Network Security

Network Communications between Nodes within the Cluster

The following details are all based on the default settings. These can vary depending on the application setup and network design (such as NAT) of the solution, so may need adjustment accordingly. Where a dependant is noted, this is fully dependant on the configuration with no default.

These communications are all related to communications between application nodes within the cluster. There are a few different deployment models so the details below cover the different models and relevant ports. So review and implement according to the deployment model in use.

Note Standalone is only a single node so this section is not relevant for that deployment model.

Proxy to Proxy Node

This is relevant if the proxy node is present in the system.

Communication	Protocol	Port
Cluster Communications	HTTPS	TCP 8443 bi-directional

Proxy to Unified/Application Node

This is relevant if the proxy node is present in the system.

Communication	Protocol	Port
User access	HTTPS	TCP 443
Cluster Communications	HTTPS	TCP 8443 bi-directional

Unified Node to Unified node

This is relevant to the communications between the unified nodes (application and database combined). If the application and database nodes are split, then see the relevant application and database node details below. Database arbiters run on port 27030.

Communication	Protocol	Port
Database access	database	TCP 27017 and 27030 bi-directional
Cluster Communications	HTTPS	TCP 8443

Application node to Application node

This is relevant to the communications between application nodes in the system. This is only relevant where the database node is separate from the application node (in other words, not Unified node).

Communication	Protocol	Port
Cluster communications	HTTPS	TCP 8443 bi-directional

Application Node to Database node

This is relevant to the communications between the application node and the database node. This is relevant if the database node is separate from the application node. Database arbiters run on port 27030.

Communication	Protocol	Port
Database access	database	TCP 27017 and 27030 bi-directional
Cluster Communications	HTTPS	TCP 8443

Database Node to Database node

This is relevant to the communications between the application node and the database node. This is relevant if the database node is separate from the application node. Database arbiters run on port 27030.

Communication	Protocol	Port
Database access	database	TCP 27017 and 27030 bi-directional

Communication	Protocol	Port
Cluster Communications	HTTPS	TCP 8443

Network Communications External to the Cluster

The following details are all based on the default settings. These can vary depending on the application setup and network design (such as NAT) of the solution, so may need adjustment accordingly. Where a dependant is noted, this is fully dependant on the configuration with no default.

These communications are all related to communications with devices external to the cluster.

Outbound Communications to Devices from the Application/Unified nodes:

Communication	Protocol	Port
Cisco Unified Communications Manager (UCM)	HTTPS	TCP 8443
Cisco Unity Connection (CUXN)	HTTPS	TCP 8443
Webex	HTTPS	TCP 443
LDAP directory	LDAP	TCP/UDP 389 and/or 636(TLS/SSL)
Single Sign-on (SSO)	HTTPS	TCP 443
Cisco HCM-F	HTTPS	TCP 8443

Outbound to external systems from the proxy node:

Communication	Protocol	Network Protocol and Port
API Sync and Async responses	HTTPS	TCP 443
Northbound Notification messages	HTTPS	dependant

Outbound to external systems from all nodes:

Communication	Protocol	Port
SNMP	SNMP	TCP/UDP 162
SFTP as required for backup destinations	SFTP	TCP 22
NTP	NTP	UDP 123

Inbound communications from external systems to the proxy node:

Communication	Protocol	Port
Web Access	HTTPS	TCP 443
API Request	HTTPS	TCP 443

Inbound communications to all nodes:

Communication	Protocol	Port
SSH and SFTP for management and files transfers	SFTP/SSH	TCP/UDP 22

Dynamic Firewall

The most important part of the network security model is the system firewall.

The platform uses a dynamic firewall which does not open a fixed set of ports but adapts to the applications installed, only allowing such traffic as the specific set of running services require.

If an application is stopped, it's ports are automatically closed. This creates a default-blacklist firewall which pinholes only those ports required for the operation of the specific setup in use.

The firewall is one of the very first services the platform brings up and among the very last it shuts down in order maximize the network security.

Where possible, the firewall will also rate limit connections to services to prevent abuse (see the section: Prevention of DOS attacks for more details).

Service and Ports list

The following external network ports are in use and need to be opened on the firewall for communication between cluster nodes:

Node type	Ports
WebProxy	22 (ssh & sftp), 80 (http), 161 & 162 (snmp), 443 & 8443 (https)
Application	22 (ssh & sftp), 80 (http), 161 & 162 (snmp), 443 & 8443 (https), 27017 & 27030 (database)
Database	22 (ssh & sftp), 161 & 162(snmp), 27019 & 27020 (database)

Additionally, the Application node interacts with external Cisco equipment (e.g. UCM, CUCx) and will require additional firewall ports to be opened.

Web Certificates

The platform installs a self-signed certificate for the web-frontend by default. This provides encryption of the web-traffic but does not provide users with valid authentication that the server is correct or protect against man-in-the-middle attacks.

For this reason we strongly advise customers to obtain a trusted CA-signed certificate and install it on the server. Once a signed, trusted certificate is obtained (this should be a single-file concatenated certificate suitable for the NginX server) copy it to the platform using scp and then install the file into the server using:

web cert add <filename>

Only one certificate file can be installed on the platform. For more details on NginX compatible certificates see the relevant nginx documentation here: http://wiki.nginx.org/HttpSslModule

Please note the importance of ensuring that SSL certificates generated match the assigned network name of the platform.

Web Certificate Expiration Notice

If a Web Certificate is due to expire, a notice will display on the status display 30 days before the expiration:

```
platform@development:~$ help

host: AS01, role: webproxy,application,database, LOAD: 3.85
date: 2014-08-28 11:24:22 +00:00, up: 6 days, 3:03
network: 172.29.42.100, ntp: 196.26.5.10
HEALTH: NOT MONITORED
database: 20Gb
application: up
WEB CERT EXPIRES AT: 2014-09-26 11:30:02

mail - local mail management       keys - ssh/sftp credentials
network - network management       backup - manage backups
voss - voss management tools       log - manage system logs
database - database management     notify - notifications control
schedule - scheduling commands     diag - system diagnostic tools
system - system administration     snmp - snmp configuration
user - manage users                cluster - cluster management
drives - manage disk drives        web - web server management
app - manage applications          template - template pack creator
```

If a Web Certificate has expired, the notice on the status displays:

```
WEB CERT EXPIRED AT: 2014-09-26 11:30:02
```

Once the certificate is expired, the system can be used as normal, but the certificate will be expired and for non self-signed certificates (like a Godaddy or Thawte certificates), the data will no longer be properly encrypted.

Set Up a Web Certificate

Procedure

Step 1 Run **web cert details** if needed to edit the details displayed from the server.

Step 2 Run **web cert gen_csr** to generate the Certification Request (CSR).

Step 3 Send the file to a Certificate Authority (CA).

Step 4 Upload the reply from the CA to the server using **scp**.

Step 5 Run **web cert add <filename of uploaded file>**.

Web Certificate Commands

The following Command Line Interface console display shows the available commands for web certificates.

- **web cert add <filename>** - Install the certificate from <filename> into the web server.
- **web cert del** - Revert to a self-signed certificate
- **web cert details** - Print the certicate details in config system.
- **web cert details edit** - Update the certicate details in config system.
- **web cert gen_csr** - Create a CSR file in /opt/platform/admin/home/media.
- **web cert gen_selfsigned** - Generate a self-signed certificate.
- **web cert print_csr** - Create a CSR file in /opt/platform/admin/home/media.
- **web cluster prepnode** - Prepares the system so that it can be joined to a cluster as a web proxy.
- **web sslv3 <on/off>** - Enable/disable SSLv3 on the system.
- **web weight add <server:port> <weight>** - Modify the weights of an upstream service. Higher weights will serve more requests, while 0 will only be used if no other servers are available.
- **web weight del <server:port>** - Delete the user-defined service weight and use system defaults.
- **web weight list** - Display the weights of upstream services

Network URI specification

All network locations are specified as a URI, for example download locations, backup destinations, notification destinations, and so on.

The following list shows the URI syntax:

- ftp: `ftp://user[:password]@host[:port][/path]`
- http: `http(s)://user[:password]@host[:port]/path`

- file: `file://{/path}+[/filename]`
- sftp: `sftp://user[:password]@host[:port][/path]`
- scp: `scp://[user@]host[:port]:[/path]`
- Email: `mailto:user@host`
- snmpv2: `snmp://community@host[:port]`
- snmpv3: `snmp://user:auth:password]@host[:port] ... minimum auth/password`