



## DCNM for SAN Web Services API

---

This chapter describes the DCNM for SAN (DCNM-SAN) Web Services application program interface (API). This chapter defines the APIs exposed by the Cisco DCNM for SAN Web Services feature.

This chapter includes the following sections:

- [Web Services Specifications, page 8-1](#)
- [Logon Service, page 8-2](#)
- [San WS, page 8-4](#)
- [Zone Manager WS - SEI, page 8-24](#)
- [Statistics WS, page 8-33](#)
- [Security WS, page 8-36](#)
- [Protocol WS, page 8-44](#)
- [Event WS - SEI, page 8-51](#)
- [Cluster WS - SEI, page 8-49](#)
- [Inventory WS - SEI, page 8-53](#)
- [Error Codes, page 8-59](#)

## Web Services Specifications

Web Services specifications combine together to provide interoperable protocols for security, communication, and syntax for representing data.

- [XML, page 8-1](#)
- [SOAP, page 8-2](#)
- [HTTP/HTTPS, page 8-2](#)
- [WDSL, page 8-2](#)

## XML

XML is the data format that defines the structure of the message. XML Web Services architecture allows programs written in different languages on different platforms to communicate with each other in a standards-based way. XML Web Services expose useful functionality to Web users through a standard Web protocol (SOAP).

## SOAP

Simple Object Access Protocol (SOAP) is the communications protocol for Web Services. SOAP is a specification that defines the XML format for messages. The advantage of SOAP is that it has been implemented on many different hardware and software platforms.

## HTTP/HTTPS

HTTP/HTTPS is the transport layer of the service. HTTP/HTTPS allows data to traverse the network easily and is widely accepted. It is also considered as platform neutral. Every Cisco DCNM-SAN Web Services operation is through HTTP/HTTPS.

## WSDL

A WSDL definition is an XML document with a root definition element from the `http://schemas.xmlsoap.org/wsdl/` namespace. Cisco DCNM-SAN Web Services uses the WSDL document to publish which operations of DCNM-SAN are available. The definitions element can contain several other elements including types, message, portType, binding, and service, all of which come from the namespace. WSDL is published on FMServer at:

`http://localhost/LogonWSService/LogonWS?wsdl`

## Logon Service

LogonWS makes IdentityManager's operations available as Web Service calls. LogonWS allows the following operations:

- [requestToken](#), page 8-2
- [validateToken](#), page 8-4

## requestToken

This method returns a token string that must be passed in as the header of the SOAP message. Once the username and password is authenticated using DCNM-SAN's SecurityManager, the token is generated and is kept valid for the number of milliseconds specified in the expiration argument.

### Parameters

username—Name of the user.

password—Password of the user.

expiration—Time (in milliseconds).

### Return Value

Session token.

### Error

Error code: 201—Invalid argument in Web Service exception.

## requestLogonRole

**Parameters**

username—Name of the user.  
password—Password of the user.  
expiration—Time (in milliseconds).

**Return Value**

Session token.

**Error**

Error code: 201—Invalid argument in Web Service exception.

## requestLogonToken

**Parameters**

username—Name of the user.  
password—Password of the user.  
expiration—Time (in milliseconds).

**Return Value**

Session token.

**Error**

Error code: 201—Invalid argument in Web Service exception.

## getCredentialByToken

**Parameters**

username—Name of the user.  
password—Password of the user.  
expiration—Time (in milliseconds).

**Return Value**

Session token.

**Error**

Error code: 201—Invalid argument in Web Service exception.

## validateToken

This method returns true or false depending on the validity of the token. If the token has expired, it returns false, or else it returns true.

### Parameters

token—Session Token.

### Return Value

Boolean value is True if DCNM-SAN accepts the token.

### Error

Error code: 201—Invalid argument in Web Service exception.

## Authentication or Token

To interact with DCNM-SAN Web Services, you must obtain a token through LogonWS and attach this token to the header message of every SOAP requests. DCNM-SAN Web Services verifies user credentials using a unique token string that is administered by LogonWS. At any given time, HTTPS should be deployed to secure the communication channel. The following example displays the format of the header message:

```
<?xml version="1.0" encoding="UTF-8" ?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Header xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
    <m:Token xmlns:m="http://www.w3schools.com/transaction/">
      token string is put here
    </m:Token></SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <getFabrics xmlns="http://ep.jaxws.dcbu.cisco.com/" />
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

## IdentityManager

IdentityManager provides identity services and manages the user credentials that are required by Web Services. It is the token provider that administers and maintains tokens. It authenticates the user, generates tokens, and validates or expires tokens by periodically checking and clearing the cache.

## San WS

San Service is an Enterprise Java Beans (EJB) component that manages SAN-related service requests and executes queries on DCNM-SAN for information. San WS checks with IdentityManager for authentication before performing the request. A valid token string indicates to San Service that the user

is a DCNM-SAN user and that it must honor and execute the request. After retrieving the required information, it sends the result back to the user. SanWS logs errors in `fms_ws.log`. The service end-point interface (SEI) of SanWS defines the operations of the service, and sends them to the end users.

## getFabrics

Returns the list of all open fabrics.

### Return Value

An array of open fabrics.

### Error

Error Code: 300— General SAN Service exception.

## getFabricByIP

Returns the list of fabrics associated with the IP address of a given switch.

### Parameters

`ipAddress`—IP address of the switch.

### Return Value

List of all fabrics associated with the specific IP address.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getPmEntity

Returns the PM entity from the database.

### Return Value

PM entity from the database.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getPmChartData

Returns the PM statistics of a specific RRD file.

### Return Value

PM statistics of a specific RRD file.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getFabricByKey

Returns the list of fabrics associated with the specified key.

**Parameters**

key—Key of the fabric.

**Return Value**

List of all fabrics associated with the specified key.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getFabricBySwitchKey

Returns the list of fabrics associated with the specified seed switch key (WWN).

**Parameters**

swkey—Seed switch key of the fabric.

**Return Value**

List of all fabrics associated with the specified seed switch key.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getSwitchesByFabric

Returns the list of switches associated with the specified fabric key.

**Parameters**

key—Key of the fabric.

**Return Value**

List of all fabrics associated with the specified fabric key.

**Error**

Error Code: 300— General SAN Service exception.

## getNeighborSwitches

Returns the list of neighboring switches associated with the specified WwnKey.

### Parameters

key—WwnKey object.

### Return Value

List of neighboring switches associated with the specified WwnKey.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

Error Code: 302—SAN does not find objects by query key exception.

## getActiveServerNodes

Returns the list of all active DCNM for SAN servers.

### Parameters

key—Key of the fabric.

### Return Value

List of all fabric servers that are active.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getFabricWithSnmpCredential

Returns the list of fabrics (except the fabric with opening status) with the SNMP credentials.

### Parameters

key—Key of the fabric.

### Return Value

List of all fabrics with their SNMP credentials.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getSwitchesByFabric

Returns the list of switches associated with the specified fabric key.

### Parameters

key—Key of the fabric.

### Return Value

List of all fabrics associated with the specified fabric key.

### Error

Error Code: 300— General SAN Service exception.

## getSwitch

Returns the list of switches on all the fabrics.

### Parameters

key—Key of the fabric.

### Return Value

List of all fabrics associated with the specified fabric key.

### Error

Error Code: 300— General SAN Service exception.

## getSwitchByKey

Returns the switch associated with the specified switch key object.

### Parameters

key—Key of the fabric.

### Return Value

Switch associated with the specified switch key.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getSwitchIPByName

Returns the IP address associated with the specified system name or switch name.



**Parameters**

sysname—Name of the system or switch.

**Return Value**

IP address associated with the specified system name.

**Error**

Error Code: 300—General SAN Service exception.

Error code: 201—Invalid argument in Web Service exception.

## getSwitchIPByKey

Returns the IP address of the switch associated with the specified WwnKey object.

**Parameters**

key—WwnKey object.

**Return Value**

IP address associated with the specified WwnKey object.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getNeighborSwitches

Returns the list of neighboring switches associated with the specified WwnKey.

**Parameters**

key—WwnKey object.

**Return Value**

List of neighboring switches associated with the specified WwnKey.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

Error Code: 302—SAN does not find objects by query key exception.

## getVsans

Returns the list of VSANs in the fabric associated with the specified fabric key.

**Parameters**

key—Fabric key object.

**Return Value**

List of VSANs in the fabric associated with the specified fabric key.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getVsan

Returns the VSAN in the fabric associated with the specified VSAN key object.

**Parameters**

key—VSAN key object.

**Return Value**

VSANs in the fabric associated with the specified VSAN key object.

**Error**

Error Code: 300— General SAN Service exception.

Error code: 201—Invalid argument in Web Service exception.

## getIsIs

Returns the list of ISLs in the VSAN associated with the specified VSAN key.

**Parameters**

key—VSAN key.

**Return Value**

Array of ISL objects in the VSAN associated with the specified VSAN key.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## discoverFabric

This API opens the fabric. This function requires the IP address of the seed switch and SNMP credentials.

**Parameters**

seed—IP address of the seed switch.

user—SNMP credential.

**Return Value**

Boolean value is True if the discovery was successful.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

Error Code: 100— Authentication failure exception.

Error Code: 101—Invalid credentials exception.

## manageFabric

Returns true or false depending on manageability of the fabric.

**Parameters**

key—Fabric key.

**Return Value**

Returns true if the fabric can be identified or managed. Returns false if the fabric cannot be identified or managed.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## unManageFabric

This function unmanages a fabric.

**Parameters**

key—Fabric key.

**Return Value**

None.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## closeFabric

This function unmanages and closes a fabric.

**Parameters**

key—Fabric key.

**Return Value**

None.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## purgeFabric

This function purges the specified fabric data both from the DCNM-SAN cache and database.

**Parameters**

key—Fabric key.

**Return Value**

None.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

Error Code: 302—SAN does not find objects by query key exception.

## getEndpoints

Returns the list of all the end ports known to DCNM-SAN.

**Return Value**

An array of all the end ports.

**Error**

Error Code: 300— General SAN Service exception.

## getEnclosures

Returns the list of all the enclosures known to DCNM-SAN.

**Return Value**

An array of enclosure objects.

**Error**

Error Code: 300— General SAN Service exception.

## getEndPointByKey

Returns the end port based on the switch WWN.

### Parameters

key—WWN of the node.

### Return Value

Returns the end port based on the switch WWN. Returns null if there are no end ports associated with the switch.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getEndPointAttachedToSw

Returns the end ports that are associated with a switch.

### Parameters

key—IP address of the switch.

### Return Value

Returns the end ports based on switch.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getEnclosureByName

Returns the enclosure based on the name.

### Parameters

name—Name of the enclosure object.

### Return Value

Returns the enclosure object.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getEnclosureByKey

Returns the enclosure based on the name.

### Parameters

name—Name of the enclosure object.

### Return Value

Returns the enclosure object.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getEnclosureByPwwn

Returns the enclosures that are associated with a physical WWN.

### Parameters

wwn—Physical WWN of the switch.

### Return Value

Returns the enclosure based on physical WWN.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## updateEnclosure

Update the enclosure with the value that is passed as parameter.

### Parameters

value—Value to update the enclosure.

### Return Value

None.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## updateEndpointEnclosure

Update the end port enclosure with the value that is passed as parameter.

**Parameters**

endportKey—Value for the end port key.

enclosureKey—Value for the enclosure key.

**Return Value**

None.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getHosts

Returns the list of all the host enclosures known to DCNM-SAN.

**Return Value**

Returns the list of all the host enclosures known to DCNM-SAN.

**Error**

Error Code: 300— General SAN Service exception.

## getHost

Returns the name of hosts in a VSAN.

**Parameters**

key—Name of the VSAN.

**Return Value**

Returns the name of the hosts in the specified VSAN.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getHostByFabric

Returns the name of hosts in a fabric.

ValidationException is displayed if any of the following situation occurs:

- If the argument passed is null.
- If the argument does not contain a valid key.

**Parameters**

key—Name of the fabric.

**Return Value**

Returns the name of the hosts in the specified VSAN.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getStorages

Returns the list of all the storage device enclosures known to DCNM-SAN.

**Return Value**

An array of all the storage device enclosures known to DCNM-SAN.

**Error**

Error Code: 300— General SAN Service exception.

## getStorageByFabric

Returns the name of storage device enclosures in a fabric.

**Parameters**

key—Name of the fabric.

**Return Value**

Returns the name of the storages in the specified fabric.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getHostPorts

Returns the list of all the host end ports in a fabric.

**Parameters**

key—Name of the fabric.

**Return Value**

An array of all the host ports in a fabric.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.



## getDomainId

Returns the domain address.

**Parameters**

key—Wwn

vsanid—Unique identifier of the VSAN.

**Return Value**

Domain IP address.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getVsanIp

Returns the IP address of a VSAN.

**Parameters**

key—Wwn

vsanid—Unique identifier of the VSAN.

**Return Value**

IP address of a VSAN.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getVsanDomains

Returns all VSAN domains in a switch.

**Parameters**

Key—Wwnkey

**Return Value**

VSAN domains in a switch.

**Error**

Error Code: 300— General SAN Service exception.

Error code: 201—Invalid argument in Web Service exception.

## getlvrEnfZoneSetName

Returns the fabric IVR-enforced zone set name.

### Parameters

Key—Fabric key

### Return Value

Zone set name.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getlvrEnfZoneSetNumber

Returns the fabric IVR-enforced zone number.

### Parameters

Key—Fabric key

### Return Value

Zone number.

### Error

Error Code: 300— General SAN Service exception.

Error code: 201—Invalid argument in Web Service exception.

## getlvrEnfZoneSetActivateTime

Returns the fabric IVR-enforced zone set activate time.

### Parameters

Key—Fabric key

### Return Value

Time stamp in the long integer format.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getlvrEnfZoneSet

Returns the fabric IVR-enforced zone set.

### Parameters

Key—Fabric key

**Return Value**

List of zone objects.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getIvrActiveZonesetChecksum

Returns the IVR active zone set checksum.

**Parameters**

Key—Fabric key

**Return Value**

Checksum value.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getAliases

Returns all the aliases used by the fabric.

**Parameters**

Key—Fabric key

**Return Value**

Aliases used by the fabric.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## useFcAlias

Returns all the FC aliases used by the fabric.

**Parameters**

Key—Fabric key

**Return Value**

FC aliases used by the fabric.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getEnfZoneSet

Returns all the VSAN enforced zone sets.

### Parameters

Key—Fabric key

### Return Value

List of zones.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getEnfZoneSetName

Returns all the VSAN enforced zone set names.

### Parameters

Key—Fabric key

### Return Value

List of zone set names.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getEnfZoneSetName

Returns all the VSAN enforced zone set name.

### Parameters

Key—Fabric key

### Return Value

List of zone set names.

### Error

Error Code: 300— General SAN Service exception.

Error code: 201—Invalid argument in Web Service exception.

## getFCAliases

Returns all the FC aliases for the fabric.

### Parameters

Key—Fabric key

### Return Value

List of aliases.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getFCAliasesByVsan

Returns all the FC aliases for the VSAN.

### Parameters

Key—Fabric key

### Return Value

List of aliases for the VSAN.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getCFS

Returns CFS.

### Parameters

Key—Fabric key

### Return Value

List of CFS features.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getCFSBySwitch

Returns CFS.

### Parameters

Key—Switch key

**Return Value**

List of CFS features.

**Error**

Error Code: 300— General SAN Service exception.

Error code: 201—Invalid argument in Web Service exception.

## getZoneMode

Returns zone operation modes.

**Parameters**

Key—Fabric key

**Return Value**

List of zone operation modes.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getZoneModeByVsan

Returns zone operation modes for VSAN.

**Parameters**

Key—VSAN key

**Return Value**

List of zone operation modes.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getZoneAttributes

Returns zone attributes.

**Parameters**

Key—Fabric key

**Return Value**

Zone attributes.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getZoneAttributesByVsan

Returns zone attributes for VSAN.

### Parameters

Key—VSAN key

### Return Value

Zone attributes for VSAN.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getSwitchPorts

Returns ports for the switch.

### Parameters

Key—Fabric key

### Return Value

List of ports for a given switch.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## isIVREnabled

Returns a boolean value depending on whether the IVR is enabled on the switch or not.

### Parameters

Key—Switch key

### Return Value

Boolean value.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getSwitchDateAndTime

Returns the switch time and date.

**Parameters**

Key—Switch key

**Return Value**

Boolean value.

**Error**

Error Code: 400—SnmpException.

## Zone Manager WS - SEI

### activateZoneset

Activates the zone set.

**Parameters**

key—WwnKey

key—Name of the VSAN

**Return Value**

Operational status of the zone set.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

### addZone

Add a new zone to the list of zones.

**Parameters**

key—WwnKey

key—Name of the VSAN

**Return Value**

Zone object.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

### addZoneAlias

Adds a zone alias.

**Parameters**



key—WwnKey

key—Name of the VSAN

**Return Value**

Zone object.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## addZoneMemberToZone

Adds a new zone member to the specific zone.

**Parameters**

key—WwnKey

key—Name of the VSAN

**Return Value**

ID of the zone member.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## addZoneMemberToZoneAlias

Adds a zone member to the zone alias.

**Parameters**

key—WwnKey

key—Name of the VSAN

**Return Value**

Zone member.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## createZone

Creates a new zone.

**Parameters**

key—WwnKey

key—Name of the VSAN

**Return Value**

Zone object.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## createZoneAlias

Creates a zone alias.

**Parameters**

key—WwnKey

key—Name of the VSAN

**Return Value**

Zone alias object.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## createZoneMemberInZone

Creates a zone member in the specified zone.

**Parameters**

key—WwnKey

key—Name of the VSAN

**Return Value**

Member ID.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## createZoneMemberInZoneAlias

Creates a zone member in the specified zone alias.

**Parameters**

key—WwnKey

key—Name of the VSAN

**Return Value**

Member ID.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## createZoneSet

Creates a zone set.

**Parameters**

key—WwnKey

key—Name of the VSAN

**Return Value**

ID of the zone set.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## deActivateZoneset

Returns the operational status of the zone set.

**Parameters**

key—WwnKey

key—Name of the VSAN

**Return Value**

Operational status of the zone set.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getEnfZoneSet

Returns the enforced zone set.

**Parameters**

key—WwnKey.

key—Name of the VSAN.

**Return Value**

ID of the zone set.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getEnfZoneSetName

Returns the name of the enforced zone set.

### Parameters

key—WwnKey

key—Name of the VSAN

### Return Value

ID of the zone set.

### Error

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getIvrActiveZonesetChecksum

Returns the IVR active zoneset checksum.

### Parameters

key—FabricKey

### Return Value

Checksum value.

### Error

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getIvrEnfZoneNumber

Returns the fabric IVR enforced zone number.

### Parameters

key—FabricKey

### Return Value

Zone number.

### Error

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getlvrEnfZoneSet

Returns the fabric IVR enforced zone set ID.

### Parameters

key—FabricKey

### Return Value

Zone set ID.

### Error

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getlvrEnfZoneSetActivateTime

Returns the fabric IVR enforced zone set activate time.

### Parameters

key—FabricKey

### Return Value

System time as the number of seconds elapsed since the start of the Unix epoch at 1 January 1970 00:00:00 UT.

### Error

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getlvrEnfZoneSetName

Returns the fabric IVR enforced zone set name.

### Parameters

key—WWNKey

### Return Value

Zone ID.

### Error

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getZone

Returns an array of zone IDs.

### Parameters

key—WwnKey

**Return Value**

Zone object.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getZoneAlias

Returns zone alias.

**Parameters**

key—WwnKey

key—VSAN Key

**Return Value**

Zone alias object.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getZoneAliases

Returns an array of zone aliases.

**Parameters**

key—WwnKey

key—VSAN Key

**Return Value**

List of zone aliases.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getZoneAttributes

Returns all the attributes for the zone.

**Parameters**

key—WwnKey

key—VSAN Key

**Return Value**

List of zone attributes.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getZoneAttributesByVsan

Returns all the zone attributes for the VSAN.

**Parameters**

key—WwnKey

**Return Value**

List of zone attributes for the VSAN.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getZoneCapabilitiesByFabric

Returns the zone information associated with the specified fabric key.

**Parameters**

key—Fabric key

**Return Value**

ZoneCapabilities object.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getZoneCapabilitiesByVsan

Returns all zone information for the VSAN.

**Parameters**

key—WwnKey

**Return Value**

ZoneCapabilities object.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getZoneMode

Returns the list of zone modes.

**Parameters**

key—WwnKey

**Return Value**

Zone information.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getZoneModeByVsan

Returns the list of zone modes for the VSAN.

**Parameters**

key—VSAN key

**Return Value**

Zone information for the VSAN.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getZoneSet

Returns the zone set.

**Parameters**

key—WwnKey

**Return Value**

Zone sets.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getZoneSets

Returns the list of zone sets.

**Parameters**

key—WwnKey



**Return Value**

List of zone sets.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getZones

Returns the list of zones.

**Parameters**

key—WwnKey

**Return Value**

List of zones.

**Error**

Error Code: 300—General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## Statistics WS

### getEndDeviceStatistics

Returns the statistics of the end device.

**Parameters**

key—WwnKey

**Return Value**

Statistics of the end device.

**Error**

Error Code: 201—Invalid argument in Web Service exception.

### getEndDeviceStatisticsByAlias

Returns the statistics of the end devices by device alias.

**Parameters**

key—WwnKey

**Return Value**

Statistics of the end devices.

**Error**

Error Code: 201—Invalid argument in Web Service exception.

## getEthPortStatisticsByKey

Returns the statistics of the Ethernet port.

### Parameters

key—WwnKey

### Return Value

Statistics of the Ethernet port.

### Error

Error Code: 201—Invalid argument in Web Service exception.

## getEthPortStatisticsBySwitch

Returns the statistics of the Ethernet port based on a switch.

### Parameters

key—WwnKey

### Return Value

Statistics of the Ethernet port based on a switch.

### Error

Error Code: 201—Invalid argument in Web Service exception.

## getFcPortStatistics

Returns the statistics of the Fibre Channel port.

### Parameters

key—WwnKey

### Return Value

Statistics of the Fibre Channel port.

### Error

Error Code: 201—Invalid argument in Web Service exception.

## getFcPortStatisticsByKey

Returns the statistics of the Fibre Channel port based on the switch WWN.

**Parameters**

key—WwnKey

**Return Value**

Statistics of the Fibre Channel port.

**Error**

Error Code: 201—Invalid argument in Web Service exception.

## getFcPortStatisticsBySwitch

Returns the statistics of the Fibre Channel port based on a switch.

**Parameters**

key—WwnKey

**Return Value**

Port statistics of the Fibre Channel port.

**Error**

Error Code: 201—Invalid argument in Web Service exception.

## getIPEndPointStatisticsByKey

Returns the statistics of the IPEndPoint based on a switch WWN.

**Parameters**

key—WwnKey

**Return Value**

Statistics of the IPEndPoint.

**Error**

Error Code: 201—Invalid argument in Web Service exception.

## getIPEndPointStatisticsBySwitch

Returns the statistics of the IPEndPoint based on a switch.

**Parameters**

key—WwnKey

**Return Value**

Statistics of the IPEndPoint.

**Error**

Error Code: 201—Invalid argument in Web Service exception.

## getTCPEndPointStatisticsByKey

Returns the statistics of the TCPEndPoint based on a switch WWN.

**Parameters**

key—WwnKey

**Return Value**

Statistics of the TCPEndPoint.

**Error**

Error Code: 201—Invalid argument in Web Service exception.

## getTCPEndPointStatisticsBySwitch

Returns the statistics of the TCPEndPoint based on a switch.

**Parameters**

key—WwnKey

**Return Value**

Statistics of the TCPEndPoint on a switch.

**Error**

Error Code: 201—Invalid argument in Web Service exception.

# Security WS

## getAaaMaxServer

Returns a value for the maximum number of server entries in a server group of the AAA configuration.

**Parameters**

key—WwnKey

**Return Value**

Maximum number of server entries in a server group of the AAA configuration.

**Error**

Error Code: 400—SnmpException

## getAaaMaxAppServer

Returns a value for the maximum number of server entries in the AAA configuration for an application type.

**Parameters**

key—WwnKey

**Return Value**

Maximum number of server entries in a server group in the AAA configuration.

**Error**

Error Code: 400—SnmpException

## isClearAcctLogSet

Checks if the clear accounting log is set.

**Parameters**

key—WwnKey

**Return Value**

Checks if the clear accounting log is set.

**Error**

Error Code: 400—SnmpException

## isMSCHAPRequired

Returns a boolean value to indicate if MSCHAP authentication mechanism is required for authenticating a user.

**Parameters**

key—WwnKey

**Return Value**

Boolean value to indicate if MSCHAP authentication mechanism is required for authenticating a user.

**Error**

Error Code: 400—SnmpException

## getAaaSetup

Returns the AAA configuration.

**Parameters**

key—WwnKey

**Return Value**

AAA configuration.

**Error**

Error Code: 400—SnmpException

## getAaaAppServerGroups

Returns the AAA server groups for a specific application type.

**Parameters**

key—WwnKey

**Return Value**

AAA server groups for a specific application type.

**Error**

Error Code: 400—SnmpException

## getAaaServerGroups

Returns all the AAA server group entries (a server group consists of a number of AAA servers implementing the same AAA protocol).

**Parameters**

key—WwnKey

**Return Value**

AAA server group entries.

**Error**

Error Code: 400—SnmpException

## getSnmpUsers

Returns information about SNMP users.

**Parameters**

key—WwnKey

**Return Value**

Information about SNMP users.

**Error**

Error Code: 400—SnmpException

## getIPACLProfiles

Returns all the IP ACL profiles.

**Parameters**

key—WwnKey

**Return Value**

All the IP ACL profiles.

**Error**

Error Code: 400—SnmpException

## getSSHConfig

Returns the SSH configuration information.

**Parameters**

key—WwnKey

**Return Value**

SSH configuration information.

**Error**

Error Code: 400—SnmpException

## getSSHEnabled

Returns a boolean value to indicate if the SSH is enabled.

**Parameters**

key—WwnKey

**Return Value**

Boolean value to indicate if the SSH is enabled.

**Error**

Error Code: 400—SnmpException

## isTelnetEnabled

Returns a boolean value to indicate if Telnet is enabled.

**Parameters**

key—WwnKey

**Return Value**

Boolean value to indicate if Telnet is enabled.

**Error**

Error Code: 400—SnmpException.

## getPkiRsaKeys

Returns the PKI RSA key pair entries.

**Parameters**

key—WwnKey

**Return Value**

PKI RSA key pair entries.

**Error**

Error Code: 400—SnmpException.

## getPkiTrustPointNames

Returns a list of PKI trustpoint names.

**Parameters**

key—WwnKey

**Return Value**

A list of PKI trustpoint names.

**Error**

Error Code: 400—SnmpException.

## getPkiTrustPointNames

Returns a list of PKI trustpoint names.

**Parameters**

key—WwnKey

**Return Value**

A list of PKI trustpoint names.

**Error**

Error Code: 400—SnmpException.

## getPkiCert

Returns certificate information of a PKI trustpoint.

**Parameters**

key—WwnKey

**Return Value**

Certificate information of a PKI trustpoint.

**Error**

Error Code: 400—SnmpException.



## getPkiAction

Returns the PKI support action of a trustpoint.

**Parameters**

key—WwnKey

**Return Value**

PKI support action of a trustpoint.

**Error**

Error Code: 400—SnmpException.

## getPkiTrustPoint

Returns the PKI trust point information, which consists of the key pair name, a list of revocation methods, and the contact HTTP URL of the external OCSP server for certificate revocation.

**Parameters**

key—WwnKey

**Return Value**

PKI trust point information.

**Error**

Error Code: 400—SnmpException.

## getFeatureControls

Returns all of the feature control names and their respective statuses.

**Parameters**

key—WwnKey

**Return Value**

Feature control names and statuses.

**Error**

Error Code: 400—SnmpException.

## getIkeFailRecoveryCfg

Returns the IKE configuration.

**Parameters**

key—WwnKey

**Return Value**

IKE configuration.

**Error**

Error Code: 400—SnmpException

## getIkeCfgPolicies

Returns the policy that is used to set up the IKE tunnels.

**Parameters**

key—WwnKey

**Return Value**

Policy that is used to set up the IKE tunnels.

**Error**

Error Code: 400—SnmpException.

## getIkeCfgInitiators

Returns the IKE initiator configuration information.

**Parameters**

key—WwnKey

**Return Value**

IKE initiator configuration information.

**Error**

Error Code: 400—SnmpException.

## getIkeTunnels

Returns the IKE tunnels information.

**Parameters**

key—WwnKey

**Return Value**

IKE tunnels information.

**Error**

Error Code: 400—SnmpException.

## getIPsecGlobalCfg

Returns the IPsec tunnel configuration information.

**Parameters**

key—WwnKey

**Return Value**

IPsec tunnel configuration information.

**Error**

Error Code: 400—SnmpException.

## getIPsecXformSets

Returns the IPsec transform set information.

**Parameters**

key—WwnKey

**Return Value**

IPsec transform set information.

**Error**

Error Code: 400—SnmpException.

## getIPsecCryptoMaps

Returns the IPsec cryptomap set.

**Parameters**

key—WwnKey

**Return Value**

IPsec cryptomap set.

**Error**

Error Code: 400—SnmpException.

## getIfsFromCryptoMap

Returns the interface name from the IPsec cryptomap.

**Parameters**

key—WwnKey

**Return Value**

Interface name from the IPsec cryptomap.

**Error**

Error Code: 400—SnmpException.

## getIPsecTunnels

Returns the information about IPsec tunnels.

**Parameters**

key—WwnKey

**Return Value**

Information about IPsec tunnels.

**Error**

Error Code: 400—SnmpException.

## isFipsModeEnabled

Returns information about the FIPS mode.

**Parameters**

key—WwnKey

**Return Value**

Information about FIPS mode.

**Error**

Error Code: 400—SnmpException.

## Protocol WS

### getNtpPeers

Returns NTP peer information.

**Parameters**

key—WwnKey

**Return Value**

NTP peer information.

**Error**

Error Code: 400—SnmpException

### getNtpInfo

Returns NTP system information.

**Parameters**

key—WwnKey

**Return Value**

NTP system information.

**Error**

Error Code: 400—SnmpException

## getFspfConfig

Returns the FSPF protocol configuration.

**Parameters**

key—WwnKey

**Return Value**

Configuration settings of the FSPF protocol.

**Error**

Error Code: 400—SnmpException

## queryInterfaceFspfConfig

Returns the FSPF configuration on the interface pertaining to the specified VSAN.

**Parameters**

key—WwnKey

vsanid—Unique identifier of the VSAN

**Return Value**

FSPF configuration settings on the interface of a specified VSAN.

**Error**

Error Code: 400—SnmpException

## getFcipProfiles

Returns FCIP profiles.

**Parameters**

Key—Fabric key

**Return Value**

List of FCIP profiles.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getFcipProfilesBySwitch

Returns FCIP profiles based on a switch.

### Parameters

Key—Switch key

### Return Value

List of FCIP profiles.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getFcipTunnels

Returns FCIP tunnels.

### Parameters

Key—Fabric key

### Return Value

List of FCIP tunnels.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getFcipTunnelsBySwitch

Returns FCIP tunnels based on a switch.

### Parameters

Key—Switch key

### Return Value

List of FCIP tunnels.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getFcipTunnelByLinkIndex

Returns FCIP tunnels based on a switch.

### Parameters

Key—Switch key

**Return Value**

List of FCIP tunnels.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getFcipTunnelByLinkIfIndex

Returns FCIP tunnels based on a switch.

**Parameters**

Key—Switch key

**Return Value**

List of FCIP tunnels.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getFcipTunnelErrors

Returns FCIP tunnels errors.

**Parameters**

Key—Fabric key

**Return Value**

List of FCIP tunnel errors.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getFcipTunnelErrorsBySwitch

Returns FCIP tunnels errors based on a switch.

**Parameters**

Key—Switch key

**Return Value**

List of FCIP tunnels errors.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getIpSettingsBySwitch

Returns IP settings from the switch.

### Parameters

Key—Switch key

### Return Value

IP settings from the switch.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getIpSettings

Returns IP settings from the switch and port.

### Parameters

Key—Port key

### Return Value

IP settings from the switch and port.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getTcpSettingsBySwitch

Returns TCP settings from the switch.

### Parameters

Key—Switch key

### Return Value

TCP settings from the switch.

### Error

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## getTcpSettings

Returns TCP settings from the switch.



**Parameters**

Key—Switch key

**Return Value**

TCP settings from the switch.

**Error**

Error Code: 300— General SAN Service exception.

Error Code: 201—Invalid argument in Web Service exception.

## Cluster WS - SEI

Service end point interface (SEI) of Cluster WS defines the operations of the service. These methods are published to the end users.

### getSwitchesByFabricKey

Returns all the switches related to a fabric key.

**Parameters**

key—Name of the fabric.

**Return Value**

All the switches associated with the fabric.

**Error**

Error Code: 400—SnmpException

### getServerIpByFabricKey

Returns the managing server IP address from a fabric key.

**Parameters**

key—Name of the fabric.

**Return Value**

IP address of the server.

**Error**

Error Code: 400—SnmpException

### getServerIpBySwitchKey

Returns the managing server IP address from a switch key.

**Parameters**

key—Name of the switch.

**Return Value**

IP address of the server.

**Error**

Error Code: 400—SnmpException

## getFabricsByServerIp

Returns name of the fabric by server IP address.

**Parameters**

key—IP address

**Return Value**

Name of the fabric.

**Error**

Error Code: 400—SnmpException

## getAllServers

Returns all the servers in this federation.

**Parameters**

key—IP address

**Return Value**

All the servers in the federation.

**Error**

Error Code: 400—SnmpException

## getFabricByEnclosureKey

Returns the fabric key from an enclosure key.

**Parameters**

key—Name of the enclosure.

**Return Value**

Name of the fabric.

**Error**

Error Code: 400—SnmpException

## getServerIpByEnclosureKey

Returns the server IP address from an enclosure key.

**Parameters**

key—Name of the fabric.

**Return Value**

IP address of the server.

**Error**

Error Code: 400—SnmpException

## getServerIpByVsanKey

Returns the server IP address from a VSAN key.

**Parameters**

key—Name of the VSAN

**Return Value**

IP address of the server.

**Error**

Error Code: 400—SnmpException

## Event WS - SEI

Service end point interface (SEI) of Event WS defines the operations of the service. These methods are published to the end users.

## isCallHomeEnabled

Returns a boolean value depending upon the activation of the CallHome feature.

**Parameters**

key—WwnKey

**Return Value**

Boolean value

**Error**

Error Code: 400—SnmpException

## getCallHomeDestProfile

Returns the CallHome destination profile.

**Parameters**

key—WwnKey

**Return Value**

**Error**

Error Code: 400—SnmpException

## getCallHomeSysInfo

Returns system information about the CallHome feature.

**Parameters**

key—WwnKey

**Return Value**

System information about the CallHome feature.

**Error**

Error Code: 400—SnmpException

## getEmailMaxEntries

Returns the maximum number of e-mail address entries for the CallHome feature.

**Parameters**

key—WwnKey

**Return Value**

Number of e-mail address entries for the CallHome feature

**Error**

Error Code: 400—SnmpException

## getEmailSetup

Returns the e-mail setup details of the CallHome feature.

**Parameters**

key—WwnKey

**Return Value**

E-mail setup details of CallHome feature.

**Error**

Error Code: 400—SnmpException

## getSyslogServers

Returns the list of syslog servers.

**Parameters**

key—WwnKey

**Return Value**

List of syslog servers.

**Error**

Error Code: 400—SnmpException

## getSyslogMessageControl

Returns a list of syslog message configuration.

**Parameters**

key—WwnKey

**Return Value**

List of syslog message configuration.

**Error**

Error Code: 400—SnmpException

## getSyslogLoggingCfg

Returns syslog logging configuration.

**Parameters**

key—WwnKey

**Return Value**

Configuration information on syslog credentials.

**Error**

Error Code: 400—SnmpException

## Inventory WS - SEI

Service end point interface (SEI) of Inventory WS defines the service end point interface for Inventory Web Service. These methods are published to the end users.

## getPowerSuppliesBySwitchWwnKey

Returns name of the power supplies by switch.

**Parameters**

key—WwnKey

**Return Value**

List of power supply names.

**Error**

Error Code: 300 or 201—SAN service error or Invalid Argument

## getPowerSuppliesBySwitchSnKey

Returns name of the power supplies by switch.

### Parameters

key—swKey

### Return Value

List of power supply names.

### Error

Error Code: 300 or 201—SAN service error or Invalid Argument

## getPowerSuppliesBySwitchIP

Returns name of the power supplies by switch IP address.

### Parameters

key—swKey

### Return Value

List of power supply names.

### Error

Error Code: 300 or 201—SAN service error or Invalid Argument

## getCardsBySwitchWwnKey

Returns name of the cards from a switch key.

### Parameters

key—swKey

### Return Value

List of card names.

### Error

Error Code: 300 or 201—SAN service error or Invalid Argument

## getCardsBySwitchSnkey

Returns name of the cards from a switch key.

### Parameters

key—swKey

**Return Value**

List of card names.

**Error**

Error Code: 300 or 201—SAN service error or Invalid Argument

## getCardsBySwitchIP

Returns name of the cards by switch IP address.

**Parameters**

key—swKey

**Return Value**

List of card names.

**Error**

Error Code: 300 or 201—SAN service error or Invalid Argument

## getFansBySwitchWwnKey

Returns name of the fans by switch key.

**Parameters**

key—swKey

**Return Value**

List of fan names.

**Error**

Error Code: 300 or 201—SAN service error or Invalid Argument

## getFansBySwitchSnKey

Returns name of the fans by switch key.

**Parameters**

key—swKey

**Return Value**

List of fan names.

**Error**

Error Code: 300 or 201—SAN service error or Invalid Argument

## getFansBySwitchIP

Returns name of the fans by switch IP address.

**Parameters**

key—swKey

**Return Value**

List of fan names.

**Error**

Error Code: 300 or 201—SAN service error or Invalid Argument

## getChassisBySwitchWwnKey

Returns name of the chassis by switch key.

**Parameters**

key—swKey

**Return Value**

List of chassis names.

**Error**

Error Code: 300 or 201—SAN service error or Invalid Argument

## getChassisBySwitchSnKey

Returns name of the chassis by switch key.

**Parameters**

key—swKey

**Return Value**

List of chassis names.

**Error**

Error Code: 300 or 201—SAN service error or Invalid Argument

## getChassisBySwitchIP

Returns name of the chassis by switch IP address.

**Parameters**

key—swKey

**Return Value**

List of chassis names.

**Error**

Error Code: 300 or 201—SAN service error or Invalid Argument



## getAllHbas

Returns name of the hosts by host IP address.

**Parameters**

key—hba WWN

**Return Value**

List of HBAs.

**Error**

Error Code: 300 or 201—SAN service error or Invalid Argument

## getHbaByWwn

Returns name of the hosts by host IP address.

**Parameters**

key—hba WWN

**Return Value**

List of HBAs.

**Error**

Error Code: 300 or 201—SAN service error or Invalid Argument

## getLicensesBySwitchWwnKey

Returns name of the licenses used by the switch.

**Parameters**

key—swKey

**Return Value**

List of licenses.

**Error**

Error Code: 300 or 201—SAN service error or Invalid Argument

## getLicensesBySwitchIP

Returns name of the licenses used by the switch.

**Parameters**

key—swKey

**Return Value**

List of licenses.

**Error**

Error Code: 300 or 201—SAN service error or Invalid Argument

## getLicenseFlags

Returns name of the licenses used by the switch.

### Parameters

key—swKey

### Return Value

List of licenses.

### Error

Error Code: 300 or 201—SAN service error or Invalid Argument

## getCardByPhysicalIndex

Returns the physical and switch index.

### Parameters

key—swKey

### Return Value

List of indexes.

### Error

Error Code: 300 or 201—SAN service error or Invalid Argument

# Error Codes

Error Code	Description
100	Authentication failure
101	Invalid credential
102	Invalid privilege
103	Invalid token
200	Web Service error
201	Invalid argument in Web Service function
202	Unreachable Web Service server
300	SAN service error
301	Invalid query key
400	SNMP error
201	Invalid argument


**Note**

DCNM-SAN Web Services supports server federation. Service requests to SanWS, SecurityWS, ProtocolWS, EventWS, and InventoryWS automatically dispatches the calls to the correct server in the federation. If you are using server federation, the following methods do not automatically mediate to the corresponding server in the federation:

**SanWS:**

```
getEnclosures()
getEndpoints()
getFabricByIP()
getHosts()
getStorages()
getSwitchIPByName()
getSwitches()
```

**InventoryWS:**

```
getAllHbas()
getLicenseFlags()
```

In those specific instances, you might need to rely on ClusterWS to determine the server that you need to send the request.

