tp
ACOS_decrypt(config-slb vserver-vport)# source-nat auto
ACOS_decrypt(config-slb vserver-vport)# service-group sg-80
ACOS_decrypt(config-slb vserver-vport)# template dynamic-service DNS
ACOS_decrypt(config-slb vserver-vport)# template client-ssl c1
ACOS_decrypt(config-slb vserver-vport)# exit
```

```
ACOS_decrypt(config-slb vserver)# exit

ACOS_decrypt(config)# slb virtual-server vs_tp 0.0.0.0
ACOS_decrypt(config-slb vserver)# port 80 http
ACOS_decrypt(config-slb vserver-vport)# template policy policy_ep_tp
ACOS_decrypt(config-slb vserver-vport)# source-nat auto
ACOS_decrypt(config-slb vserver-vport)# service-group sg-80
ACOS_decrypt(config-slb vserver-vport)# template dynamic-service DNS
ACOS_decrypt(config-slb vserver-vport)# template client-ssl c1
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat port-translation
ACOS_decrypt(config-slb vserver-vport)# exit

ACOS_decrypt(config-slb vserver)# port 443 https
ACOS_decrypt(config-slb vserver-vport)# template policy policy_ep_tp
ACOS_decrypt(config-slb vserver-vport)# source-nat auto
ACOS_decrypt(config-slb vserver-vport)# service-group sg-80
ACOS_decrypt(config-slb vserver-vport)# template dynamic-service DNS
ACOS_decrypt(config-slb vserver-vport)# template client-ssl c1
ACOS_decrypt(config-slb vserver-vport)# no-dest-nat port-translation
ACOS_decrypt(config-slb vserver-vport)# exit
ACOS_decrypt(config-slb vserver)# exit
```

ACOS Encrypt (Outside) Configuration

```
ACOS_encrypt(config)# active-partition ssli_out
ACOS_encrypt(config)# ip dns primary 4.2.2.2
ACOS_encrypt(config)# ip dns secondary 8.8.8.8

ACOS_encrypt(config)# vlan 40
ACOS_encrypt(config-vlan:40)# tagged ethernet 4
ACOS_encrypt(config-vlan:40)# router-interface ve 40
ACOS_encrypt(config-vlan:40)# name server
ACOS_encrypt(config-vlan:40)# traffic-distribution-mode dip
ACOS_encrypt(config-vlan:40)# exit

ACOS_encrypt(config)# interface ethernet 7
```

```
ACOS_encrypt(config-if:ethernet:7)# traffic-distribution-mode sip
ACOS_encrypt(config-if:ethernet:7)# ip address 9.5.2.8 255.255.0.0
ACOS_encrypt(config-if:ethernet:7)# ip allow-promiscuous-vip
ACOS_encrypt(config-if:ethernet:7)# ip nat inside
ACOS_encrypt(config-if:ethernet:7)# exit

ACOS_encrypt(config)# interface ve 40
ACOS_encrypt(config-if:ethernet:40)# ip address 40.40.40.1 255.255.255.0
ACOS_encrypt(config-if:ethernet:40)# exit

ACOS_encrypt(config)# ip route 0.0.0.0 /0 40.40.40.10
ACOS_encrypt(config)# ip route 30.30.0.0 /16 9.5.2.7
ACOS_encrypt(config)# ip nat pool SSLI_NAT 40.40.40.123 40.40.40.124
netmask /24 gateway 40.40.40.10

ACOS_encrypt(config)# slb template server-ssl SSLI_OUT
ACOS_encrypt(config-server ssl)# forward-proxy-enable
ACOS_encrypt(config-server ssl)# exit

ACOS_encrypt(config)# slb server GW 40.40.40.10
ACOS_encrypt(config-real server)# health-check-disable
ACOS_encrypt(config-real server)# port 0 tcp
ACOS_encrypt(config-real server-node port)# health-check-disable
ACOS_encrypt(config-real server-node port)# exit

ACOS_encrypt(config-real server)# port 0 udp
ACOS_encrypt(config-real server-node port)# health-check-disable
ACOS_encrypt(config-real server-node port)# exit

ACOS_encrypt(config-real server)# port 443 tcp
ACOS_encrypt(config-real server-node port)# health-check-disable
ACOS_encrypt(config-real server-node port)# exit

ACOS_encrypt(config)# slb service-group GW_443 tcp
ACOS_decrypt(config-slb svc group)# health-check-disable
ACOS_decrypt(config-slb svc group)# member GW 443
ACOS_decrypt(config-slb svc group)# exit
```

```
ACOS_encrypt(config)# slb virtual-server SSLI_OUT 0.0.0.0
ACOS_encrypt(config-slb vserver)# port 80 http
ACOS_encrypt(config-slb vserver-vport)# source-nat pool SSLI_NAT
ACOS_encrypt(config-slb vserver-vport)# service-group GW_443
ACOS_encrypt(config-slb vserver-vport)# template server-ssl SSLI_OUT
ACOS_encrypt(config-slb vserver-vport)# no-dest-nat port-translation
ACOS_encrypt(config-slb vserver-vport)# exit
ACOS_encrypt(config-slb vserver)# exit
```

Certificate Fetching and Forging Configuration

This section provides the configuration example for certificate fetching and forging in multi-PU platforms.

In multi-PU platforms, the certificate forging connections are distributed between PU1 and PU2. When a client sends the HTTPS request to the server using odd and even client IPs, the odd client IP is sent to PU1, and the even client IP is sent to PU2. PU1 and PU2 maintain separate certificate cache entries for the domain with different serial numbers.

The serial number of the forged certificate is created uniquely between PUs by adding a 4-bit PU ID on the least significant byte. For example, for PU1: 0x11, 0x21, etc. and for PU2: 0x12, 0x22, etc.

For more information on certificate management, see [Certificate and Keys Management](#).

CLI Configuration

1. Enable the odd-even IP NAT globally using the `odd-even-nat-enable` command.
2. Configure IP NAT pools using the `ip nat pool` or `ipv6 nat pool` commands.
3. Create the SLB client SSL template.
4. Bind the forward proxy source NAT pool to the client SSL template using the `forward-proxy-source-nat pool` command.

Make sure to configure at least two source IP addresses: odd IP address and even IP address.

```
ACOS(config)# slb common
ACOS(config-common)# odd-even-nat-enable
ACOS(config)# exit

ACOS(config)# ip nat pool FORWARD_PROXY_NAT 10.10.10.123 10.10.10.124
netmask /24

ACOS(config)# slb template client-ssl c1
ACOS(config-client ssl)# local-logging
ACOS(config-client ssl)# ssli-logging all
ACOS(config-client ssl)# forward-proxy-ca-certificate test key test
ACOS(config-client ssl)# forward-proxy-source-nat pool FORWARD_PROXY_
NAT
ACOS(config-client ssl)# handshake-logging-enable
ACOS(config-client ssl)# exit
```

5. Verify the serial number of the forged certificate by capturing the packet in PCAP format and use **tshark** command.

```
tshark -nVr <packetcapturefilename> | grep serialNumber: | cut -f2 -d:
| while read f; do printf "serialNumber: 0x%x\n" $f; done
```

Show Commands

This section describes the various show commands to view traffic statistics on SSLi multi-PU. All the SSL module statistics are either aggregated as a single device (PU1+PU2) or merged or separate for PU1 and PU2.

For more information, see 'Show Commands' section under 'Secure Socket Layer Insight' in the *Command Line Interface Reference*.

View the SSL Counter Details

In this example, all the counters show aggregated values (PU1 + PU2). However, the **Handshake Average time** shows separate counters for PU1 and PU2.

On Primary Processing Unit:

```
-----
ACOS(config)#show slb ssl-counters
```

```
Global Stats
-----
```

```
Client ssl stats
```

```
Cumulative sessions = 1578
```

ID	Name	Successes	Failures
0x03000035	TLS1_RSA_AES_256_SHA	1578	0

Key Exchange Methods	Successes	Failures
RSA		
1024 bits	1578	0
ECDHE		
DHE		

SSL/TLS Version	Successes	Failures
TLS1.0	1578	0

Session Cache	Count
New	1578
Hit	0
Miss	0
Expired	0
Current	0

Handshake Average time: PU1:2ms PU2:19ms

```
Handshake Failures = 0
```

```
Certificate Auth = 0
```

```
SNI Auto-Map Successes = 0
```

```
SNI Auto-Map Failures = 0
```

```
SNI Auto-Map Failures Connection Closed = 0
```

```
SNI Auto-Map Failures Max Active Connections = 0
```

```
SNI Auto-Map Failures Missing Cert/Key = 0
```

```
SNI Bypass due to Missing Cert/Key = 0
```



```

SNI Bypass due to Certificate Expired = 0
SNI Bypass due to Matched Explicit Bypass List = 0

Renegotiation Counters
Total renegotiations = 0

Renegotiated SSL/TLS Versions                Successes      Failures
(none used)

Global Stats
-----
Server ssl stats
Cumulative sessions = 5

ID          Name                               Successes      Failures
0x0300c02b  TLS1_ECDHE_ECDSA_AES_128_GCM_SHA256  1              0
0x0300cca9  TLS1_ECDHE_ECDSA_CHACHA20_POLY1305_SHA256  4              0

Key Exchange Methods                Successes      Failures
RSA
ECDHE
    prime256v1                        5              0
DHE

SSL/TLS Version                Successes      Failures
TLS1.2                          5              0

Session Cache                Count
New                             5
Hit                             0
Miss                           0
Expired                        0
Current                        0

Handshake Average time: PU1:41ms PU2:37ms
Handshake Failures = 0

```

```
Certificate Auth = 0
```

```
Renegotiation Counters
```

```
Total renegotiations = 0
```

```
Renegotiated SSL/TLS Versions
```

```
Successes
```

```
Failures
```

```
(none used)
```

View the SSL Forward Proxy Certificate Details

In this example, the certificate details of both the PUs are merged.

```
ACOS(config)# show slb ssl-forward-proxy-cert vs_ep 3128 all
Virtual server (vs_ep: 3128):
----Start One Certificate---
real server: 200.0.60.96:443 tcp
server name: ax16-ssli-2.org
ALPN protocol: ALPN HTTP
state: ready
hit times: 1
idle time: 443 seconds
timeout after 3157 seconds
expires after 604347 seconds
----End One Certificate---

----Start One Certificate---
real server: 200.0.60.96:443 tcp
server name: ax16-ssli-11.org
ALPN protocol: ALPN HTTP
state: ready
hit times: 1
idle time: 1411 seconds
timeout after 2189 seconds
expires after 603379 seconds
version: 3
subject: SSL Certificate
common name: ssl-server.com
division: A10
locality: BJ
state or province: BJ
country name: CN
subject alternative name:
email:
start time: Nov 8 11:46:37 2018 GMT
expire time: Nov 5 11:46:41 2028 GMT
issuer: A10 Networks
serial(hex): 896f000000000534cc90
----End One Certificate---
```

View the SSL Statistics

In this example, all the counters show aggregated values (PU1+PU2).

```
ACOS(config)# show slb ssl stats
```

```
SSL module: QAT
Config module: Hardware
Number of SSL modules: 18
PU1 SSL module 1
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 28
PU1 SSL module 2
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 23
PU1 SSL module 3
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 23
PU1 SSL module 4
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 27
PU1 SSL module 5
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 18
PU1 SSL module 6
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 25
PU1 SSL module 7
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 0
PU1 SSL module 8
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 0
PU1 SSL module 9
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
```

```
    Number of requests handled: 0
PU2 SSL module 1
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 13
PU2 SSL module 2
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 16
PU2 SSL module 3
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 15
PU2 SSL module 4
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 14
PU2 SSL module 5
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 15
PU2 SSL module 6
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 17
PU2 SSL module 7
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 16
PU2 SSL module 8
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 11
PU2 SSL module 9
    Number of enabled crypto engines: 48
    Number of available crypto engines: 64
    Number of requests handled: 11
    Current clientside SSL connections: 0
```

```
Total clientside SSL connections: 19
Current serverside SSL connections: 0
Total serverside SSL connections: 34
Total Non SSL Bypass connections: 0
Total times of stateful session reuse in client ssl: 0
Total times of stateful session reuse in server ssl: 0
Total times of stateless session reuse in client ssl: 0
Total times of stateless session reuse in server ssl: 0
Total clientside early data connections: 0
Total serverside early data connections: 0
Total clientside failed early data connections: 0
Total serverside failed early data connections: 0
Failed SSL handshakes: 0
Failed crypto operations: 0
SSL memory usage: 778804 bytes
SSL server certificate errors: 0
SSL client certificate authorization failed: 0
SSL fail CA verification 0
HW Context Memory alloc failed 0
HW ring full 0
Record too big 0
Total client ssl context malloc failures: 0
Maximum SSL contexts N/A
Current SSL contexts in use 4
Static SSL contexts in use 4
Dynamic SSL contexts in use 0
SSL Forward Proxy
Bypass Failsafe SSL sessions: 0
Bypass Username sessions: 0
Bypass AD-Group sessions: 0
Bypass SNI sessions: 0
Bypass Certificate subject sessions: 0
Bypass Certificate issuer sessions: 0
Bypass Certificate SAN sessions: 0
Bypass NO SNI sessions: 0
Reset NO SNI sessions: 0
Bypass ESNI sessions: 0
Drop ESNI sessions: 0
```

```

Bypass Client Auth sessions: 0
Failed in SSL handshakes: 0
Failed in crypto operations: 0
Failed in TCP: 0
Failed in Certificate verification: 0
Failed in Certificate signing: 0
Invalid OCSP Stapling Response: 0
Revoked OCSP Response: 0
Unsupported SSL version: 0
Bypass triggered by aFlex: 0
SSLi Errors
Cert fetch, Fatal alert: 0
Cert fetch, TCP FIN/RST: 0
Cert fetch, validation error: 0
Client SSL, Fatal Alert: 0
Client SSL, TCP FIN/RST, Pinning: 0
Client SSL, Internal error: 0
Client SSL, Unknown error: 0
SSL Session, TCP FIN/RST: 0
Server SSL, Fatal alert: 0
Server SSL, TCP FIN/RST: 0
Server SSL, Internal error: 0
Server SSL, Unknown error: 0

```

View the Certificate Candidate Local List Details

In this example, the details of both PU1 and PU2 are merged under `pu` column.


```
ACOS#show slb ssl-cert-pinning-candidate-list local-list
Local Generated Cert-Pinning Candidate List: (Total entries: 50)
SNI                               PU      Conn failures  Bypass count
TTL(min)
www.gmail.com                     1       70             20
1200
www.yahoo.com                     1       56             37
1350
www.google.com                    2       22             10
860
youtube.com                       1       10             5
1440
gmail.com                         1       7              50
1220
www.yahoo.com                     2       5              10
1400
```

View the SSL OSCP Details

In this example, the details from PU2 are displayed.

```
ACOS(config)# show slb ssl-ocsp cache detail
Total: 841
-----
----
Name:      kohls.com
Status:    Good
Subject:    /C=US/ST=Illinois/L=Aurora/O=Kohl's, Inc./CN=kohls.com
Length:    2219
URI:        http://ocsp.digicert.com/
Expire:     575658
Hits:       0
PU:      2
```

Miscellaneous Features

The following topics are covered:

File Inspection	531
SSLi Source NAT	534
Self-Signed Certificates	542
Persistent Proxied Certificates	543
SSL CPU Offload	545
Chrome Browser Configuration Options	553

File Inspection

File inspection is an ACOS feature that uses an internal Cylance file inspection engine to examine files in HTTP data streams. The Cylance engine is implemented through an internal ICAP server and detects malware on the basis of millions of file signatures. The internal assigns a score to inspected files that ACOS uses as file management criteria. Files can be passed to their final destination, dropped, or referred to an external ICAP server for further inspection. The external server can be any ICAP based AMP. The feature supports inspecting client side download files.

Refer to the *Command Line Interface Reference* for more information about commands used in this section.

CLI Configuration

The following topics are covered:

Verifying the Device has a Cylance License	531
Creating a File Inspection Template	531
Binding the File Inspection Template to a Port	533
Importing a Cylance BW List	533
Implementing File Inspection on ADP	534

Verifying the Device has a Cylance License

File inspection requires an enabled Cylance license. The `show license-info` displays licenses enabled on a device. The following example indicates that an enabled Cylance license is installed on the device.

```
ACOS# show license-info | sec CYLANCE
CYLANCE                21-August-2018
ACOS#
```

Creating a File Inspection Template

File inspection templates are assigned to HTTP virtual ports to specify the device action upon files that are inspected. The *file-inspection template* command creates a

template and places the device in file-inspection template mode for modifying template parameters. When the command specifies an extant template, the subsequent commands edit that template.

Commands that configure the template include:

- *inspect downloads* – enables file inspection for ports upon which the template is bound; also specifies the data streams that are inspected and the ICAP server that inspects the files.
- *downloads bad* – specifies the action for files that are evaluated as “bad”. Valid actions include allowing the file to pass, dropping the file (default), or resetting the TCP connection.
- *downloads good* – specifies the action for files that are evaluated as “good”. Available actions include allowing the file to pass (default), dropping the file, or resetting the TCP connection.
- *downloads suspect* – specifies the action for files that are evaluated as “suspect”. Available actions include allowing the file to pass (default), dropping the file, resetting the TCP connection, or submitting the file to an external ICAP server for addition evaluation.

To use file inspection, the feature must be enabled globally and on each individual port where files are to be inspected. The *file-inspection service enable* command enables file inspection on the device.

This command enables file inspection globally on the device:

```
ACOS(config)# file-inspection service enable
```

This command disables file inspection globally. Virtual ports that are bound to a file-inspection template does not invoke Cylance inspection while the feature is globally disabled.

```
ACOS(config)# no file-inspection service enable
```

The *show process system* command indicates the status of the a10fi (file inspection) process. Use this command to verify the file-inspection process is running.

```
ACOS# show process system | sec a10fi
a10fi is not running
ACOS#
```

These CLI commands create a file inspection template and configures it to 1) allow good files to pass; 2) dropping bad files; and 3) sending suspect files to an external ICAP server; and 4) enables the port for inspecting downloaded client files.

```
ACOS(config)# file-inspection template FLOW_A
ACOS(config-file-inspection)# downloads bad drop log
ACOS(config-file-inspection)# downloads good reset no-log
ACOS(config-file-inspection)# downloads suspect external-inspect SERVER-1
log
ACOS(config-file-inspection)# inspect downloads
ACOS(config-file-inspection)# exit
```

These CLI commands implement an external ICAP server for inspecting files

```
ACOS(config)# slb template respmod-icap SERVER-1
ACOS(config-respmod-icap)# service-url icap://10.10.2.2/c-server
ACOS(config-respmod-icap)# exit
```

Binding the File Inspection Template to a Port

To implement file inspection, bind a file-inspection template to a HTTP virtual port. These command create a virtual port and configures that port to utilize the internal Cylance server for download files.

```
ACOS(config)# slb virtual-server VIP-1 10.1.1.1
ACOS(config-slb vserver)# port 80 HTTP
ACOS(config-slb vserver-vport)# template file-inspection FLOW_A
ACOS(config-slb vserver-vport)# exit
ACOS(config-slb vserver)# exit
```

Importing a Cylance BW List

Cylance maintains a global black and white list that contains files that were determined to be either good or bad by qualification means that are outside the machine learning algorithm. The *import file-inspection-bw-list* and *import-period file-inspection-bw-list* commands access this file from Cylance and installs it into the internal Cylance engine.

Refer to the *Command Line Interface Reference* for instructions on using the *import* and *import-periodic commands*.

Implementing File Inspection on ADP

In addition to the shared partition, file inspection is available in L3V and service partitions. The following guidelines are applicable to implementing file inspection on private partitions:

- File-inspection service is enabled using a global command in the shared partition.
- Viewing file-inspection statistics requires a file-inspection logging command in the shared partition.
- Since file-inspection implementation is only available to downloaded content, applying file-inspection templates to the SSLi re-encrypt partition is recommended
- A10 file-inspection only applies to downloaded content that follows the Http content-disposition header.

For information about Application Delivery Partitions, refer to the *Configuring Application Delivery Partitions* Guide .

SSLi Source NAT

In some applications of SSLi, it is important to choose the source IP address. For example, when SSLi is configured for transparent HTTP proxy chaining, SSLi source NAT allows the network administrator to specify source IP addresses on a client-initiated FETCH session. The source NAT addresses can be used by the chained upstream HTTP proxy server to differentiate the fetched traffic from all other traffic. It can then apply different policies to the fetched traffic from the policies it applies to all other traffic.

Static Source NAT CLI Configuration

This section provides detailed steps for configuring SSLi source NAT with statically specified IP addresses from a NAT address pool. For information on auto-SSLi source NAT, see the `forward-proxy-source-nat` command in the *Command Line Interface Reference*.

ACOS_decrypt CLI Configuration

The **blue highlighted** sections of this configuration show the commands required

to enable SSLi static source NAT.

- The `ip nat pool p199` and `ip nat pool p1` commands configure the IP address pools that provide the IP addresses referred to in the `forward-proxy-source-nat` and `source-nat pool` commands, respectively.
- The `forward-proxy-source-nat pool p199` command in the client-SSL configuration enables SSLi static-source NAT and specifies that NAT pool `p199` is used for client-initiated FETCH authentication sessions.
- The `source-nat pool p1` command under virtual port configurations, enable source NAT and specifies the NAT pool `p1` is used for normally authenticated SSL sessions.

```
ACOS-Inside# show running-config
!
access-list 100 permit ip any any vlan 10
!
vlan 10
    tagged ethernet 1
    router-interface ve 10
!
vlan 15
    tagged ethernet 1
    router-interface ve 15
!
hostname ACOS-Inside
!
interface ethernet 1
    enable
!
interface ve 10
    ip address 10.10.1.2 255.255.255.0
    ip allow-promiscuous-vip
!
interface ve 15
    ip address 10.15.1.2 255.255.255.0
!
ip nat pool p199 192.168.2.100 192.168.2.101 netmask /24
ip nat pool p1 192.168.2.102 192.168.2.103 netmask /24
```

```
!  
slb server FW1_Inspect 10.15.1.12  
    port 8080 tcp  
        health-check-disable  
    port 0 tcp  
        health-check-disable  
    port 0 udp  
        health-check-disable  
!  
slb service-group ALL_TCP_SG tcp  
    member FW1_Inspect 0  
!  
slb service-group ALL_UDP_SG udp  
    member FW1_Inspect 0  
!  
slb service-group FW1_Inspect_SG tcp  
    member FW1_Inspect 8080  
!  
slb template client-ssl SSLInsight_ClientSide  
    forward-proxy-ca-certificate Cert123.pem key key123  
    forward-proxy-enable  
    forward-proxy-source-nat pool p199  
!  
slb virtual-server Decrypt_VIP 0.0.0.0 acl 100  
    port 443 https  
        source-nat pool p1  
        service-group FW1_Inspect_SG  
        template client-ssl SSLInsight_ClientSide  
        no-dest-nat port-translation  
    port 0 tcp  
        source-nat pool p1  
        service-group ALL_TCP_SG  
        no-dest-nat  
    port 0 udp  
        source-nat pool p1  
        service-group ALL_UDP_SG  
        no-dest-nat  
    port 0 others  
        source-nat pool p1
```



```
service-group ALL_UDP_SG
no-dest-nat
!
end
```

ACOS_encrypt Configuration

No changes to the configuration of the outside ACOS device are needed to support SSLi source NAT.

Auto Source NAT CLI Configuration

This section provides detailed steps for configuring SSLi source NAT with automatically-acquired IP addresses matching the IP address of the ACOS interface facing the SSL Server.

The configuration example in this section is identical to [Static Source NAT CLI Configuration](#) except that SSLi auto source NAT is enabled on the virtual server.

Inside ACOS device Configuration

The **blue highlighted** sections of this configuration show the commands required to enable SSLi auto source NAT.

- The `ip nat pool p1` command configures the IP address pool that provides the IP addresses referred to in the `source-nat pool` command.
- The `forward-proxy-source-nat pool auto` command in the client-SSL configuration enables SSLi auto-NAT provides IP address(es) on the ACOS interface connected to the real server for client-initiated FETCH authentication sessions.
- The `source-nat pool p1` command under virtual port configurations, enable source NAT and specifies the NAT pool `p1` is used for normally authenticated SSL sessions.

```
ACOS-Inside# show running-config
!
access-list 100 permit ip any any vlan 10
!
vlan 10
```

```
    tagged ethernet 1
    router-interface ve 10
!
vlan 15
    tagged ethernet 1
    router-interface ve 15
!
hostname ACOS-Inside
!
interface ethernet 1
    enable
!
interface ve 10
    ip address 10.10.1.2 255.255.255.0
    ip allow-promiscuous-vip
!
interface ve 15
    ip address 10.15.1.2 255.255.255.0
!
ip nat pool p1 192.168.2.102 192.168.2.103 netmask /24
!
slb server FW1_Inspect 10.15.1.12
    port 8080 tcp
        health-check-disable
    port 0 tcp
        health-check-disable
    port 0 udp
        health-check-disable
!
slb service-group ALL_TCP_SG tcp
    member FW1_Inspect 0
!
slb service-group ALL_UDP_SG udp
    member FW1_Inspect 0
!
slb service-group FW1_Inspect_SG tcp
    member FW1_Inspect 8080
!
slb template client-ssl SSLInsight_ClientSide
```

```
forward-proxy-ca-certificate Cert123.pem key key123
forward-proxy-enable
forward-proxy-source-nat auto
!
slb virtual-server Decrypt_VIP 0.0.0.0 acl 100
  port 443 https
    source-nat pool p1
    service-group FW1_Inspect_SG
    template client-ssl SSLInsight_ClientSide
    no-dest-nat port-translation
  port 0 tcp
    source-nat pool p1
    service-group ALL_TCP_SG
    no-dest-nat
  port 0 udp
    source-nat pool p1
    service-group ALL_UDP_SG
    no-dest-nat
  port 0 others
    source-nat pool p1
    service-group ALL_UDP_SG
    no-dest-nat
!
end
```

Outside ACOS device Configurations

No changes to the configuration of the outside ACOS device are needed.

However, this sections shows the configuration of the real server, `Default_Gateway`, and its interface address, `20.1.1.10`, because this IP address is used by SSLi auto source NAT in fetched SSL sessions.

```
ACOS-Outside# show running-config
```

```
!
```

```
...
slb server Default_Gateway 20.1.1.10
  port 443 tcp
  health-check-disable
```

```
port 0 tcp
    health-check-disable
port 0 udp
    health-check-disable
...
```

Consolidated Configuration

For SSLi implementations, you can specify source NAT for the forwarding traffic either in the SLB policy template or in the client SSL template. If you have configured source NAT in both templates, the source NAT configuration in the SLB policy template has higher precedence than that of the client SSL template. This is the default behavior.

You can bypass this precedence so that ACOS uses the source NAT configuration defined in the client SSL template by using the **precedence** option in the **forward-proxy-source-nat** command.

The following is a sample configuration of the ACOS_decrypt. In the configuration example:

- The **ip nat pool** command configures the IP address pool for source NAT. In this example, p1, p2, and p3 are the three source NAT pools created.
- The **forward-proxy-source-nat pool** command in the client-SSL configuration provides IP address(es) on the ACOS interface connected to the real server for client-initiated FETCH authentication sessions.
- This example uses statically configured source NAT IP addresses. For dynamically configured IP addresses for source NAT, you can use the **forward-proxy-source-nat auto** command with the **precedence** option.

```
ip nat pool p1 10.105.1.88 10.105.1.88 netmask /24
!
ip nat pool p2 10.105.5.100 10.105.5.100 netmask /24
!
ip nat pool p3 10.105.5.101 10.105.5.101 netmask /24
!
ip nat alg pptp enable
!
```

```
ip route 10.105.2.0 /24 10.105.5.2
!
ip route 10.106.0.0 /16 10.105.5.2
!
slb template dynamic-service DNS
    dns server 10.105.1.140
!
slb server gw1 10.105.5.2
    health-check-disable
    port 0 tcp
        health-check-disable
    port 0 udp
        health-check-disable
    port 8080 tcp
        health-check-disable
!
slb service-group gw1_tcp_0 tcp
    member gw1 0
!
slb service-group gw1_tcp_8080 tcp
    member gw1 8080
!
slb service-group gw1_udp_0 udp
    member gw1 0
!
slb template client-ssl c-ssl2
    forward-proxy-ca-certificate Cert123.pem key key123
    forward-proxy-enable
    forward-proxy-source-nat pool p3 precedence
!The precedence option provides priority for the source NAT configured
here.
!
slb template policy EP1
    forward-policy
        action Permit_to_Internet
            forward-to-internet gw1_tcp_8080 snat p2
!
```

In absence of the precedence option, SNAT p2 is used to fetch the server certificate; otherwise, snat p3 configured in the client-ssl template is used to fetch the server certificate.

```

    log
    source any
    match-any
    destination any action Permit_to_Internet
!
slb virtual-server vs_ep 10.105.1.16
    port 8080 http
    service-group gw1_tcp_8080
    template policy EP1
    template dynamic-service DNS
    template client-ssl c-ssl2
    no-dest-nat port-translation

```

Self-Signed Certificates

A self-signed certificate is one in which the subject and issuer fields are the same. Because a self-signed certificate is a security risk, the ACOS device does not forward traffic to the self-signed certificate site.

CLI Configuration

To redirect clients from sites using self-signed certificates, enter the `forward-proxy-selfsign-redir` command in the configuration of the Client-SSL template. The ACOS device will redirect traffic away from the self-signed site and to a warning page in which the client sees, “The page you have tried to reach uses an untrusted certificate, please contact your administrator.”

Static-Port SSLi: Inside ACOS device SLB SSL Client Template

```

ACOS-inside(config)# slb template client-ssl SSLInsight_ClientSide
ACOS-inside(config-client ssl)# forward-proxy-ca-certificate Cert123.pem
key key123
ACOS-inside(config-client ssl)# forward-proxy-enable
ACOS-inside(config-client ssl)# forward-proxy-selfsign-redir

```

Dynamic-Port SSLi: Inside ACOS device SLB SSL Client Template

```

ACOS-inside(config)# slb template client-ssl Client-SSL
ACOS-inside(config-client ssl)# forward-proxy-ca-certificate Cert123.pem
key key123
ACOS-inside(config-client ssl)# forward-proxy-enable
ACOS-inside(config-client ssl)# forward-proxy-selfsign-redir
ACOS-inside(config-client ssl)# non-ssl-bypass service-group Outbound_
TCP

```

Show Configuration

Static-Port SSLi: Inside ACOS device SLB SSL Client Template

```

ACOS-Inside# show running-config slb template client-ssl
!Section configuration: 330 bytes
!
slb template client-ssl SSLInsight_ClientSide
    forward-proxy-ca-certificate Cert123.pem key key123
    forward-proxy-enable
    forward-proxy-selfsign-redir

```

Dynamic-Port SSLi: Inside ACOS device SLB SSL Client Template

```

ACOS-Inside# show running-config slb template client-ssl
!Section configuration: 330 bytes
!
slb template client-ssl Client-SSL
    forward-proxy-ca-certificate Cert123.pem key key123
    forward-proxy-enable
forward-proxy-selfsign-redir
    non-ssl-bypass service-group Outbound_TCP
!

```

Persistent Proxied Certificates

When an ACOS device restarts, or the forward-proxy process restarts, the cache of proxied certificates is emptied.

To save a group of proxied certificates that will be automatically re-installed after a restart, you need to configure a persistent forward-proxy class list and bind that class list to the client-SSL template.

Because the saved file of proxied certificates is periodically refreshed, it is possible that some proxied certificates will not persist if they were cached just before the system reset.

NOTE: Every unique SSLi virtual port needs a separate client-ssl template. This requirement only applies to virtual ports enabled for SSLi and does not apply to SSL offload or SSL proxy.

Creating a Persistent Forward-Proxy Class List

This example shows how to create the persistent forward-proxy class list and bind it to a client-SSL template:

1. To create or change persistent forward-proxy class list, use the `class-list` command with the `ac` option.

The `class-list` command creates a class list and gives it a name. The `file` option saves the list as a file that you can export. Without this option, the class list entries are saved in the configuration file instead. The `ac` option is required for the persistent certificates feature, and specifies that the list type is Aho-Corasick.

If an SNI in a certificate matches an entry in this class list, it is retained; otherwise, it is dropped.

```
ACOS-Inside# configure
ACOS-Inside(config)# class-list persist-servers-CL ac
ACOS-Inside(config-class list)# contains jsmith.com
ACOS-Inside(config-class list)# contains EnterpriseABC.com
ACOS-Inside(config-class list)# equals UofKgmc.edu/admissions
```

2. Bind the new or changed class list to the client-SSL template:

```
ACOS-Inside(config)# slb template client-ssl SSLInsight_ClientSide
ACOS-Inside(config-client ssl)# forward-proxy-cache-persistence class-
list persist-servers-CL
ACOS-Inside(config-client ssl)# forward-proxy enable
```



```
ACOS-Inside(config-client ssl)# forward-proxy-ca-certificate  
Cert123.pem key key123
```

3. Commit the changes to ACOS memory.

```
ACOS-Inside(config)# write memory
```

4. Use the **show class-list** command to display the **persist-servers-CL** class-list.

Binding a Separate Client-SSL Template

The following example illustrates the requirement of this feature that a separate client-SSL template must be bound to each unique SSLi virtual port (port 443 https):

```
ACOS-Inside(config)# slb virtual-server vip1 0.0.0.0 acl 1  
ACOS-Inside(config-slb vserver)# port 443 https  
ACOS-Inside(config-slb vserver-vport)# template client-ssl test1  
ACOS-Inside(config-slb vserver-vport)# exit  
ACOS-Inside(config-slb vserver)# exit  
ACOS-Inside(config)# slb virtual-server vip2 0.0.0.0 acl 2  
ACOS-Inside(config-slb vserver)# port 443 https  
ACOS-Inside(config-slb vserver-vport)# template client-ssl test2
```

SSL CPU Offload

ACOS supports CPU offload infrastructure and SSL software offload engine to offload asynchronous SSL crypto operations. This solution improves performance, handles many Connections per Second (CPS), reduces CPU load, and frees up more resources for other jobs.

This feature has the following implementation:

- CPU Offload Infrastructure
 - Support configuration on the system level
 - Support offload CPUs allocation
 - Support jobs submission to handle on offload CPUs and displays their statistics.

- **SSL Software Offload Engine:**
 - Support configuration on the SSL module level
 - Support SSL operations submission to offload Infrastructure
 - Support SSL connections suspension/resumption based on offload results.

Known Limitations

- The CPU Offload Infrastructure supports 8192 jobs per data CPU
- The SSL offload engine supports both client and server SSL and does not support SSLi.

CLI Configuration

For CPU offload infrastructure, use following commands:

- To enable the CPU offload infrastructure, use the following command:

```
ACOS(config)# offload-cpus <1-12>
```

- To view the CPU offload infrastructure statistics, use the following command:

```
ACOS(config)# show system job-offload [statistics]
```

- To view the CPU usage statistics, use the following command:

```
ACOS(config)# show cpu
```

- To view the number of offload CPUs, use the following command:

```
ACOS(config)# show run
```

For SSL offload engine, use the following commands:

- To enable the SSL software offload infrastructure, use the following command:

```
ACOS(config)# ssl-module software-tls13-offload
```

- To view the configured SSL offload engine, use the following command:

```
ACOS(config)# show slb ssl stats
```

For detailed CLI configuration, refer *Command Line Reference* and *Command Line Reference for ADC* guides.

GUI Configuration

To configure CPU offload infrastructure and view the statistics, navigate to **System > Settings > Offload CPUs**.

AutoUpdateO365 Script

ACOS supports the AutoUpdateO365 shell script to automatically check Office 365 IP address changes and update the configured class lists under SSLi policy configuration (forward-proxy-bypass). For more information, see [Bypass, Inspect, and Exception](#).

Known Limitation

If you are using Virtual Chassis System (VCS) and plan to upgrade or disable VCS for any reason, the script should be disabled. Because automatic class list updates can cause VCS instability due to VCS sequence number mismatch while the master and blade are disconnected.

To disable the script, see [Disabling the Script](#).

Enabling the script

1. Download the script from <https://glm.a10networks.com/secure/download/support-axseries/downloads/pkg-1591084417.tar.gz>.
2. Import the script as an external health monitor program.
3. Configure a real server to use the external health method.

```
slb server localhost 127.0.0.1
!
health monitor AutoUpdateO365
    retry 1
    interval 180 timeout 80
    method external program act_o365_auto_update_domains.sh
!
```

For more information, see 'Using External Health Methods' section in the *Application Delivery Controller Guide*.

Disabling the Script

1. Execute the `show running-config slb server` command.

```
slb server localhost 127.0.0.1
    health-check AutoUpdate0365
!
health monitor AutoUpdate0365
    retry 1
    interval 180 timeout 80
    method external program act_o365_auto_update_domains.sh
!
class-list o365-urls-ALLOW
class-list o365-urls-DEF
class-list o365-urls-OPT
class-list o365-ipv4-ALLOW
class-list o365-ipv4-DEF
class-list o365-ipv4-OPT
!
```

2. Remove the `health monitor AutoUpdate0365` line.

AutoUpdate0365 Script Example

```
#!/bin/bash
set -u
## set file permission to RW
umask 0111
## suppress curl memory consumption
export NSS_SDB_USE_CACHE=NO
## parameters to upload files via aXAPI
readonly HOST="localhost"
readonly PARTITION="shared"
readonly TARGET="class-list"
readonly AC_MATCH="ends-with"
readonly PROXY=""

## working directory
readonly WORKDIR="/a10data/guest/O365/${PARTITION}"

## Generating different GUID
guid(){
    uuidgen
}
```

```
## parameters to get Office 365 endpoints information
readonly
URL="https://endpoints.office.com/endpoints/worldwide?clientrequestid=$(
guid)"
readonly LIST="${WORKDIR}/urllist"

## parameters to find valid execution timing
readonly CURR_TIME=$(date +%H:%M)
readonly BEGIN_TIME="00:00"
readonly END_TIME="23:59"

## exec schedule
param_check()
{
    if [ "${AC_MATCH}" == "contains" ] || [ "${AC_MATCH}" == "ends-with"
]; then
        :
    else
        echo "AC_MATCH:${AC_MATCH} isn't valid. It should be 'contains' or
'ends-with'."
        exit 1
    fi
}

time_check()
{
    if [[ "${CURR_TIME}" < "${BEGIN_TIME}" ]] || [[ "${CURR_TIME}" >
"${END_TIME}" ]]; then
        echo "${CURR_TIME} => NO GO"
        exit 1
    else
        echo "${CURR_TIME} => GO"
    fi
}

## check file dir and move
```

```

dir_check() {
    if [ -d "${WORKDIR}" ]; then
        cd ${WORKDIR} || exit 1
    else
        mkdir -p -m 777 ${WORKDIR} || exit 1
        cd ${WORKDIR} || exit 1
    fi
}

## get source list and save
get_source() {
    if [ -z "$PROXY" ]; then
        curl -sS -k -m 5 --retry 1 -o ${LIST} "${URL}" || exit 1
    else
        curl -sS -k -m 5 --retry 1 -o ${LIST} "${URL}" -x ${PROXY} || exit 1
    fi
}

## create class-list files
create_list() {
    ## divide original list into ID based dictionary files
    if [ -w ${LIST} ]; then
        awk '/{/,/}/' ${LIST} | awk '/{/{x="dict."++i;}{print > x;}'
    else
        exit 1
    fi

    ## create null files to be updated
    for route_type in ALLOW DEF OPT; do
        : >dict.${route_type}
        for key_name in urls ipv4 ipv6; do
            : >o365-${key_name}-${route_type}
        done
    done

    ## list file names which "expressRoute" is True ==>> ALLOW(ress)
    awk '/"category": "Allow"/ {print FILENAME}' dict.* | while read dict_

```

```

name; do
    cat "${dict_name}" >>dict.ALLOW
done

## list file names which "expressRoute" is False ==>> DEF(ault)
awk '/"category": "Default"/ {print FILENAME}' dict.* | while read
dict_name; do
    cat "${dict_name}" >>dict.DEF
done

## list file names which "expressRoute" is False ==>> DEF(ault)
awk '/"category": "Optimize"/ {print FILENAME}' dict.* | while read
dict_name; do
    cat "${dict_name}" >>dict.OPT
done

## create list of urls, ipv4 and ipv6 for both ALLOW and DEF
for route_type in ALLOW DEF OPT; do
    awk '/urls/,/]/' dict.${route_type} | awk '! /\[\[\]]/' | sed -e 's/^
*\"//\" -e 's/\".*//\" -e 's/.*\"//\" | sort -u >urls.${route_type}
    awk '/\"ips\"/,/]/' dict.${route_type} | awk '! /\[\[\]]/ && /\./' |
sed -e "s/^ *\"//\" -e "s/\".*//\" | sort -u >ipv4.${route_type}
    awk '/\"ips\"/,/]/' dict.${route_type} | awk '! /\[\[\]]/ && /\:/' |
sed -e "s/^ *\"//\" -e "s/\".*//\" | sort -u >ipv6.${route_type}
done

## Prioritize Duplicate Entries in the order, Optimize Allowed
Default
for key_name in urls ipv4 ipv6; do
    sort ${key_name}.OPT | uniq > o365-${key_name}-OPT
    sort ${key_name}.ALLOW | uniq > SORTED-${key_name}.ALLOW
    sort ${key_name}.DEF | uniq > SORTED-${key_name}.DEF

    comm -23 SORTED-${key_name}.ALLOW o365-${key_name}-OPT > o365-${key_
name}-ALLOW
    comm -23 SORTED-${key_name}.DEF o365-${key_name}-ALLOW > TEMP-o365-
${key_name}-DEF

```

```
comm -23 TEMP-o365-${key_name}-DEF o365-${key_name}-OPT > o365-${key_
name}-DEF
done

## convert the lists to class-list format
for route_type in ALLOW DEF OPT; do
    sed -i -e "s/^/${AC_MATCH} /" o365-urls-${route_type}
    sed -i "1iclass-list o365-urls-${route_type} ac file" o365-urls-
${route_type}
    for key_name in ipv4 ipv6; do
        sed -i "1iclass-list o365-${key_name}-${route_type} ${key_name}
file" o365-${key_name}-${route_type}
    done
done

}

## import class-list files
import_list() {

    ## import class-lists
    for route_type in ALLOW DEF OPT; do
        for key_name in urls ipv4 ipv6; do
            /a10/bin/rimaccli -u -e -f "o365-${key_name}-${route_type}"
            sleep 5
        done
    done
}

## clean up created files
cleanup() {
    rm -f dict.*
    rm -f url*
    rm -f ipv*
    rm -f o365*
}
```



```
## exec
time_check && param_check && dir_check && cleanup && get_source &&
create_list && import_list
```

Chrome Browser Configuration Options

The Chrome browser is popular and so are Google services, such as search and Gmail. Many enterprise customers want to inspect this Google traffic. However, the Chrome browser can use the QUIC protocol instead of standard HTTPS. To inspect Google traffic, enforce the use of HTTPS by denying QUIC messages with an ACL to deny the destination port 443 udp traffic as shown in the configuration example below.

```
access-list 103 deny udp any any eq 80
access-list 103 deny udp any any eq 443
access-list 103 permit ip any any
!
interface ethernet 2
    enable
    access-list 103 in
    ip address 172.16.1.1 255.255.255.0
    ip allow-promiscuous-vip
```

Error Logging

This section describes details about error and SSLi logging functionalities.

The following topics are covered:

Overview	555
CLI Configuration	557
Failure Event Error Reasons	558
Generic Failure Logs	584
Event-based Logging	587
Logging and Exporting SSL Session Secrets	587

Overview

SSLi logging provides comprehensive monitoring and analysis for debugging. ACOS supports logging of all SSL handshake states in the system log. This includes capturing both client and server SSL logs for successful and failed events. These events are mapped to different severity levels.

The severity levels of the events are captured as:

- **Error** - An event is categorized as an error in case of a failure. By default, SSLi logging is enabled to capture only error level logs. The `ssli-logging disable` can be set to disable the default setting.
- **Information (Info)** - An event is categorized as info for intercept and bypass actions (forward-proxy-bypass). The `ssli-logging all` can be set to capture both error and information level logs.

NOTE:

Although the default SSLi logging captures only error level logs, there is one exception. In case the forward-proxy-bypass is configured, the 'info' level logs are also captured for 'SSL Inspection Bypassed.'

This additional logging occurs because 'inspection' is considered as standard setting, while 'bypass' (no inspection) is considered as exception handling setting. Therefore, you should treat such log as an exception information rather than an error log.

Three types of SSLi events are categorized as Error and Info:

Inspection successful event

This has two types of logs, one for the start of inspection and the other for the completion of inspection. The log for these events includes session statistics.

The following is an example log for an SSLi inspection start event, log level is info:

```
May 15 2018 21:27:19 Info      [SSL]: ssl99 CEF:0|A10|ADC|4.1.4-
P2|SSLi 486706518616440836|SSL Inspection Start|6|src=10.105.11.97
dst=10.105.22.94 spt=52214 dpt=443 act=inspected dhost=s94 cs1=vs-
wildcard-internal cs1Label=VIP name cs2=https cs2Label=VIP protocol
cs3=TLSv1.2 cs3Label=SSL version cs4=AES256-GCM-SHA384 cs4Label=Cipher
suite cs6=VALID cs6Label=Certificate Validity status cn1=443
cn1Label=VIP port cn2=16 cn2Label=Log ID
```

The following is an example log for an SSLi inspection successful event, log level is info:

```
May 15 2018 21:27:24 Info      [SSL]: ssl99 CEF:0|A10|ADC|4.1.4-
P2|SSLi 486706518616440833|SSL Inspection Successful|6|src=10.105.11.97
dst=10.105.22.94 spt=52214 dpt=443 act=inspected dhost=s94 cs1=vs-
wildcard-internal cs1Label=VIP name cs2=https cs2Label=VIP protocol
cs3=TLSv1.2 cs3Label=SSL version cs4=AES256-GCM-SHA384 cs4Label=Cipher
suite cs6=VALID cs6Label=Certificate Validity status cn1=443
cn1Label=VIP port cn2=8 cn2Label=FWD Packets cn3=1769 cn3Label=REV
Packets cn4=6 cn4Label=Duration seconds cn5=26 cn5Label=Log ID cn6=962
cn6Label=FWD Bytes Transferred cn7=11 cn7Label=REV Bytes Transferred
```

Inspection failure event

The log for this event does not include session statistics.

The following is an example log for an SSLi error event, log level is error:

```
May 15 2018 21:25:56 Error      [SSL]: ssl99 CEF:0|A10|ADC|4.1.4-
P2|SSLi 486706518616440835|SSL Inspection Failed|3|src=10.105.11.97
dst=10.105.22.94 spt=52210 dpt=443 act=dropped dhost=s94 cs1=vs-
wildcard-internal cs1Label=VIP name cs2=https cs2Label=VIP protocol
cs3=TLSv1.2 cs3Label=SSL version cs6=INVALID cs6Label=Certificate
Validity status cs7=CERT Fetch, Validation Error cs7Label=Error type
cn1=443 cn1Label=VIP port cn2=2 cn2Label=Log ID
```

Inspection bypass event

The log for this event does not include session statistics.

The following is an example log for an SSLi bypass event, log level is info:

```
May 15 2018 21:24:11 Info      [SSL]: ssli99 CEF:0|A10|ADC|4.1.4-
P2|SSLi 486706518616440834|SSL Inspection Bypassed|6|src=10.105.11.97
dst=172.217.164.110 spt=59644 dpt=443 act=bypassed dhost=google.com
cs1=vs-wildcard-internal cs1Label=VIP name cs2=https cs2Label=VIP
protocol cs3=TLSv1.2 cs3Label=SSL version cs5=Search Engines
cs5Label=Web Category cs6=UNKNOWN cs6Label=Certificate Validity status
cn1=443 cn1Label=VIP port cn2=6 cn2Label=Log ID
```

All the logs are generated in the CEF standard format. The log output displays information such as web category, certificate valid status, session duration, log ID, and so on, in session start, end, bypass, or error case.

A log message compliant with CEF follows a specific format. The information before “[Extension]” is mandatory and called a CEF header as shown in the following sample:

```
ACOS#show log
Jan 18 11:07:53 host CEF:Version|Device Vendor|Device Product|Device
Version|Device Event Class ID|Name|Severity|[Extension]
```

NOTE: ACOS does not support Syslog format for SSLi log type. Therefore, if you enable an `acos-events` template with Syslog format, the log server will not log any messages.

CLI Configuration

By default, SSLi logging is enabled for capturing SSL errors.

- To enable SSLi logging for all events, run the following commands:

```
ACOS(config)# slb template client-ssl clientssl
ACOS(config-client ssl)# ssli-logging all
```

- To disable SSLi logging, run the following commands:

```
ACOS(config)# slb template client-ssl clientssl
ACOS(config-client ssl)# ssli-logging disable
```

- To reset SSLi logging to default, run the following commands:

```
ACOS(config)# slb template client-ssl clientssl
ACOS(config-client ssl)# no ssli-logging disable
```

Failure Event Error Reasons

The following table lists the error reasons and their associated explanations.

Sr.No	Error Reason	Description
1	CERT Fetch, Receive SSL Fatal Alert	ACOS receives an SSL fatal alert packet while fetching the SSL certificate.
	The below section lists the error when the SSL handshake occurs between ACOS_decrypt and the server, and a TCP FIN/RST packets are received by ACOS_decrypt.	
2	Handshake Failure	A failure occurred during SSL handshake.
3	CERT Fetch, connection finish by peer OR Connection finish by peer	The server finishes the connection while fetching the SSL certificate.
4	CERT Fetch, DNS resolved error OR DNS resolved error	The DNS server could not resolve the domain name while fetching the SSL certificate.
5	CERT Fetch, DNS timeout OR DNS timeout	The DNS server connection timed out while fetching the SSL certificate.
6	CERT Fetch, DNS server failed OR DNS server failed	The DNS server failed while fetching the SSL certificate.
7	CERT Fetch, TCP timeout OR TCP timeout	The TCP connection or session timed out while fetching the SSL certificate.
8	CERT Fetch, UDP timeout OR UDP timeout	The UDP connection or session timed out while fetching the SSL certificate.

Sr.No	Error Reason	Description
9	CERT Fetch, connection reset by peer OR Connection reset by peer	The server resets the connection while fetching the SSL certificate.
10	CERT Fetch, delete connection failed OR Delete connection failed	The connection could not be deleted after failure to fetch the SSL certificate.
11	CERT Fetch, create connection failed OR Create connection failed	The connection could not be created or established before fetching the SSL certificate.
12	CERT Fetch, port not found OR Port not found	The server port was not found while fetching the SSL certificate.
13	CERT Fetch, IP not found OR IP not found	The server IP address was not found while fetching the SSL certificate.
14	CERT Fetch, cannot obtain NAT address OR Cannot obtain NAT address	The server NAT address was not obtained while fetching the SSL certificate.
15	CERT Fetch, invalid session OR Invalid session	The server session was invalid while fetching the SSL certificate.
16	CERT Fetch, TCP process data event error OR TCP process data event error	The data process failed while fetching the SSL certificate.
17	CERT Fetch, TCP process error OR TCP process error	The TCP process failed while fetching the SSL certificate.
	The below section lists the error occurs when the SSL handshake takes place between ACOS_decrypt and the server, and ACOS_decrypt is unable to validate the server certificate.	

Sr.No	Error Reason	Description
18	Unable to get issuer certificate	The issuer certificate of a looked-up certificate could not be found. This happens because the list of trusted certificates is incomplete.
19	Unable to get certificate CRL	The CRL of a certificate could not be found.
20	Unable to decrypt certificate's signature	The server could not decrypt the certificate's signature. This error occurs because the actual signature value could not be determined or not matching the expected value.
21	Unable to decrypt CRL's signature	The server could not decrypt the CRL's signature. This error occurs because the actual signature value could not be determined or not matching the expected value.
22	Unable to decode issuer public key	The public key in the certificate SubjectPublicKeyInfo could not be read.
23	Certificate signature failure	The signature of the certificate is invalid.
24	CRL signature failure	The signature of the CRL is invalid.
25	Certificate is not yet valid	The SSL handshake failed as the certificate is not the yet valid. This error occurs because the system clock is not Today's Date.
26	CRL is not yet valid	The SSL handshake failed as the CRL is not the yet valid. This error occurs because the system clock is not Today's Date.
27	Certificate has expired	The SSL handshake failed as the certificate is not the yet valid. This error occurs because the system clock is

Sr.No	Error Reason	Description
		not Today's Date.
28	CRL has expired	The SSL handshake failed as the CRL is not the yet valid. This error occurs because the system clock is not Today's Date.
29	Format error in certificate's notBefore field	The certificate notBefore field contains an invalid time.
30	Format error in certificate's notAfter field	The CRL notAfter field contains an invalid time.
31	Format error in CRL's lastUpdate field	The CRL lastUpdate field contains an invalid time.
32	Format error in CRL's nextUpdate field	The CRL nextUpdate field contains an invalid time.
33	Out of memory	An error occurred trying to allocate memory.
34	Self signed certificate	The passed certificate is self-signed and the same certificate cannot be found in the list of trusted certificates.
35	Self signed certificate in certificate chain	The certificate chain could be built up using the untrusted certificates, but the root could not be found locally.
36	Unable to get local issuer certificate	The issuer certificate could not be found. This error occurs if the issuer certificate of an untrusted certificate cannot be found.
37	Unable to verify the first certificate	No signatures could be verified because the chain contains only one certificate, and it is not self-signed.
38	Certificate chain too long	The certificate chain length is greater than the supplied maximum depth.
39	Certificate revoked	The server certificate has been

Sr.No	Error Reason	Description
	Cert revoked	revoked.
40	Invalid CA certificate	The server has an invalid the CA certificate.
41	Path length constraint exceeded	The basic Constraints path length parameter has been exceeded.
42	Unsupported certificate purpose	The supplied certificate cannot be used for the specified purpose.
43	Certificate not trusted	The root CA is not marked as trusted for the specified purpose.
44	Certificate rejected	The root CA is marked to reject the specified purpose.
45	Application verification failure	An error occurred due to application specific error.
46	Subject issuer mismatch	An error occurred due to the certificate Subject issuer mismatch. This could be because the subject name in the trusted certificates file and path are different.
47	Authority and subject key identifier mismatch	The issuer's certificate authority key identifier and the subject key identifier mismatched.
48	Key usage does not include certificate signing	The current candidate issuer certificate was rejected because its key usage extension does not permit certificate signing.
49	Unable to get CRL issuer certificate	The issuer certificate of a looked up CRL could not be found. This happens because the list of trusted certificates is incomplete.
50	Unhandled critical extension	The certificate was rejected if an unhandled critical extension is present which is not supported by OpenSSL.

Sr.No	Error Reason	Description
51	Key usage does not include CRL signing	The current candidate issuer certificate was rejected because its keyUsage extension does not permit certificate signing.
52	Unhandled critical CRL extension	The CRL was rejected if an unhandled critical extension is present which is not supported by OpenSSL.
	The below section lists the error occurs when the SSL handshake takes place between client and server.	
53	Client SSL, Receive SSL Fatal Alert	The SSL fatal alert is encountered from client side during SSL handshake.
54	Client SSL, Connection finish by peer	The client closes the connection by sending the FIN packet during SSL handshake.
55	Client SSL, Connection reset by peer	The client resets the connection by sending the RST packet during SSL handshake.
56	SSL Session, Receive TCP FIN/RST packet	The SSL session failed due to TCP FIN/RST from the origin server. This can happen when the SSLi decrypt zone does not support a cipher suite of the origin server.
57	Server SSL, Receive SSL Fatal Alert	The SSL fatal alert is encountered from server-side during SSL handshake.
58	Server SSL, Connection finish by peer	The server closes the connection by sending the FIN packet during SSL handshake.
59	Server SSL, Connection reset by peer	The server resets the connection by sending the RST packet during SSL handshake.
60	SSLi encrypt side cannot do	The SSL negotiation with the client

Sr.No	Error Reason	Description
	HTTP/2	failed to establish HTTP/2 connection.
61	Client SSL, Undefined fail reason	The SSL handshake failure may occur due to undefined reason.
62	Server SSL, Undefined fail reason	The SSL handshake failure may occur due to undefined reason.
	The below section lists the errors in Cavium N3 during hardware encryption or decryption process.	
63	Invalid (Undefined) Opcode	The hardware error occurred due to invalid or undefined operation code (Opcode).
64	Authentication failed due to bad record	The authentication failed due to bad record.
65	Number of scatter elements are zero, or number of gather elements are zero, or Dlen is less than the SG size calculated from s_Len and g_size	Number of scatter elements and number of gather elements should not be zero. And Dlen should not be less than the SG size calculated from s_Len and g_size.
66	Resulting point is at infinity	The resulting point is invalid, it should not be at infinity.
67	Opcode is not point Addition, Double or Multiply	The Opcode is incorrect, it should be Addition, Double or Multiply.
68	Curve is not Prime Curve	The curve is not a prime curve.
69	Prime Curve Length is not 256 or 384.	The prime curve length is not 256 or 384.
70	Additional input length > 656 bytes.	The additional input length is less than 656 bytes.
71	Authentication failed due to bad HMAC digest	The authentication failed due to invalid HMAC digest record.
72	Specified total length of scatter buffers is less than calculated Result Length.	The specified total length of scatter buffers is less than calculated result length.
73	Exponent length, module length	An error occurred due to invalid

Sr.No	Error Reason	Description
	or data length is not in valid range.	length range was provided in exponent, module, or data.
74	Key_Len != < 16, 24, or 32 bytes.	An error occurred due to incorrect key length.
75	TLS1.2 and MAC algorithm is other than MD5, SHA1, SHA256, SHA384, AES-GCM-128, AESGCM-256, or HMAC_Type != NULL, MD5, or SHA1	An error occurred due to incorrect cipher was passed in TLS1.2 and MAC algorithm.
76	In TLS1.2, Encrypt type is other than AES-128, AES-256, 3DES, RC4-128	An invalid encryption was passed in TLS 1.2.
77	Leading byte non-zero or Bad padding, or Bad pad type, or Pad too short, or Data too large, or No zero_byte found or Leading byte is non zero.	An error occurred due to possible issues in the leading byte length.
78	No Authentication and no cipher.	An error occurred due to no authentication or cipher was passed.
79	Cipher type is not supported.	An error occurred due to unsupported cipher type.
80	Authentication type is not supported.	An error occurred due to unsupported authentication type.
81	Encrypt length is not aligned.	The encrypted length is not aligned correctly.
82	Authentication error due to bad MAC.	An authentication error occurred due to invalid MAC record.
83	ControlWord with Scatter Gather (SG) mode not supported.	An error occurred due to unsupported ControlWord with Scatter Gather (SG) mode was used.
84	AES_Type != 128, 192 or 256.	An error occurred due to incorrect value in AES_Type.

Sr.No	Error Reason	Description
85	ControlWord with other than CBC mode not supported.	An error occurred due to unsupported ControlWord with other than CBC mode was used.
86	If DSIV with SSL and TLS1.0.	The DISV has invalid parameters, SSL and TLS 1.0.
	The below section lists the error in Cavium N5 during hardware encryption or decryption process.	
87	Crypto request timeout error.	An error occurred because the crypto error timed out.
88	POM length invalid.	An error occurred due to invalid POM length.
89	Invalid record length, length is 0.	An error occurred due to invalid record length.
90	Invalid context length: not CTXL-L=22 words.	An error occurred due to invalid context length.
91	Unsupported cipher select: cipher is AESGCM for protocols other than TLS1.2/DTLS1.2.	An error occurred due to unsupported cipher selection.
92	MAC_Select > SHA2 for TLS1.0/TLS1.1/ DTLS/DTLS1.0 or MAC_Select != SHA384 for AES256-GCM and MAC_Select != SHA256 for AES128-GCM.	An error occurred due to incorrect MAC value was mapped with the cipher.
93	Unsupported protocol version.	An error occurred due to unsupported protocol version.
94	Unsupported verify hash select. Param2[15:12] is set for protocols other than TLS1.2/DTLS1.2.	The selected hash value was unable to verify or unsupported.
95	Invalid pre-master secret length: length is other than $4 < \text{PMS} < 256$ for SSLv3 or $16 < \text{PMS} < 256$ for TLS/DTLS.	An error occurred due to invalid pre-master secret key length.

Sr.No	Error Reason	Description
96	Pad length invalid and MAC miscompare (bad record).	The pad length is invalid, and the MAC record was unable to compare in the cipher block.
97	Invalid pre-master secret length from DPTR. Error is returned when PMS and first block of handshake data (hash block size) are not in the first BMI buffer.	An error occurred due to invalid pre-master secret key length in the data pointer (DPTR).
	The below section lists the errors when OCSP stapling or connection takes place.	
98	OCSP stapling, decode error, length mismatch	The response received from the cert status could not be decoded due to length mismatch.
99	OCSP stapling, decode error, unsupported status type	The response received from the certificate status could not be decoded due to unsupported status type.
100	OCSP stapling, lack of memory error	The lack of memory error occurred while receiving the response of certificate status during OCSP stapling.
101	OCSP stapling, invalid status response, bad certificate status	An invalid certificate status response is received during OCSP stapling.
102	OCSP stapling, SSL get message failed	Failed to get certificate status response during OCSP stapling.
103	OCSP Revoked	The server certificate is revoked.
104	OCSP connection failed, undefined SVM type or failed to set vwire l2 information	The OCSP connection failed due to undefined SVM type or failed to set virtual wire L2 information.
105	OCSP failed, can not find certificate chain	The OCSP connection failed due to untrusted self-signed certificate in the chain.
106	OCSP failed, can not find peer ssl	The OCSP connection failed when the SSL data on the peer channel cannot be found.

Sr.No	Error Reason	Description
107	OCSP failed, can not malloc for svm_ctx	The OCSP connection failed when the malloc function failed to allocate memory for data structure which stores svm information.
108	OCSP failed, can not find cert key of peer ssl	The OCSP connection failed when the peer SSL could not find certificate key.
109	Cert Status Unknown	The OCSP failed due to an undefined status.
110	Unsupported SSL Version	The server reports incompatible or unsupported SSL/TLS version.
111	Bypassed by aFlex process	SSLi is bypassed due to the rules set in aFlex scripts. The aFlex bypass process encounters an error.
112	Bypassed due to failsafe	SSLi is bypassed due to fail safe.
113	Bypassed by configuration rules	SSLi is bypassed due to configuration rules.
	The below section lists the internal error reasons (OpenSSL library) occurred in Client or Server template.	
114	app data in handshake	The client has sent close_notify and is expecting a close_notify back from the server, but instead there is application data to be read first.
115	attempt to reuse session in different context	The client is attempting to reuse the session in different context.
116	bad alert record	The SSL certificate has invalid alert record.
117	bad authentication type	The SSL certificate has invalid authentication type.
118	bad change cipher spec	The SSL certificate has invalid cipher specification.
119	bad checksum	The SSL certificate has invalid check-

Sr.No	Error Reason	Description
		sum or failed checksum verification.
120	bad data returned by callback	The SSL certificate has returned invalid data by callback function.
121	bad decompression	The compressed data could not be decompressed.
122	bad dh g length	The SSL certificate has invalid modulus g length in Diffie-Hellman.
123	bad dh pub key length	The SSL certificate has invalid public length in Diffie-Hellman.
124	bad dh p length	The SSL certificate has invalid modulus p length in Diffie-Hellman.
125	bad digest length	The SSL certificate has invalid MD5 message-digest length.
126	bad dsa signature	The SSL certificate has invalid DSA signature.
127	bad hello request	The SSL certificate has invalid hello request.
128	bad length	The SSL certificate has invalid length.
129	bad mac decode	The SSL certificate is invalid or could not decode the MAC.
130	bad message type	The SSL certificate has invalid message type.
131	bad packet length	The SSL certificate has invalid packet length.
132	bad protocol version number	The SSL certificate has invalid protocol version number.
133	bad response argument	The SSL certificate has invalid response argument.
134	bad rsa decrypt	The SSL certificate has invalid data exception during RSA decrypt.
135	bad rsa encrypt	The SSL certificate has invalid data

Sr.No	Error Reason	Description
		exception during RSA encrypt.
136	bad rsa e length	The SSL certificate has invalid modulus e length in RSA.
137	bad rsa modulus length	The SSL certificate has invalid modulus length in RSA.
138	bad rsa signature	The SSL certificate has invalid RSA signature.
139	bad signature	The SSL certificate has invalid signature.
140	bad ssl filetype	The SSL certificate has invalid filetype.
141	bad ssl session id length	The SSL certificate has invalid session ID length.
142	bad state	The SSL certificate has invalid state.
143	bad write retry	The SSL_write function failed while sending the response.
144	bio not set	The size of the memory BIO is not set.
145	block cipher pad is wrong	The SSL certificate has invalid cipher pad is incorrect.
146	bn lib	The memory allocation error may have occurred due to BIGNUM (BN) object.
147	ca dn length mismatch	The CA certificate distinguished name (DN) length mismatched.
148	ca dn too long	The CA certificate distinguished name (DN) length is too long.
149	ccs received early	The ChangeCipherSpec (ccs) messages received before the finished function and after the master secret has been generated.
150	certificate verify failed	The SSL certificate verification failed.

Sr.No	Error Reason	Description
151	cert length mismatch	The SSL certificate length mismatched.
152	challenge is different	The SSL certificate challenge password was invalid or different.
153	cipher code wrong length	The SSL certificate cipher code has incorrect length.
154	cipher or hash unavailable	The SSL certificate cipher or hash was unavailable.
155	cipher table src error	An error occurred in the cipher table source.
156	compressed length too long	The SSL certificate compressed length was too long.
157	compression failure	The SSL certificate compression failed.
158	compression library error	An error occurred in the compression library.
159	connection id is different	The SSL connection ID was different.
160	connection type not set	The SSL connection type was not set.
161	data between ccs and finished	The data between ChangeCipherSpec (ccs) messages and the finished function may not be encrypted.
162	data length too long	The SSL certificate or chain length was too long.
163	decryption failed	The SSL certificate decryption failed.
164	decryption failed or bad record mac	The SSL certificate decryption failed or has invalid MAC record.
165	dh key too small	The Diffie-Hellman key length was too small.
166	dh public value length is wrong	The Diffie-Hellman public value length was incorrect.
167	digest check failed	The SSL certificate digest calculation

Sr.No	Error Reason	Description
		or check failed.
168	encrypted length too long	The SSL certificate encrypted length was too long.
169	error generating tmp rsa key	An error occurred while generating the temporary RSA key.
170	error in received cipher list	An error occurred in the cipher list.
171	excessive message size	The large or excessive packets were exchanged during the SSL handshake.
172	extra data in message	The message has extra data.
173	got a fin before a ccs	The finished function occurred before sending the ChangeCipherSpec (ccs) messages.
174	https proxy request	The HTTP request through HTTPS proxy failed.
175	http request	The HTTP request failed.
176	illegal padding	The certificate serial number may contain illegal extra padding.
177	inappropriate fallback	The inappropriate fallback occurred due to downgraded TLS version.
178	invalid challenge length	The SSL certificate challenge length was invalid.
179	inconsistent extms	The master key extension was inconsistent.
180	invalid command	An invalid command was executed from OpenSSL.
181	invalid purpose	The purpose of the certificate verification was invalid.
182	invalid status response	The callback status response was invalid.
183	invalid trust	The trusted certificate is invalid.
184	key arg too long	The passphrase argument was too

Sr.No	Error Reason	Description
		long.
185	krb5	An error occurred in the Kerberos 5 header file.
186	krb5 client cc principal (no tkt?)	The Kerberos 5 ticket was unavailable while exchanging the kerberos principal with the client.
187	krb5 client get cred	The client credentials could not be authenticated on Kerberos 5.
188	krb5 client init	The Kerberos 5 client fails to initialize.
189	krb5 client mk_req (expired tkt?)	The Kerberos 5 client mk_req returns failure.
190	krb5 server bad ticket	The Kerberos 5 server has an invalid ticket.
191	krb5 server init	The Kerberos 5 server fails to initialize.
192	krb5 server rd_req (keytab perms?)	The Kerberos 5 server rd_req returns failure.
193	krb5 server tkt expired	The Kerberos 5 server ticket has expired.
194	krb5 server tkt not yet valid	The Kerberos 5 server ticket is not yet valid.
195	krb5 server tkt skew	The Kerberos 5 server's clock is not within the maximum clock skew.
196	length mismatch	The certificate length has mismatched.
197	length too short	The certificate length was too short.
198	library bug	The Heartbleed security bug was encountered in the OpenSSL cryptography library.
199	library has no ciphers	The OpenSSL library has no ciphers.

Sr.No	Error Reason	Description
200	master key too long	The master key length is too long.
201	message too long	The signature message was too long.
202	missing dh dsa cert	The Diffie-Hellman DSA certificate was missing.
203	missing dh key	The Diffie-Hellman key was missing.
204	missing dh rsa cert	The Diffie-Hellman RSA certificate was missing.
205	missing dsa signing cert	The DSA signing certificate was missing.
206	missing export tmp dh key	The exported temporary Diffie-Hellman key was missing.
207	missing export tmp rsa key	The exported temporary RSA key was missing.
208	missing rsa certificate	The RSA certificate was missing.
209	missing rsa encrypting cert	The RSA encrypted certificate was missing.
210	missing rsa signing cert	The RSA signing certificate was missing.
211	missing tmp dh key	The temporary Diffie-Hellman key was missing.
212	missing tmp rsa key	The temporary RSA key was missing.
213	missing tmp rsa pkey	The temporary RSA private key was missing.
214	missing verify message	The verification message was missing.
215	non sslv2 initial packet	The initial packet is non-SSLv2 protocol.
216	no certificates returned	No certificates were returned during SSL handshake.
217	no certificate assigned	No certificate was assigned to the server.
218	no certificate returned	No certificate was returned during

Sr.No	Error Reason	Description
		SSL handshake.
219	no certificate set	No certificate was set on the server.
220	no certificate specified	No certificate is specified on the server.
221	no ciphers available	No ciphers were available during SSL handshake.
222	no ciphers passed	No ciphers were passed during SSL handshake.
223	no ciphers specified	No ciphers were specified during SSL handshake.
224	no cipher list	No cipher list were available during SSL handshake.
225	no cipher match	No cipher was matched during SSL handshake.
226	no client cert received	No client certificate was received during SSL handshake.
227	no compression specified	No compression was specified during SSL handshake.
228	no method specified	No method was specified during SSL handshake.
229	no privatekey	No private key was available in the certificate.
230	no private key assigned	No private key was assigned in the certificate.
231	no protocols available	No protocols was available during SSL handshake.
232	no publickey	No public key was available in the certificate.
233	no shared cipher	No shared cipher were available during SSL handshake.
234	no verify callback	The callback function was unable to

Sr.No	Error Reason	Description
		verify the certificate.
235	null ssl ctx	The SSL_CTX function returned null value.
236	null ssl method passed	The null SSL method was passed during SSL handshake.
237	old session cipher not returned	The old session cipher was not returned during SSL handshake.
238	packet length too long	The packet length was too long.
239	path too long	The path length was too long.
240	peer did not return a certificate	The peer server did not return a valid certificate.
241	peer error	An error occurred on the peer server.
242	peer error certificate	An error occurred on the peer server certificate.
243	peer error no certificate	No certificate was available on the peer server.
244	peer error no cipher	No cipher was available on the peer server.
245	peer error unsupported certificate type	The peer server certificate type is unsupported.
246	pre mac length too long	The MAC length is too long before encryption.
247	problems mapping cipher functions	An error occurred while mapping the cipher function.
248	protocol is shutdown	The protocol is shutdown during SSL handshake.
249	public key encrypt error	An error occurred during public key encryption.
250	public key is not rsa	The public key is not RSA.
251	public key not rsa	The public key is not RSA.
252	read bio not set	The read function is not set in the

Sr.No	Error Reason	Description
		BIO library.
253	read wrong packet type	The incorrect packet type was read.
254	record length mismatch	The length of a data record does not match the length of the current record position
255	record too large	The length of a data record is too large.
256	record too small	The length of a data record is too small.
257	required cipher missing	The required cipher is missing from the certificate.
258	reuse cert length not zero	The reused certificate length is non zero.
259	reuse cert type not zero	The reused certificate type is non zero.
260	reuse cipher list not zero	The reused cipher list is non zero.
261	scsv received when renegotiating	The renegotiating_info extension is missing.
262	session id context uninitialized	The session ID context was not initialized.
263	short read	An error occurred when the connection ended before the server could send enough data to decode the current message.
264	signature for non signing certificate	The signature was invalid in the SSL certificate.
265	ssl23 doing session id reuse	The SSLv23_method is called when it is unclear which protocol to use.
266	ssl2 connection id too long	The SSLv2_method connection ID is too long.
267	ssl3 session id too long	The SSLv3_method session ID is too long.

Sr.No	Error Reason	Description
268	ssl3 session id too short	The SSLv3_method session ID is too short.
269	ssl3 ext invalid servername	The client received an invalid server name extension.
270	ssl3 ext invalid servername type	The client received an invalid server name type extension.
271	ssl3 alert bad certificate	The SSLv3 method alerted invalid certificate.
272	ssl3 alert bad record mac	The SSLv3 method alerted invalid MAC record.
273	ssl3 alert certificate expired	The SSLv3 method alerted an expired certificate.
274	ssl3 alert certificate revoked	The SSLv3 method alerted the certificate is revoked.
275	ssl3 alert certificate unknown	The SSLv3 method alerted the certificate is unknown.
276	ssl3 alert decompression failure	The SSLv3 method alerted failure during decompression.
277	ssl3 alert handshake failure	The SSLv3 method alerted failure during SSL handshake.
278	ssl3 alert illegal parameter	The SSLv3 method alerted an illegal parameter in the certificate.
279	ssl3 alert no certificate	The SSLv3 method alerted no certificate.
280	ssl3 alert peer error certificate	The SSLv3 method alerted error in the peer server certificate.
281	ssl3 alert peer error no certificate	The SSLv3 method alerted no certificate on the peer server.
282	ssl3 alert peer error no cipher	The SSLv3 method alerted no cipher in the peer server certificate.
283	ssl3 alert peer error unsupported certificate type	The SSLv3 method alerted unsupported certificate type

Sr.No	Error Reason	Description
		ported certificate type on the peer server.
284	sslv3 alert unexpected message	The SSLv3 method alerted unexpected message.
285	sslv3 alert unknown remote error type	The SSLv3 method alerted unknown remote error type.
286	sslv3 alert unsupported certificate	The SSLv3 method alerted unsupported certificate.
287	ssl ctx has no default ssl version	The SSL_CTX methods do not have default SSL version.
288	ssl handshake failure	The SSL handshake failed.
289	ssl library has no ciphers	The SSL library has no ciphers.
290	ssl session id callback failed	The callback function failed to retrieve the SSL session ID.
291	ssl session id conflict	The SSL session ID conflicts with the original session ID.
292	ssl session id context too long	The SSL session ID context is too long.
293	ssl session id has bad length	The SSL session ID has invalid length.
294	ssl session id is different	The server returned different SSL session ID.
295	tlsv1 alert access denied	The TLSv1 method alerted access denied due to certificate authentication failure.
296	tlsv1 alert decode error	The TLSv1 method alerted decode error because some field was out of the specified range, or the length of the message was incorrect.
297	tlsv1 alert decryption failed	The TLSv1 method alerted decryption failed when the TLS Cipher text record decrypted in an invalid way.

Sr.No	Error Reason	Description
298	tlsv1 alert decrypt error	The TLSv1 method alerted decryption failed because some field was out of the specified range, or the length of the message was incorrect.
299	tlsv1 alert export restriction	The TLSv1 method alerted export restriction because it detected a negotiation that was not in compliance with export restrictions.
300	tlsv1 alert insufficient security	The TLSv1 method alerted insufficient security.
301	tlsv1 alert internal error	The TLSv1 method alerted internal error.
302	tlsv1 alert no renegotiation	The TLSv1 method alerted no renegotiation in response to a hello request sent by the server or client after initial handshaking.
303	tlsv1 alert protocol version	The TLSv1 method alerted unsupported or unrecognized protocol version.
304	tlsv1 alert record overflow	The TLSv1 method alerted record overflow.
305	tlsv1 alert unknown ca	The TLSv1 method alerted unknown CA certificate.
306	tlsv1 alert user cancelled	The TLSv1 method alerted when handshake is cancelled due to unrelated protocol failure.
307	tls client cert req with anon cipher	Creating an SSL connection using the openssl stream driver a client certificate is required.
308	tls peer did not respond with certificate list	The peer server did not respond with the certificate list.
309	tls rsa encrypted value length is wrong	The RSA encryption value length is incorrect.

Sr.No	Error Reason	Description
310	tried to use unsupported cipher	The cipher does not support TLS version.
311	unable to decode dh certs	The SSL handshake failed due to Diffie-Hellman certificate could not be decoded.
312	unable to extract public key	The SSL handshake failed because the public key cannot be extracted from certificate.
313	unable to find dh parameters	The SSL handshake failed because the Diffie-Hellman parameter in the certificate was not found.
314	unable to find public key parameters	The SSL handshake failed because the public key parameter in the certificate could not be found.
315	unable to find ssl method	The SSL handshake failed as unable to find SSL encryption method.
316	unable to load ssl2 md5 routines	The SSL handshake failed because the SSL2 and MD5 routines were unable to load.
317	unable to load ssl3 md5 routines	The SSL handshake failed because the SSL3 and MD5 routines were unable to load.
318	unable to load ssl3 sha1 routines	The SSL handshake failed because the SSL3 and SHA1 routines were unable to load.
319	unexpected message	The SSL handshake failed due to unexpected message.
320	unexpected record	The SSL handshake failed due to unexpected record.
321	uninitialized	The SSL handshake was unable to initialize.
322	unknown alert type	The SSL handshake failed due to

Sr.No	Error Reason	Description
		unknown alert type.
323	unknown certificate type	The SSL handshake failed due to unknown certificate type.
324	unknown cipher returned	The SSL handshake failed due to unknown cipher returned.
325	unknown cipher type	The SSL handshake failed due to unknown cipher type.
326	unknown key exchange type	The SSL handshake failed due to unknown key exchange type.
327	unknown pkey type	The SSL handshake failed due to unknown pkey type.
328	unknown protocol	The SSL handshake failed due to unknown protocol.
329	unknown remote error type	The SSL handshake failed due to unknown error from TLS Client
330	unknown ssl version	The SSL handshake failed due to unknown SSL version.
331	unknown state	The SSL handshake failed due to unknown state.
332	unsupported cipher	The SSL handshake failed due to unsupported cipher.
333	unsupported compression algorithm	The SSL handshake failed due to unsupported compression algorithm
334	unsupported option	The SSL handshake failed due to unsupported option.
335	unsupported protocol	The SSL handshake failed due to unsupported protocol.
336	unsupported ssl version	The SSL handshake failed due to unsupported SSL version.
337	unsupported status type	The SSL handshake failed due to unsupported verification status type.

Sr.No	Error Reason	Description
338	write bio not set	The write function is not set in the BIO library.
339	wrong cipher returned	The SSL handshake failed due to incorrect cipher returned.
340	wrong message type	The SSL handshake failed due to incorrect message type.
341	wrong number of key bits	The SSL handshake failed due to incorrect number of key bits.
342	wrong signature length	The SSL handshake failed due to incorrect signature length.
343	wrong signature size	The SSL handshake failed due to incorrect signature size.
344	wrong ssl version	The SSL handshake failed due to incorrect SSL version.
345	wrong version number	The SSL handshake failed due to incorrect SSL version number.
346	x509 lib	An error occurred in x509 library.
347	x509 verification setup problems	An error occurred while verifying x509 certificate.
348	clienthello tlsext	The ClientHello does not contain the supported point formats extension.
349	parse tlsext	The client was unable to parse the ServerHello point formats extension.
350	serverhello tlsext	The ServerHello does not contain the supported point formats extension.
351	too many warn alerts	An error occurred due to too many warning alerts.

Examples

The CEF error example is provided below. The error reason (mentioned in the above table) will be displayed in the `cs7` field as shown below.

CEF Error Log

```
May  4 02:07:41 AX1030 CEF:0|A10|CFW|5.3.0-d|SSLi 486706518616440835|SSL
Inspection Failed|5|src=172.16.1.95 dst=23.185.0.3 spt=40434 dpt=443
act=bypassed dhost=a10networks.com cs1=in_vip_wild cs1Label=VIP name
cs2=https cs2Label=VIP protocol cs3=TLSv1.2 cs3Label=SSL version cs6=OCSP_
UNKNOWN cs6Label=Certificate Validity status cs7=OCSP connection failed,
undefined SVM type or failed to set vwire l2 information cs7Label=Error
type cn1=443 cn1Label=VIP port cn2=6 cn2Label=Log ID cs10=a10networks.com
cs10Label=subject_dn cs11=043eea65632b57091509370b6c86c3dcd199
cs11Label=serial_num
```

Generic Failure Logs

The ACOS_decrypt device in an SSLi configuration generates a system log if SSLi fails. The log includes the SNI, IP address of the outside server that the client was attempting to connect to, and the reason for failure.

- SSL log is generated if the ACOS_decrypt device cannot retrieve the server certificate during the SSL handshake with client.
- SSL Insight can also fail for other reasons such as the SSLi bypass, or abrupt connection closure by server FIN due to malformed packet, and other.
- SSLi failure log messages are only seen by the inside ACOS device.

The SSLi failure logs consist of the similar error reason as mentioned in the [Failure Event Error Reasons](#).

NOTE: No CLI configurations are required to turn logging on or off.

SSLi Bypass Logs

The following example shows logs generated when the SSLi is bypassed or otherwise fails. Client auth bypass will be treated as handshake failure:

```
ACOS-Inside# show log | include SSL intercept failed
...
Nov 10 2016 16:02:03 Info          [SYSTEM]:SSL intercept failed. server
(null) (Src port: 43461 Src IP: 61.61.61.61  Dst port: 47873 Dst IP:
51.51.51.51) reason: Can't verify Cert - Decrypted
...
```

SSL CA Verification Failure Logs

The following example shows a log generated when the outside server's certificate fails verification:

```
ACOS# show log | include CA Verification Failed

Nov 10 2016 16:02:03 Info          [SSL]:SSL Server CA Verification Failed
with Host Name: (null) and Destination IP: 51.51.51.51
```

Example

In this example, "SSLVerifyClient require" and "SSLVerifyDepth 10" is set up on APACHE ssl.conf, on the server. The following log shows there was an SSLi failure when retrieving the certificate because no client-side authentication has been configured.

As a result, the following log is generated:

```
ACOS# show log
Log Buffer: 30000
Aug 08 2016 11:44:23 Info          [SYSTEM]:<l3v1> SSL intercept failed, server
example.com (ip 10.10.10.101) reason: Crypto Error - bypassed
ACOS#
```

Additional Failure Logs

```
[SYSTEM]:SSL intercept failed, server vast.bp3871200.btrll.com (ip Src
port: 53161 Src IP: 172.17.20.242 Dst port: 443 Dst IP: 162.208.20.178)
reason: Can't verify Cert - Rejected
Feb 10 2017 18:16:20 Info [SSL]:SSL Server CA Verification Failed
with Host Name: vast.bp3871200.btrll.com and Destination IP:
162.208.20.178
Feb 10 2017 18:16:20 Info [SYSTEM]:SSL intercept failed, server
vast.bp3862928.btrll.com (ip Src port: 53149 Src IP: 172.17.20.242 Dst
port: 443 Dst IP: 162.208.20.178) reason: Can't verify Cert - Rejected
Feb 10 2017 18:16:20 Info [SSL]:SSL Server CA Verification Failed
with Host Name: vast.bp3862928.btrll.com and Destination IP:
162.208.20.178
Feb 10 2017 18:16:20 Info [SYSTEM]:SSL intercept failed, server
(null) (ip Src port: 53018 Src IP: 172.17.1.145 Dst port: 443 Dst IP:
64.4.54.253) reason: Unknown - Bypass
Feb 10 2017 18:16:20 Info [SYSTEM]:SSL intercept failed, server
settings-win.data.microsoft.com (ip Src port: 53017 Src IP: 172.17.1.145
Dst port: 443 Dst IP: 64.4.54.253) reason: Can't verify Cert - Rejected
Feb 10 2017 18:16:20 Info [SSL]:SSL Server CA Verification Failed
with Host Name: settings-win.data.microsoft.com and Destination IP:
64.4.54.253
Feb 10 2017 18:16:20 Info [SYSTEM]:SSL intercept failed, server
settings-win.data.microsoft.com (ip Src port: 56019 Src IP: 172.17.3.165
Dst port: 443 Dst IP: 64.4.54.253) reason: Can't verify Cert - Rejected
Feb 10 2017 18:16:20 Info [SSL]:SSL Server CA Verification Failed
with Host Name: settings-win.data.microsoft.com and Destination IP:
64.4.54.253
Feb 10 2017 18:16:19 Info [SYSTEM]:SSL intercept failed, server
(null) (ip Src port: 53016 Src IP: 172.17.1.145 Dst port: 443 Dst IP:
64.4.54.254) reason: Unknown - Bypass
Feb 10 2017 18:16:19 Info [SYSTEM]:SSL intercept failed, server
vortex-win.data.microsoft.com (ip Src port: 53015 Src IP: 172.17.1.145
Dst port: 443 Dst IP: 64.4.54.254) reason: Can't verify Cert - Rejected
Feb 10 2017 18:16:19 Info [SSL]:SSL Server CA Verification Failed
with Host Name: vortex-win.data.microsoft.com and Destination IP:
64.4.54.254
```

```
Feb 10 2017 18:16:07 Info      [SYSTEM]:SSL intercept failed, server
(null) (ip Src port: 51633 Src IP: 172.17.1.245  Dst port: 443 Dst IP:
40.77.228.92) reason: Unknown - Bypass
Feb 10 2017 18:16:07 Info      [SYSTEM]:SSL intercept failed, server
watson.telemetry.microsoft.com (ip Src port: 51632 Src IP: 172.17.1.245
Dst port: 443 Dst IP: 40.77.228.92) reason: Can't verify Cert - Rejected
```

Event-based Logging

ACOS support sending SSLi event logs over the ACOS event-based logging infrastructure. This provides a centralized logging infrastructure where applications generate and send logs through a common interface.

For instructions on implementing Event-based Logging, refer 'Event Logging System' section in *System Configuration and Administration Guide* and *Event Logging Guide*.

Logging and Exporting SSL Session Secrets

ACOS supports logging SSL session secrets using an aFlex script. This log file can be exported and used to decrypt packet flows in Wireshark or other network forensic tools. This feature utilizes the OpenSSL callback feature to log the SSL session secrets. The logged file adheres to the standardized format defined in the [IETF Keylog File](#).

The key benefits of logging and exporting the SSL session secrets are as follows:

- Decrypt and inspect encrypted traffic.
- Identify and analyze security threats.
- Simplify security auditing and regulatory compliance.

Supported Features:

- Key Exchange Methods: RSA and DH
- SSL Modules: Intel QuickAssist Technology (QAT), Software TLS 1.3, and Nitrox V (N5) hardware acceleration
- aFlex Events: `CLIENTSSL_HANDSHAKE` and `SERVERSSL_HANDSHAKE`.

You can configure the aFlex command `SSL::sessionsecret` in the client-SSL and server-SSL handshake events. Then, bind the aFlex script to the SLB virtual port.

For more information on the aFlex commands, see *aFlex Scripting Language Reference*.

Configuration Example

1. Create a sample script (aflex_SSLsession) using the following command:

```
when CLIENTSSL_HANDSHAKE {
    log "SSL session secrets for current session is
[SSL::sessionsecret]"
}
```

2. Bind the aFlex script to the SLB virtual port using the following command:

```
slb virtual-server VS1 10.10.10.90
    port 443 https
        aflex aflex_SSLsession
```

3. Check if the SSL session secret is logged in the SSL session key file on the Syslog server after the SSL/TLS connection is established between client and server.

The SSL session secrets are logged in the hex string format: **<label> <client random> <secret>**. The following is the sample SSL session key file:

```
SERVER_HANDSHAKE_TRAFFIC_SECRET
19acc5019abe56ac3d531fa7f8ef54f57b9bcb3edc14bdadabe51a28d37d9ec4
3cad4c78ae9cad4f73f1b2be0c0b1ebba736107247111b0dc16b392077b233cf822c1e
d354eb456d28feb2e71e3ea5db
EXPORTER_SECRET
19acc5019abe56ac3d531fa7f8ef54f57b9bcb3edc14bdadabe51a28d37d9ec4
c0fa0ecff7795b9ab9874f5e483abe9444b24de47d29d5cae1aa9be07e07547941715d
bfb93ca4bec178b493390787a1
SERVER_TRAFFIC_SECRET_0
19acc5019abe56ac3d531fa7f8ef54f57b9bcb3edc14bdadabe51a28d37d9ec4
03bfbcb47a4c9f9bb22c919450fdcb4992f8f52d1de6bb3c16234f2c02cd9239ba5a549
38a62c6af880f35bb5459f6b1d
```

```
CLIENT_HANDSHAKE_TRAFFIC_SECRET
19acc5019abe56ac3d531fa7f8ef54f57b9bcb3edc14bdadabe51a28d37d9ec4
dd949b485c6347a280e48efe2f869133ae4bb85f7b7886ab287f6fb2a5fa6c7c001a01
77851cefa91785431f0e4da438
CLIENT_TRAFFIC_SECRET_0
19acc5019abe56ac3d531fa7f8ef54f57b9bcb3edc14bdadabe51a28d37d9ec4
8b29b981be9eda270297fbe036d461edacf56a3a35fd25b0f6b9df43444cc12ee77787
6c6164420bce0a7c5323a5d5f2
```

4. Copy the SSL session secrets to a file called SSL Session Key.
5. Manually export or load the SSL session key file into the Wireshark/tshark of the connection to decrypt the packets.

```
tshark -o tls.keylog_file:<key_file_name>
```

Limitation

- For TLS 1.3, only the first set of application secrets (CLIENT_TRAFFIC_SECRET_0 and SERVER_TRAFFIC_SECRET_0) are logged. Any new secrets generated with TLS 1.3 KeyUpdate are not logged.
- If you use the 'show log' command to view the secret session key, the complete secret key for the SSL session will not be displayed. This is because the 'show log' command output has a maximum limit of 1024 bytes for each log.

Glossary

B

bypass

A go-around functionality, where a specific action is performed via an external or alternative route instead of the intended route. In network security, a bypass is defined as a security system flaw that allows attackers access to network by circumventing the security mechanism.

D

DNS

Domain Name System. A hierarchical model and decentralized naming system that identifies computers, resources and network-based services over a private network or the Internet. It specifies information on web domain names associated with respective entities.

DSR

Direct Server Return. A load balancing mode where packets are routed to the backend server by modifying only the destination MAC address.

F

failover

A backup operational mode that allows the functions of a system component such as a network, server or database to be assumed by secondary components in instances when the primary components are unavailable due to failure or downtime.

G

gateway

A hardware device such as a firewall, router, or server, that acts as a gate between two networks and allows

the inward and outward flow of traffic among the networks. It secures the nodes within a network and also serves as a node itself.

H

HTML

Hypertext Markup Language. The standard markup language developed for displaying documents in a web browser.

HTTP

HyperText Transfer Protocol. An underlying web protocol that defines the way messages can be formatted and sent, and the actions to be taken by web servers and browsers for responding to multiple commands.

I

IPv4

The fourth version of the Internet Protocol used as a core protocol in standardized internetworking methods over the Internet and packet-switched networks.

L

L3

A Network Layer, the third layer in the seven-layered OSI reference model used for routing traffic and forwarding packets across intermediate routers.

L4

A Transport Layer, the fourth layer of the seven-layered OSI reference model used for establishing host-to-host communications for applications.

load balancing

The process of distributing a set of tasks over a set of resources for making their overall processing more efficient and improving the performance.

P**packet flow**

A sequence of packets sent from a source to a destination over packet-switching networks, across a host, a broadcast domain, or a multicast group.

S**service-group**

A group of one or more services linked together for making object configurations simple.

SIP

Session Initiation Protocol. A signaling protocol that initiates,

maintains, and ends real-time voice, video and messaging sessions of applications.

SMPP

Short Message Peer-to-Peer. A standardized protocol that provides scalable data communication interfaces for short message data transmissions between ESME, message centres, and routing entities.

source NAT pool

A pool of source NAT protocols used when an internal host begins a session with an external host and a dual NAT without using the switch IP.

SPDY

Speedy. A deprecated and open-spec networking protocol, developed by Google, which transports web content and manages HTTP traffic at a high

speed by lowering webpage load latency and boosting web security.

subnet

An IP network subdivision.

T

TCP

Transmission Control Protocol. Key part of the main IP suite protocols used during initial network implementation.

TLS

Transport Layer Security. A cryptographic protocol that provides data transport and communications security over a network.

U

UDP

User Datagram Protocol. An alternative to TCP and used for setting up

connections with low-latency and loss-tolerance between internet applications.

URL

Uniform Resource Locator. A web address that works as a reference to specify the location of a web resource on a computer network and also runs a mechanism for its retrieval.

V

VIP

Virtual Internet Protocol. An IP address which does not correspond to any real physical network interface but is used for mobility, network address translation, and fault-tolerance.

virtual port

An emulation or virtualization of a hardware port.

VLAN

Virtual Local Area Network. A LAN broadcast domain which is separated and isolated at the data link layer in a network.





©2025 A10 Networks, Inc. All rights reserved. A10 Networks, the A10 Networks logo, ACOS, A10 Thunder, Thunder TPS, A10 Harmony, SSLi and SSL Insight are trademarks or registered trademarks of A10 Networks, Inc. in the United States and other countries. All other trademarks are property of their respective owners. A10 Networks assumes no responsibility for any inaccuracies in this document. A10 Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice. For the full list of trademarks, visit: www.a10networks.com/company/legal/trademarks/.