Configuring Communication Services

This chapter includes the following sections:

Communication Services

You can use the communication services defined below to interface third-party applications with Cisco UCS.

Cisco UCS Manager supports both IPv4 and IPv6 address access for the following services:

- CIM XML
- HTTP
- HTTPS
- SNMP
- SSH
- Telnet

Cisco UCS Manager supports out-of-band IPv4 address access to the **Cisco UCS KVM Direct** launch page from a web browser. To provide this access, you must enable the following service:

- CIMC Web Service

Communication Service	Description
CIM XML	This service is disabled by default and is only available in read-only mode. The default port is 5988. This common information model is one of the standards defined by the Distributed Management Task Force.
CIMC Web Service	This service is disabled by default. When this service is enabled, users can directly access a server CIMC using one of the out-of-band management IP addresses assigned directly to the server, or associated with the server through a service profile. **Note** CIMC Web Service can only be enabled or disabled globally. You cannot configure KVM direct access for individual CIMC IP addresses.
HTTP	This service is enabled on port 80 by default. You must enable either HTTP or HTTPS to run Cisco UCS Manager GUI. If you select HTTP, all data is exchanged in clear text mode. For security purposes, we recommend that you enable HTTPS and disable HTTP. By default, Cisco UCS redirects any attempt to communicate via HTTP to the HTTPS equivalent. We recommend that you do not change this behavior. **Note** If you are upgrading to Cisco UCS, version 1.4(1), this does not happen by default. If you want to redirect any attempt to communicate via HTTP to an HTTPS equivalent, you should enable **Redirect HTTP to HTTPS** in Cisco UCS Manager.
HTTPS	This service is enabled on port 443 by default. With HTTPS, all data is exchanged in encrypted mode through a secure server. For security purposes, we recommend that you only use HTTPS and either disable or redirect HTTP communications.
SMASH CLP	This service is enabled for read-only access and supports a limited subset of the protocols, such as the **show** command. You cannot disable it. This shell service is one of the standards defined by the Distributed Management Task Force.
SNMP	This service is disabled by default. If enabled, the default port is 161. You must configure the community and at least one SNMP trap. Enable this service only if your system includes integration with an SNMP server.
SSH	This service is enabled on port 22. You cannot disable it, nor can you change the default port. This service provides access to the Cisco UCS Manager CLI.

Communication Service	Description
Telnet	This service is disabled by default. This service provides access to the Cisco UCS Manager CLI.

Configuring CIM XML

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope system**	Enters system mode.
Step 2	UCS-A /system # **scope services**	Enters system services mode.
Step 3	UCS-A /system/services # **enable cimxml**	Enables the CIM XLM service.
Step 4	UCS-A /system/services # **set cimxml port** *port-num*	Specifies the port to be used for the CIM XML connection.
Step 5	UCS-A /system/services # **commit-buffer**	Commits the transaction to the system configuration.

The following example enables CIM XML, sets the port number to 5988, and commits the transaction:

```
UCS-A# scope system
UCS-A /system # scope services
UCS-A /system/services # enable cimxml
UCS-A /system/services* # set cimxml port 5988
UCS-A /system/services* # commit-buffer
UCS-A /system/services #
```

Configuring HTTP

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope system**	Enters system mode.
Step 2	UCS-A /system # **scope services**	Enters system services mode.
Step 3	UCS-A /system/services # **enable http**	Enables the HTTP service.
Step 4	UCS-A /system/services # **set http port** *port-num*	Specifies the port to be used for the HTTP connection.

	Command or Action	Purpose
Step 5	UCS-A /system/services # **commit-buffer**	Commits the transaction to the system configuration.

The following example enables HTTP, sets the port number to 80, and commits the transaction:

```
UCS-A# scope system
UCS-A /system # scope services
UCS-A /system/services # enable http
UCS-A /system/services* # set http port 80
Warning: When committed, this closes all the web sessions.
UCS-A /system/services* # commit-buffer
UCS-A /system/services #
```

Unconfiguring HTTP

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope system**	Enters system mode.
Step 2	UCS-A /system # **scope services**	Enters system services mode.
Step 3	UCS-A /system/services # **disable http**	Disables the HTTP service.
Step 4	UCS-A /system/services # **commit-buffer**	Commits the transaction to the system configuration.

The following example disables HTTP and commits the transaction:

```
UCS-A# scope system
UCS-A /system # scope services
UCS-A /system/services # disable http
UCS-A /system/services* # commit-buffer
UCS-A /system/services #
```

Configuring HTTPS

Certificates, Key Rings, and Trusted Points

HTTPS uses components of the Public Key Infrastructure (PKI) to establish secure communications between two devices, such as a client's browser and Cisco UCS Manager.

Encryption Keys and Key Rings

Each PKI device holds a pair of asymmetric Rivest-Shamir-Adleman (RSA) encryption keys, one kept private and one made public, stored in an internal key ring. A message encrypted with either key can be decrypted with the other key. To send an encrypted message, the sender encrypts the message with the receiver's public key, and the receiver decrypts the message using its own private key. A sender can also prove its ownership of a public key by encrypting (also called 'signing') a known message with its own private key. If a receiver can successfully decrypt the message using the public key in question, the sender's possession of the corresponding private key is proven. Encryption keys can vary in length, with typical lengths from 512 bits to 2048 bits. In general, a longer key is more secure than a shorter key. Cisco UCS Manager provides a default key ring with an initial 1024-bit key pair, and allows you to create additional key rings.

The default key ring certificate must be manually regenerated if the cluster name changes or the certificate expires.

This operation is only available in the UCS Manager CLI.

Certificates

To prepare for secure communications, two devices first exchange their digital certificates. A certificate is a file containing a device's public key along with signed information about the device's identity. To merely support encrypted communications, a device can generate its own key pair and its own self-signed certificate. When a remote user connects to a device that presents a self-signed certificate, the user has no easy method to verify the identity of the device, and the user's browser will initially display an authentication warning. By default, Cisco UCS Manager contains a built-in self-signed certificate containing the public key from the default key ring.

Trusted Points

To provide stronger authentication for Cisco UCS Manager, you can obtain and install a third-party certificate from a trusted source, or trusted point, that affirms the identity of your device. The third-party certificate is signed by the issuing trusted point, which can be a root certificate authority (CA) or an intermediate CA or trust anchor that is part of a trust chain that leads to a root CA. To obtain a new certificate, you must generate a certificate request through Cisco UCS Manager and submit the request to a trusted point.

Important The certificate must be in Base64 encoded X.509 (CER) format.

Creating a Key Ring

Cisco UCS Manager supports a maximum of 8 key rings, including the default key ring.

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope security**	Enters security mode.
Step 2	UCS-A /security # **create keyring** *keyring-name*	Creates and names the key ring.
Step 3	UCS-A /security/keyring # **set modulus** {**mod1024** | **mod1536** | **mod2048** | **mod512**}	Sets the SSL key length in bits.

	Command or Action	Purpose
Step 4	UCS-A /security/keyring # **commit-buffer**	Commits the transaction.

The following example creates a keyring with a key size of 1024 bits:

```
UCS-A# scope security
UCS-A /security # create keyring kr220
UCS-A /security/keyring* # set modulus mod1024
UCS-A /security/keyring* # commit-buffer
UCS-A /security/keyring #
```

What to Do Next

Create a certificate request for this key ring.

Regenerating the Default Key Ring

The default key ring certificate must be manually regenerated if the cluster name changes or the certificate expires.

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope security**	Enters security mode.
Step 2	UCS-A /security # **scope keyring default**	Enters key ring security mode for the default key ring.
Step 3	UCS-A /security/keyring # **set regenerate yes**	Regenerates the default key ring.
Step 4	UCS-A /security/keyring # **commit-buffer**	Commits the transaction.

The following example regenerates the default key ring:

```
UCS-A# scope security
UCS-A /security # scope keyring default
UCS-A /security/keyring* # set regenerate yes
UCS-A /security/keyring* # commit-buffer
UCS-A /security/keyring #
```

Creating a Certificate Request for a Key Ring

Creating a Certificate Request for a Key Ring with Basic Options

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope security**	Enters security mode.
Step 2	UCS-A /security # **scope keyring** *keyring-name*	Enters configuration mode for the key ring.
Step 3	UCS-A /security/keyring # **create certreq** {**ip** [*ipv4-addr* | *ipv6-v6*] |**subject-name** *name*}	Creates a certificate request using the IPv4 or IPv6 address specified, or the name of the fabric interconnect. You are prompted to enter a password for the certificate request.
Step 4	UCS-A /security/keyring/certreq # **commit-buffer**	Commits the transaction.
Step 5	UCS-A /security/keyring # **show certreq**	Displays the certificate request, which you can copy and send to a trust anchor or certificate authority.

The following example creates and displays a certificate request with an IPv4 address for a key ring, with basic options:

```
UCS-A# scope security
UCS-A /security # scope keyring kr220
UCS-A /security/keyring # create certreq ip 192.168.200.123 subject-name sjc04
Certificate request password:
Confirm certificate request password:
UCS-A /security/keyring* # commit-buffer
UCS-A /security/keyring # show certreq
Certificate request subject name: sjc04
Certificate request ip address: 192.168.200.123
Certificate request e-mail name:
Certificate request country name:
State, province or county (full name):
Locality (eg, city):
Organization name (eg, company):
Organization Unit name (eg, section):
Request:
-----BEGIN CERTIFICATE REQUEST-----
MIIBfTCB5wIBADARMQ8wDQYDVQQDEwZzYW1jMDQwgZ8wDQYJKoZIhvcNAQEBBQAD
gY0AMIGJAoGBALpKn1t8qMZO4UGqILKFXQQc2c8b/vW2rnRF8OPhKbhghLA1YZ1F
JqcYEG5Yl1+vgohLBTd45s0GC8m4RTLJWHo4SwccAUXQ5Zngf45YtX1WsylwUWV4
0re/zgTk/WCd56RfOBvWR2Dtztu2pGA14sd761zLxt29K7R8mzj6CAUVAgMBAAGg
LTArBgkqhkiG9w0BCQ4xHjAcMBoGA1UdEQEB/wQQMA6CBnNhbWMwNIcECsEiXjAN
BgkqhkiG9w0BAQQFAAOBgQCsxN0qUHYGFoQw56RwQueLTNPnrndqUwuZHUO03Teg
nhsyu4satpyiPqVV9viKZ+spvc6x5PWIcTWgHhH8BimOb/0OKuG8kwfIGGsEDlAv
TTYvUP+BZ9OFiPbRIA718S+V8ndXr1HejiQGxlDNqoN+odCXPc5kjoXD0lZTL09H
BA==
-----END CERTIFICATE REQUEST-----

UCS-A /security/keyring #
```

Creating a Certificate Request for a Key Ring with Advanced Options

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope security**	Enters security mode.
Step 2	UCS-A /security # **scope keyring** *keyring-name*	Enters configuration mode for the key ring.
Step 3	UCS-A /security/keyring # **create certreq**	Creates a certificate request.
Step 4	UCS-A /security/keyring/certreq* # **set country** *country name*	Specifies the country code of the country in which the company resides.
Step 5	UCS-A /security/keyring/certreq* # **set dns** *DNS Name*	Specifies the Domain Name Server (DNS) address associated with the request.
Step 6	UCS-A /security/keyring/certreq* # **set e-mail** *E-mail name*	Specifies the email address associated with the certificate request.
Step 7	UCS-A /security/keyring/certreq* # **set ip** {*certificate request ip-address*|*certificate request ip6-address* }	Specifies the IP address of the Fabric Interconnect.
Step 8	UCS-A /security/keyring/certreq* # **set locality** *locality name (eg, city)*	Specifies the city or town in which the company requesting the certificate is headquartered.
Step 9	UCS-A /security/keyring/certreq* # **set org-name** *organization name*	Specifies the organization requesting the certificate.
Step 10	UCS-A /security/keyring/certreq* # **set org-unit-name** *organizational unit name*	Specifies the organizational unit.
Step 11	UCS-A /security/keyring/certreq* # **set password** *certificate request password*	Specifies an optional password for the certificate request.
Step 12	UCS-A /security/keyring/certreq* # **set state** *state, province or county*	Specifies the state or province in which the company requesting the certificate is headquartered.
Step 13	UCS-A /security/keyring/certreq* # **set subject-name** *certificate request name*	Specifies the fully qualified domain name of the Fabric Interconnect.
Step 14	UCS-A /security/keyring/certreq* # **commit-buffer**	Commits the transaction.
Step 15	UCS-A /security/keyring # **show certreq**	Displays the certificate request, which you can copy and send to a trust anchor or certificate authority.

The following example creates and displays a certificate request with an IPv4 address for a key ring, with advanced options:

```
UCS-A# scope security
UCS-A /security # scope keyring kr220
UCS-A /security/keyring # create certreq
UCS-A /security/keyring/certreq* # set ip 192.168.200.123
UCS-A /security/keyring/certreq* #  set subject-name sjc04
UCS-A /security/keyring/certreq* # set country US
UCS-A /security/keyring/certreq* # set dns bg1-samc-15A
UCS-A /security/keyring/certreq* # set email test@cisco.com
UCS-A /security/keyring/certreq* # set locality new york city
UCS-A /security/keyring/certreq* # set org-name "Cisco Systems"
UCS-A /security/keyring/certreq* # set org-unit-name Testing
UCS-A /security/keyring/certreq* # set state new york
UCS-A /security/keyring/certreq* # commit-buffer
UCS-A /security/keyring/certreq # show certreq
Certificate request subject name: sjc04
Certificate request ip address: 192.168.200.123
Certificate request e-mail name: test@cisco.com
Certificate request country name: US
State, province or county (full name): New York
Locality name (eg, city): new york city
Organization name (eg, company): Cisco
Organization Unit name (eg, section): Testing
Request:
-----BEGIN CERTIFICATE REQUEST-----
MIIBfTCB5wIBADARMQ8wDQYDVQQDEwZzYW1jMDQwgZ8wDQYJKoZIhvcNAQEBBQAD
gY0AMIGJAoGBALpKn1t8qMZO4UGqILKFXQQc2c8b/vW2rnRF8OPhKbhghLA1YZ1F
JqcYEG5Yl1+vgohLBTd45s0GC8m4RTLJWHo4SwccAUXQ5Zngf45YtX1WsylwUWV4
0re/zgTk/WCd56RfOBvWR2Dtztu2pGA14sd761zLxt29K7R8mzj6CAUVAgMBAAGg
LTArBgkqhkiG9w0BCQ4xHjAcMBoGA1UdEQEB/wQQMA6CBnNhbWMwNIcECsEiXjAN
BgkqhkiG9w0BAQQFAAOBgQCsxN0qUHYGFoQw56RwQueLTNPnrndqUwuZHUO03Teg
nhsyu4satpyiPqVV9viKZ+spvc6x5PWIcTWgHhH8BimOb/0OKuG8kwfIGGsEDlAv
TTYvUP+BZ9OFiPbRIA718S+V8ndXr1HejiQGxlDNqoN+odCXPc5kjoXD0lZTL09H
BA==
-----END CERTIFICATE REQUEST-----

UCS-A /security/keyring/certreq #
```

What to Do Next

- Copy the text of the certificate request, including the BEGIN and END lines, and save it in a file. Send the file with the certificate request to a trust anchor or certificate authority to obtain a certificate for the key ring.
- Create a trusted point and set the certificate chain for the certificate of trust received from the trust anchor.

Creating a Trusted Point

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope security**	Enters security mode.
Step 2	UCS-A /security # **create trustpoint** *name*	Creates and names a trusted point.
Step 3	UCS-A /security/trustpoint # **set certchain** [*certchain*]	Specifies certificate information for this trusted point.

	Command or Action	Purpose
		If you do not specify certificate information in the command, you are prompted to enter a certificate or a list of trustpoints defining a certification path to the root certificate authority (CA). On the next line following your input, type ENDOFBUF to finish. **Important** The certificate must be in Base64 encoded X.509 (CER) format.
Step 4	UCS-A /security/trustpoint # **commit-buffer**	Commits the transaction.

The following example creates a trusted point and provides a certificate for the trusted point:

```
UCS-A# scope security
UCS-A /security # create trustpoint tPoint10
UCS-A /security/trustpoint* # set certchain
Enter lines one at a time. Enter ENDOFBUF to finish. Press ^C to abort.
Trustpoint Certificate Chain:
> -----BEGIN CERTIFICATE-----
> MIIDMDCCApmgAwIBAgIBADANBgkqhkiG9w0BAQQFADB0MQswCQYDVQQGEwJVUzEL
> BxMMU2FuIEpvc2UsIENBMRUwEwYDVQQKEwxFeGFtcGxlIEluYy4xEzARBgNVBAsT
> ClRlc3QgR3JvdXAxGTAXBgNVBAMTEHRlc3QuZXhhbXBsZS5jb20xHzAdBgkqhkiG
> 9w0BCQEWEHVzZXJAZXhhbXBsZS5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJ
> AoGBAMZw4nTepNIDhVzb0j7Z2Je4xAG56zmSHRMQeOGHemdh66u2/XAoLx7YCcYU
> ZgAMivyCsKgb/6CjQtsofvtrmC/eAehuK3/SINv7wd6Vv2pBt6ZpXgD4VBNKONDl
> GMbkPayVlQjbG4MD2dx2+H8EH3LMtdZrgKvPxPTE+bF5wZVNAgMBAAGgJTAjBgkq
> hkiG9w0BCQcxFhMUQSBjaGFsbGVuZ2UgcGFzc3dvcmQwDQYJKoZIhvcNAQEFBQAD
> gYEAG61CaJoJaVMhzCl903O6Mg51zq1zXcz75+VFj2I6rH9asckCld3mkOVx5gJU
> Ptt5CVQpNgNLdvbDPSsXretysOhqHmp9+CLv8FDuy1CDYfuaLtvlWvfhevskV0j6
> jtcEMyZ+f7+3yh421ido3nO4MIGeBgNVHSMEgZYwgZOAFLlNjtcEMyZ+f7+3yh42
> 1ido3nO4oXikdjB0MQswCQYDVQQGEwJVUzELMAkGA1UECBMCQ0ExFDASBgNVBAcT
> C1NhbnRhIENsYXJhMRswGQYDVQQKExJOdW92YSBTeXN0ZW1zIEluYy4xFDASBgNV
> BAsTC0VuZ2luZWVyaW5nMQ8wDQYDVQQDEwZ0ZXN0Q0GCAQAwDAYDVR0TBAUwAwEB
> /zANBgkqhkiG9w0BAQQFAAOBgQAhWaRwXNR6B4g6Lsnr+fptHv+WVhB5fKqGQqXc
> wR4pYiO4z42/j9Ijenh75tCKMhW51az8copP1EBmOcyuhf5C6vasrenn1ddkkYt4
> PR0vxGc40whuiozBolesmsmjBbedUCwQgdFDWhDIZJwK5+N3x/kfa2EHU6id1avt
> 4YL5Jg==
> -----END CERTIFICATE-----
> ENDOFBUF
UCS-A /security/trustpoint* # commit-buffer
UCS-A /security/trustpoint #
```

What to Do Next

Obtain a key ring certificate from the trust anchor or certificate authority and import it into the key ring.

Importing a Certificate into a Key Ring

Before You Begin

- Configure a trusted point that contains the certificate chain for the key ring certificate.
- Obtain a key ring certificate from a trust anchor or certificate authority.

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope security**	Enters security mode.
Step 2	UCS-A /security # **scope keyring** *keyring-name*	Enters configuration mode for the key ring that will receive the certificate.
Step 3	UCS-A /security/keyring # **set trustpoint** *name*	Specifies the trusted point for the trust anchor or certificate authority from which the key ring certificate was obtained.
Step 4	UCS-A /security/keyring # **set cert**	Launches a dialog for entering and uploading the key ring certificate. At the prompt, paste the certificate text that you received from the trust anchor or certificate authority. On the next line following the certificate, type ENDOFBUF to complete the certificate input. **Important** The certificate must be in Base64 encoded X.509 (CER) format.
Step 5	UCS-A /security/keyring # **commit-buffer**	Commits the transaction.

The following example specifies the trust point and imports a certificate into a key ring:

```
UCS-A# scope security
UCS-A /security # scope keyring kr220
UCS-A /security/keyring # set trustpoint tPoint10
UCS-A /security/keyring* # set cert
Enter lines one at a time. Enter ENDOFBUF to finish. Press ^C to abort.
Keyring certificate:
> -----BEGIN CERTIFICATE-----
> MIIB/zCCAWgCAQAwgZkxCzAJBgNVBAYTAlVTMQswCQYDVQQIEwJDQTEVMBMGA1UE
> BxMMU2FuIEpvc2UsIENBMRUwEwYDVQQKEwxFeGFtcGxlIEluYy4xEzARBgNVBAsT
> ClRlc3QgR3JvdXAxGTAXBgNVBAMTEHRlc3QuZXhhbXBsZS5jb20xHzAdBgkqhkiG
> 9w0BCQEWEHVzZXJAZXhhbXBsZS5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJ
> AoGBAMZw4nTepNIDhVzb0j7Z2Je4xAG56zmSHRMQeOGHemdh66u2/XAoLx7YCcYU
> ZgAMivyCsKgb/6CjQtsofvtrmC/eAehuK3/SINv7wd6Vv2pBt6ZpXgD4VBNKONDl
> GMbkPayVlQjbG4MD2dx2+H8EH3LMtdZrgKvPxPTE+bF5wZVNAgMBAAGgJTAjBgkq
> hkiG9w0BCQcxFhMUQSBjaGFsbGVuZ2UgcGFzc3dvcmQwDQYJKoZIhvcNAQEFBQAD
> gYEAG61CaJoJaVMhzCl903O6Mg51zq1zXcz75+VFj2I6rH9asckCld3mkOVx5gJU
> Ptt5CVQpNgNLdvbDPSsXretysOhqHmp9+CLv8FDuy1CDYfuaLtvlWvfhevskV0j6
> mK3Ku+YiORnv6DhxrOoqau8r/hyI/L43l7IPN1HhOi3oha4=
> -----END CERTIFICATE-----
> ENDOFBUF
UCS-A /security/keyring* # commit-buffer
UCS-A /security/keyring #
```

What to Do Next

Configure your HTTPS service with the key ring.

Configuring HTTPS

Caution After you complete the HTTPS configuration, including changing the port and key ring to be used by HTTPS, all current HTTP and HTTPS sessions are closed without warning as soon as you save or commit the transaction.

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope system**	Enters system mode.
Step 2	UCS-A /system # **scope services**	Enters system services mode.
Step 3	UCS-A /system/services # **enable https**	Enables the HTTPS service.
Step 4	UCS-A /system/services # **set https port** *port-num*	(Optional) Specifies the port to be used for the HTTPS connection.
Step 5	UCS-A /system/services # **set https keyring** *keyring-name*	(Optional) Specifies the name of the key ring you created for HTTPS.
Step 6	UCS-A /system/services # **set https cipher-suite-mode** *cipher-suite-mode*	(Optional) The level of Cipher Suite security used by the Cisco UCS domain. *cipher-suite-mode* can be one of the following keywords: • **high-strength** • **medium-strength** • **low-strength** • **custom**—Allows you to specify a user-defined Cipher Suite specification string.
Step 7	UCS-A /system/services # **set https cipher-suite** *cipher-suite-spec-string*	(Optional) Specifies a custom level of Cipher Suite security for this Cisco UCS domain if **cipher-suite-mode** is set to **custom**. *cipher-suite-spec-string* can contain up to 256 characters and must conform to the OpenSSL Cipher Suite specifications. You cannot use any spaces or special characters except ! (exclamation point), + (plus sign), - (hyphen), and : (colon). For details, see http://httpd.apache.org/docs/2.0/mod/mod_ssl.html#sslciphersuite.. For example, the medium strength specification string Cisco UCS Manager uses as the default is: ALL:!ADH:!EXPORT56:!LOW:RC4+RSA:+HIGH:+MEDIUM:+EXP:+eNULL

	Command or Action	Purpose
		Note This option is ignored if **cipher-suite-mode** is set to anything other than **custom**.
Step 8	UCS-A /system/services # **commit-buffer**	Commits the transaction to the system configuration.

The following example enables HTTPS, sets the port number to 443, sets the key ring name to kring7984, sets the Cipher Suite security level to high, and commits the transaction:

```
UCS-A# scope system
UCS-A /system # scope services
UCS-A /system/services # enable https
UCS-A /system/services* # set https port 443
Warning: When committed, this closes all the web sessions.
UCS-A /system/services* # set https keyring kring7984
UCS-A /system/services* # set https cipher-suite-mode high
UCS-A /system/services* # commit-buffer
UCS-A /system/services #
```

Deleting a Key Ring

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope security**	Enters security mode.
Step 2	UCS-A /security # **delete keyring** *name*	Deletes the named key ring.
Step 3	UCS-A /security # **commit-buffer**	Commits the transaction.

The following example deletes a key ring:

```
UCS-A# scope security
UCS-A /security # delete keyring key10
UCS-A /security* # commit-buffer
UCS-A /security #
```

Deleting a Trusted Point

Before You Begin

Ensure that the trusted point is not used by a key ring.

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope security**	Enters security mode.
Step 2	UCS-A /security # **delete trustpoint** *name*	Deletes the named trusted point.
Step 3	UCS-A /security # **commit-buffer**	Commits the transaction.

The following example deletes a trusted point:

```
UCS-A# scope security
UCS-A /security # delete trustpoint tPoint10
UCS-A /security* # commit-buffer
UCS-A /security #
```

Unconfiguring HTTPS

Before You Begin

Disable HTTP to HTTPS redirection.

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope system**	Enters system mode.
Step 2	UCS-A /system # **scope services**	Enters system services mode.
Step 3	UCS-A /system/services # **disable https**	Disables the HTTPS service.
Step 4	UCS-A /system/services # **commit-buffer**	Commits the transaction to the system configuration.

The following example disables HTTPS and commits the transaction:

```
UCS-A# scope system
UCS-A /system # scope services
UCS-A /system/services # disable https
UCS-A /system/services* # commit-buffer
UCS-A /system/services #
```

Enabling HTTP Redirection

Before You Begin

Enable both HTTP and HTTPS.

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope system**	Enters system mode.
Step 2	UCS-A /system # **scope services**	Enters system services mode.
Step 3	UCS-A /system/services # **enable http-redirect**	Enables the HTTP redirect service. If enabled, all attempts to communicate via HTTP are redirected to the equivalent HTTPS address. This option effectively disables HTTP access to this Cisco UCS domain.
Step 4	UCS-A /system/services # **commit-buffer**	Commits the transaction to the system configuration.

The following example enables HTTP to HTTPS redirection and commits the transaction:

```
UCS-A# scope system
UCS-A /system # scope services
UCS-A /system/services # enable http-redirect
Warning: When committed, this closes all the web sessions.
UCS-A /system/services* # commit-buffer
UCS-A /system/services #
```

Configuring SNMP

Information about SNMP

The Simple Network Management Protocol (SNMP) is an application-layer protocol that provides a message format for communication between SNMP managers and agents. SNMP provides a standardized framework and a common language used for the monitoring and management of devices in a network.

SNMP Functional Overview

The SNMP framework consists of three parts:

- An SNMP manager—The system used to control and monitor the activities of network devices using SNMP.
- An SNMP agent—The software component within Cisco UCS, the managed device, that maintains the data for Cisco UCS and reports the data, as needed, to the SNMP manager. Cisco UCS includes the agent and a collection of MIBs. To enable the SNMP agent and create the relationship between the manager and agent, enable and configure SNMP in Cisco UCS Manager.
- A managed information base (MIB)—The collection of managed objects on the SNMP agent. Cisco UCS release 1.4(1) and higher support a larger number of MIBs than earlier releases.

Cisco UCS supports SNMPv1, SNMPv2c and SNMPv3. Both SNMPv1 and SNMPv2c use a community-based form of security. SNMP is defined in the following:

- RFC 3410 (http://tools.ietf.org/html/rfc3410)
- RFC 3411 (http://tools.ietf.org/html/rfc3411)
- RFC 3412 (http://tools.ietf.org/html/rfc3412)
- RFC 3413 (http://tools.ietf.org/html/rfc3413)
- RFC 3414 (http://tools.ietf.org/html/rfc3414)
- RFC 3415 (http://tools.ietf.org/html/rfc3415)
- RFC 3416 (http://tools.ietf.org/html/rfc3416)
- RFC 3417 (http://tools.ietf.org/html/rfc3417)
- RFC 3418 (http://tools.ietf.org/html/rfc3418)
- RFC 3584 (http://tools.ietf.org/html/rfc3584)

SNMP Notifications

A key feature of SNMP is the ability to generate notifications from an SNMP agent. These notifications do not require that requests be sent from the SNMP manager. Notifications can indicate improper user authentication, restarts, the closing of a connection, loss of connection to a neighbor router, or other significant events.

Cisco UCS Manager generates SNMP notifications as either traps or informs. Traps are less reliable than informs because the SNMP manager does not send any acknowledgment when it receives a trap, and Cisco UCS Manager cannot determine if the trap was received. An SNMP manager that receives an inform request acknowledges the message with an SNMP response protocol data unit (PDU). If the Cisco UCS Manager does not receive the PDU, it can send the inform request again.

SNMP Security Levels and Privileges

SNMPv1, SNMPv2c, and SNMPv3 each represent a different security model. The security model combines with the selected security level to determine the security mechanism applied when the SNMP message is processed.

The security level determines the privileges required to view the message associated with an SNMP trap. The privilege level determines whether the message needs to be protected from disclosure or authenticated. The supported security level depends upon which security model is implemented. SNMP security levels support one or more of the following privileges:

- noAuthNoPriv—No authentication or encryption
- authNoPriv—Authentication but no encryption
- authPriv—Authentication and encryption

SNMPv3 provides for both security models and security levels. A security model is an authentication strategy that is set up for a user and the role in which the user resides. A security level is the permitted level of security within a security model. A combination of a security model and a security level determines which security mechanism is employed when handling an SNMP packet.

Supported Combinations of SNMP Security Models and Levels

The following table identifies what the combinations of security models and levels mean.

Table 1: SNMP Security Models and Levels

Model	Level	Authentication	Encryption	What Happens
v1	noAuthNoPriv	Community string	No	Uses a community string match for authentication.
v2c	noAuthNoPriv	Community string	No	Uses a community string match for authentication.
v3	noAuthNoPriv	Username	No	Uses a username match for authentication.
v3	authNoPriv	HMAC-MD5 or HMAC-SHA	No	Provides authentication based on the Hash-Based Message Authentication Code (HMAC) Message Digest 5 (MD5) algorithm or the HMAC Secure Hash Algorithm (SHA).
v3	authPriv	HMAC-MD5 or HMAC-SHA	DES	Provides authentication based on the HMAC-MD5 or HMAC-SHA algorithms. Provides Data Encryption Standard (DES) 56-bit encryption in addition to authentication based on the Cipher Block Chaining (CBC) DES (DES-56) standard.

SNMPv3 Security Features

SNMPv3 provides secure access to devices by a combination of authenticating and encrypting frames over the network. SNMPv3 authorizes management operations only by configured users and encrypts SNMP messages. The SNMPv3 User-Based Security Model (USM) refers to SNMP message-level security and offers the following services:

- Message integrity—Ensures that messages have not been altered or destroyed in an unauthorized manner and that data sequences have not been altered to an extent greater than can occur non-maliciously.
- Message origin authentication—Ensures that the claimed identity of the user on whose behalf received data was originated is confirmed.
- Message confidentiality and encryption—Ensures that information is not made available or disclosed to unauthorized individuals, entities, or processes.

SNMP Support in Cisco UCS

Cisco UCS provides the following support for SNMP:

Support for MIBs

Cisco UCS supports read-only access to MIBs.

For information about the specific MIBs available for Cisco UCS and where you can obtain them, see the http://www.cisco.com/en/US/docs/unified_computing/ucs/sw/mib/b-series/b_UCS_MIBRef.html for B-series servers, and http://www.cisco.com/en/US/docs/unified_computing/ucs/sw/mib/c-series/b_UCS_Standalone_C-Series_MIBRef.html C-series servers.

Authentication Protocols for SNMPv3 Users

Cisco UCS supports the following authentication protocols for SNMPv3 users:

- HMAC-MD5-96 (MD5)
- HMAC-SHA-96 (SHA)

AES Privacy Protocol for SNMPv3 Users

Cisco UCS uses Advanced Encryption Standard (AES) as one of the privacy protocols for SNMPv3 message encryption and conforms with RFC 3826.

The privacy password, or priv option, offers a choice of DES or 128-bit AES encryption for SNMP security encryption. If you enable AES-128 configuration and include a privacy password for an SNMPv3 user, Cisco UCS Manager uses the privacy password to generate a 128-bit AES key. The AES privacy password can have a minimum of eight characters. If the passphrases are specified in clear text, you can specify a maximum of 64 characters.

Enabling SNMP and Configuring SNMP Properties

SNMP messages from a Cisco UCS domain display the fabric interconnect name rather than the system name.

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope monitoring**	Enters monitoring mode.
Step 2	UCS-A /monitoring # **enable snmp**	Enables SNMP.
Step 3	UCS-A /monitoring # **set snmp community**	Enters snmp community mode.
Step 4	UCS-A /monitoring # **Enter a snmp community:** *community-name*	Specifies SNMP community. Use the community name as a password. The community name can be any alphanumeric string up to 32 characters.
Step 5	UCS-A /monitoring # **set snmp syscontact** *system-contact-name*	Specifies the system contact person responsible for the SNMP. The system contact name can be any alphanumeric string up to 255 characters, such as an email address or name and telephone number.
Step 6	UCS-A /monitoring # **set snmp syslocation** *system-location-name*	Specifies the location of the host on which the SNMP agent (server) runs. The system location name can be any alphanumeric string up to 512 characters.
Step 7	UCS-A /monitoring # **commit-buffer**	Commits the transaction to the system configuration.

The following example enables SNMP, configures an SNMP community named SnmpCommSystem2, configures a system contact named contactperson, configures a contact location named systemlocation, and commits the transaction:

```
UCS-A# scope monitoring
UCS-A /monitoring # enable snmp
UCS-A /monitoring* # set snmp community
UCS-A /monitoring* # Enter a snmp community: SnmpCommSystem2
UCS-A /monitoring* # set snmp syscontact contactperson1
UCS-A /monitoring* # set snmp syslocation systemlocation
UCS-A /monitoring* # commit-buffer
UCS-A /monitoring #
```

What to Do Next

Create SNMP traps and users.

Creating an SNMP Trap

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope monitoring**	Enters monitoring mode.
Step 2	UCS-A /monitoring # **enable snmp**	Enables SNMP.

	Command or Action	Purpose
Step 3	UCS-A /monitoring # **create snmp-trap** {*hostname* | *ip-addr* | *ip6-addr*}	Creates an SNMP trap host with the specified host name, IPv4 address, or IPv6 address. The host name can be a fully qualified domain name of an IPv4 address.
Step 4	UCS-A /monitoring/snmp-trap # **set community** *community-name*	Specifies the SNMP community name to be used for the SNMP trap.
Step 5	UCS-A /monitoring/snmp-trap # **set port** *port-num*	Specifies the port to be used for the SNMP trap.
Step 6	UCS-A /monitoring/snmp-trap # **set version** {**v1** | **v2c** | **v3**}	Specifies the SNMP version and model used for the trap.
Step 7	UCS-A /monitoring/snmp-trap # **set notification type** {**traps** | **informs**}	(Optional) The type of trap to send. This can be: • **traps** if you select v2c or v3 for the version. • **informs** if you select v2c for the version. **Note** An inform notification can be send only if you select v2c for the version.
Step 8	UCS-A /monitoring/snmp-trap # **set v3 privilege** {**auth** | **noauth** | **priv**}	(Optional) If you select v3 for the version, the privilege associated with the trap. This can be: • **auth—**Authentication but no encryption • **noauth—**No authentication or encryption • **priv—**Authentication and encryption
Step 9	UCS-A /monitoring/snmp-trap # **commit-buffer**	Commits the transaction to the system configuration.

The following example enables SNMP, creates an SNMP trap using an IPv4 address, specifies that the trap will use the SnmpCommSystem2 community on port 2, sets the version to v3, sets the notification type to traps, sets the v3 privilege to priv, and commits the transaction:

```
UCS-A# scope monitoring
UCS-A /monitoring # enable snmp
UCS-A /monitoring* # create snmp-trap 192.168.100.112
UCS-A /monitoring/snmp-trap* # set community SnmpCommSystem2
UCS-A /monitoring/snmp-trap* # set port 2
UCS-A /monitoring/snmp-trap* # set version v3
UCS-A /monitoring/snmp-trap* # set notificationtype traps
UCS-A /monitoring/snmp-trap* # set v3 privilege priv
UCS-A /monitoring/snmp-trap* # commit-buffer
UCS-A /monitoring/snmp-trap #
```

The following example enables SNMP, creates an SNMP trap using an IPv6 address, specifies that the trap will use the SnmpCommSystem3 community on port 2, sets the version to v3, sets the notification type to traps, sets the v3 privilege to priv, and commits the transaction:

```
UCS-A# scope monitoring
UCS-A /monitoring # enable snmp
UCS-A /monitoring* # create snmp-trap  2001::1
UCS-A /monitoring/snmp-trap* # set community SnmpCommSystem3
UCS-A /monitoring/snmp-trap* # set port 2
UCS-A /monitoring/snmp-trap* # set version v3
UCS-A /monitoring/snmp-trap* # set notificationtype traps
UCS-A /monitoring/snmp-trap* # set v3 privilege priv
UCS-A /monitoring/snmp-trap* # commit-buffer
UCS-A /monitoring/snmp-trap #
```

Deleting an SNMP Trap

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope monitoring**	Enters monitoring mode.
Step 2	UCS-A /monitoring # **delete snmp-trap** {*hostname* | *ip-addr*}	Deletes the specified SNMP trap host with the specified hostname or IP address.
Step 3	UCS-A /monitoring # **commit-buffer**	Commits the transaction to the system configuration.

The following example deletes the SNMP trap at IP address 192.168.100.112 and commits the transaction:

```
UCS-A# scope monitoring
UCS-A /monitoring # delete snmp-trap 192.168.100.112
UCS-A /monitoring* # commit-buffer
UCS-A /monitoring #
```

Creating an SNMPv3 User

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope monitoring**	Enters monitoring mode.
Step 2	UCS-A /monitoring # **enable snmp**	Enables SNMP.
Step 3	UCS-A /monitoring # **create snmp-user** *user-name*	Creates the specified SNMPv3 user. An SNMP username cannot be the same as a local username. Choose an SNMP username that does not match a local username.

	Command or Action	Purpose
Step 4	UCS-A /monitoring/snmp-user # **set aes-128** {**no** | **yes**}	Enables or disables the use of AES-128 encryption.
Step 5	UCS-A /monitoring/snmp-user # **set auth** {**md5** | **sha**}	Specifies the use of MD5 or DHA authentication.
Step 6	UCS-A /monitoring/snmp-user # **set password**	Specifies the user password. After you enter the **set password** command, you are prompted to enter and confirm the password.
Step 7	UCS-A /monitoring/snmp-user # **set priv-password**	Specifies the user privacy password. After you enter the **set priv-password** command, you are prompted to enter and confirm the privacy password.
Step 8	UCS-A /monitoring/snmp-user # **commit-buffer**	Commits the transaction to the system configuration.

The following example enables SNMP, creates an SNMPv3 user named snmp-user14, disables AES-128 encryption, specifies the use of MD5 authentication, sets the password and privacy password, and commits the transaction:

```
UCS-A# scope monitoring
UCS-A /monitoring # enable snmp
UCS-A /monitoring* # create snmp-user snmp-user14
UCS-A /monitoring/snmp-user* # set aes-128 no
UCS-A /monitoring/snmp-user* # set auth md5
UCS-A /monitoring/snmp-user* # set password
Enter a password:
Confirm the password:
UCS-A /monitoring/snmp-user* # set priv-password
Enter a password:
Confirm the password:
UCS-A /monitoring/snmp-user* # commit-buffer
UCS-A /monitoring/snmp-user #
```

Deleting an SNMPv3 User

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope monitoring**	Enters monitoring mode.
Step 2	UCS-A /monitoring # **delete snmp-user** *user-name*	Deletes the specified SNMPv3 user.
Step 3	UCS-A /monitoring # **commit-buffer**	Commits the transaction to the system configuration.

The following example deletes the SNMPv3 user named snmp-user14 and commits the transaction:

```
UCS-A# scope monitoring
UCS-A /monitoring # delete snmp-user snmp-user14
UCS-A /monitoring* # commit-buffer
UCS-A /monitoring #
```

Enabling Telnet

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope system**	Enters system mode.
Step 2	UCS-A /system # **scope services**	Enters system services mode.
Step 3	UCS-A /services # **enable telnet-server**	Enables the Telnet service.
Step 4	UCS-A /services # **commit-buffer**	Commits the transaction to the system configuration.

The following example enables Telnet and commits the transaction:

```
UCS-A# scope system
UCS-A /system # scope services
UCS-A /services # enable telnet-server
UCS-A /services* # commit-buffer
UCS-A /services #
```

Enabling the CIMC Web Service

To enable the CIMC Web Service:

- You must be logged in with admin privileges.
- The CIMC web service must be disabled, as it is enabled by default.

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope system** /	Enters the system mode.
Step 2	UCS-A /system #**scope services**/	Enters the services mode for the system.
Step 3	UCS-A/system/services #**enable cimcwebsvc**/	Enable the CIMC web service.
Step 4	UCS-A/system/services *# **commit-buffer**/	Commits the transaction to the system configuration.

The following example shows how to enable the CIMC web service and save the transaction:

```
UCS-A# scope system
UCS-A/system # scope services
UCS-A/system/services # enable cimcwebsvc
UCS-A/system/services *# commit-buffer
UCS-A/system/services # commit-buffer
UCS-A/system/services # show cimcwebsvc
Name: cimcwebservice
    Admin State: Enabled
```

Disabling the CIMC Web Service

To disable the CIMC Web Service:

- You must be logged in with admin privileges.
- The CIMC web service must be enabled.

Note The CIMC web service is enabled by default.

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope system** /	Enters the system mode.
Step 2	UCS-A /system #**scope services**/	Enters the services mode for the system.
Step 3	UCS-A/system/services #**disable cimcwebsvc**/	Disables the CIMC web service.
Step 4	UCS-A/system/services *# **commit-buffer**/	Commits the transaction to the system configuration.

The following example shows how to disable the CIMC web service and save the transaction:

```
UCS-A# scope system
UCS-A/system # scope services
UCS-A/system/services # disable cimcwebsvc
UCS-A/system/services *# commit-buffer
UCS-A/system/services # commit-buffer
UCS-A/system/services # show cimcwebsvc
Name: cimcwebservice
    Admin State: Disabled
```

Disabling Communication Services

Procedure

	Command or Action	Purpose
Step 1	UCS-A# **scope system**	Enters system mode.
Step 2	UCS-A /system # **scope services**	Enters system services mode.
Step 3	UCS-A /system/services # **disable** *service-name*	Disables the specified service, where the *service-name* argument is one of the following keywords: • *cimxml* **—**Disables CIM XML service • **http —**Disables HTTP service • **https —**Disables HTTPS service • **telnet-server —**Disables Telnet service
Step 4	UCS-A /system/services # **commit-buffer**	Commits the transaction to the system configuration.

The following example disables CIM XML and commits the transaction:

```
UCS-A# scope system
UCS-A# scope services
UCS-A /system/services # disable cimxml
UCS-A /system/services* # commit-buffer
UCS-A /system/services #
```