Cisco BroadWorks

Network Server Portal API

Specification Guide

Release 23.0

Document Version 2

Notification

The BroadSoft BroadWorks has been renamed to Cisco BroadWorks. Beginning in September 2018, you will begin to see the Cisco name and company logo, along with the new product name on the software, documentation, and packaging. During this transition process, you may see both BroadSoft and Cisco brands and former product names. These products meet the same high standards and quality that both BroadSoft and Cisco are known for in the industry.

Copyright Notice

Trademarks

Any product names mentioned in this document may be trademarks or registered trademarks of Cisco or their respective companies and are hereby acknowledged.

Document Revision History

Release	Version	Reason for Change	Date	Author
14.0	1	Updated document for rebranding.	January 31, 2006	Roberta Boyle
14.0	1	Rewrote entire document.	June 9, 2006	Michel Tougas
14.0	1	Edited document.	August 3, 2006	Patricia Renaud
14.0	2	Updated document to reflect changes to portal API (*webImpactingPatches* replaced *minimumWebPatchLevel* and *patchLevel* in Release 14.0).	August 29, 2006	Joël Collin
14.0	2	Edited changes.	March 2, 2007	Patricia Renaud
14.0	3	Changed the directory number (DN) length to 21 in section *5 Validation Requirements*.	July 7, 2008	Yves Racine
14.0	3	Edited changes and published document.	July 13, 2008	Patricia Renaud
15.0	1	Updated document for Releases 14.sp1 through 14.sp6 and Release 15.0.	July 14, 2008	Patricia Renaud
16.0	1	Updated document for Release 16.0.	March 30, 2009	Yves Racine
16.0	1	Edited changes and published document.	May 27, 2009	Andrea Fitzwilliam
16.0	2	Added the parameter, *clusterMode,* to the command GetAllHostingNeNode Addresses, for EV 93608.	June 1, 2009	Yves Racine
16.0	2	Edited changes and published document.	June 10, 2009	Andrea Fitzwilliam
17.0	1	Updated document for Release 17.0.	March 12, 2010	Juliette D'Almeida
17.0	1	Edited changes and published document.	March 31, 2010	Andrea Fitzwilliam
18.0	1	Updated document for Release 18.0.	October 7, 2011	Pierre Drapeau
18.0	1	Edited changes and published document.	October 19, 2011	Andrea Fitzwilliam
18.0	2	Updated section *4.1.3 Error Example – Invalid DN* for EV 146496. Changed the response to invalid DN when trying to get a DN that does not exist with GetServingAS.	January 19, 2011	Yves Racine
18.0	2	Edited changes and published document.	February 20, 2012	Andrea Fitzwilliam
19.0	1	Updated GetAllHostingNeNodes Addresses servlet and deprecated GetServingAS servlet.	October 11, 2012	Pierre Drapeau
19.0	1	Edited changes and published document.	October 30, 2012	Patricia Renaud
19.0	2	Updated GetDeviceFileServingAS description in section *4.8 GetDeviceFileServingAS* for EV 182704.	July 31, 2013	Pierre Drapeau
20.0	1	Updated document for Release 20.0.	September 5, 2013	Pierre Drapeau
20.0	1	Edited changes and published document.	October 18, 2013	Joan Renaud
21.0	1	Updated document for Release 21.0.	September 24, 2014	Yves Racine

Release	Version	Reason for Change	Date	Author
21.0	1	Edited changes and updated copyright notice.	September 29, 2014	Joan Renaud
21.0	1	Rebranded and published document.	December 9, 2014	Joan Renaud
22.0	1	Updated document for Release 22.0.	October 7, 2016	Eric Ross
22.0	1	Edited changes and published document.	December 12, 2016	Joan Renaud
23.0	1	Updated document for Release 23.0.	September 10, 2018	Yves Racine
23.0	1	Rebranded document for Cisco. Edited changes and published document.	September 30, 2018	Joan Renaud
23.0	2	Completed rebranding for Cisco and republished document.	March 14, 2019	Patricia Renaud

Table of Contents

1 Summary of Changes

This section describes the changes to this document for each release and document version.

1.1 Changes for Release 23.0, Document Version 2

This version of the document includes the following change:

- Completed rebranding for Cisco and republished document.

1.2 Changes for Release 23.0, Document Version 1

This version of the document includes the following changes:

- Updated the GetServingAS command to remove the parameters extension, groupId, and hostId.
- Updated the GetDeviceFileServingAS command to add the *fileName* parameter.

1.3 Changes for Release 22.0, Document Version 1

This version of the document includes the following change:

- Updated document to reflect the support for Hypertext Transfer Protocol Secure Sockets (HTTPS).

1.4 Changes for Release 21.0, Document Version 1

This document was created from Release 20.0, version 1.

1.5 Changes for Release 20.0, Document Version 1

This document was created from Release 19.0, version 2.

1.6 Changes for Release 19.0, Document Version 2

This version of the document includes the following change:

- Updated the GetDeviceFileServingAS command description in section *4.8 GetDeviceFileServingAS* for EV 182704.

1.7 Changes for Release 19.0, Document Version 1

This document is updated to reflect the introduction of the Network Element (NE) maintenance partitions and its impacts on the Portal API. In Release 19.0, the network can be split into one or more NE maintenance partitions. An NE maintenance partition is defined as a set of zero or more Execution Servers, zero or more Profile Servers, and zero or more Xtended Services Platforms. Therefore, an NE maintenance partition is a subset of the entire network, and the Portal API response should only return servers listed under a specific NE maintenance partition.

This version of the document includes the following change:

- Updated GetAllHostingNENodeAddresses servlet and deprecated GetServingAS servlet.

1.8 Changes for Release 18.0, Document Version 2

This version of the document includes the following change:

- Updated section *4.1.3 Error Example – Invalid DN* for EV 146496. (Changed the response to invalid DN when trying to get a DN that does not exist with GetServingAS.)

1.9 Changes for Release 18.0, Document Version 1

This version of the document includes the following change:

- Added parameter, *callPRequest,* to the Network Server portal API commands GetHostingNEInfo, GetServingAS, and GetAllHostingNENodeAddresses.

1.10 Changes for Release 17.0, Document Version 1

This version of the document includes the following changes:

- Added the parameter, *returnCompatibleXSP,* to the command GetServingAS.
- Added the *version* parameter as part of the cluster in the response to the GetDeviceFileServingAS command.

1.11 Changes for Release 16.0, Document Version 2

This version of the document includes the following change:

- Added the parameter, *clusterMode,* to the command GetAllHostingNeNodeAddresses for EV 93608.

1.12 Changes for Release 16.0, Document Version 1

This version of the document includes the following change:

- Introduced the command GetAllHostingNeNodeAddresses.

1.13 Changes for Release 15.0, Document Version 1

There were no changes to this document for Release 15.0.

1.14 Changes for Releases 14.sp1 through 14.sp6, Document Version 1

There were no changes to this document for Releases 14.sp1 through 14.sp6.

2 Overview

The Network Server portal application programming interface (API) constitutes a set of commands based on Hypertext Transfer Protocol (HTTP) requests, which allows a portal server to communicate with the Network Server. The interface consists of commands that are used to query the Network Server based on user information.

The command interface is built upon the HTTP protocol. A client application sends an HTTP-based portal API request to the Cisco BroadWorks web portal running on the Network Server. The Network Server Provisioning Server (PS) processes the request and returns the response to the Cisco BroadWorks web portal, which then returns an HTTP response to the client.

To prevent unauthorized portal API requests, the Network Server maintains an access control list that contains the only addresses from which portal API requests are accepted.

Support of the portal API over Hypertext Transfer Protocol Secure Sockets (HTTPS) is possible.

3 Network Server Portal API Request and Response

3.1 Network Server Portal API Request URL

To invoke a portal API command, the client application needs to send an HTTP GET command to the appropriate Network Server. To do this, the following uniform resource locator (URL) format shall be used:

```
http://<host>/servlet/<servletNameAndParameters>
```

… where <host> is the address or host name of the Network Server and <servletNameAndParameters> is the Network Server portal API command to invoke. The <host> portion can also include a port, like in ns.provider.com:80.

The <servletNameAndParameters> contains the name of the servlet to invoke and, optionally, URL parameters provided by the client application. If <servletNameAndParameters> contains URL parameters, it is encoded as follows:

```
<servletNameAndParameters> = <servletName>?<p1>=<v1>&<p2>=<v2>
```

… where <servletName> is the name of the servlet, <p1> is a parameter name, <v1> is the value given to <p1>, <p2> is another parameter name, <v2> is the value given to <p2>, and so on.

The Network Server portal API supports the following commands (<servletName>). The URL parameters that each command supports are listed beside each command name.

Servlet Name	Required URL Parameters
GetWebServerPortal	*<none>,* or *version,* or *version, webImpactingPatches*
GetHostingNEInfo	*url,* or *url, callPRequest*
AuthorizeToken	*userId, password, token*
GetHostNodeAddresses	*address,* or *address, private*
GetServingAS (deprecated)	*dn,* or *dn, returnCompatibleXSP,* or *url,* or *url, returnCompatibleXSP,* or *dn, private,* or *dn, private, returnCompatibleXSP,* or *url, private,* or *url, private, returnCompatibleXSP,* or *url, callPRequest*
GetHostsForEnterprise	*<none>* or *enterpriseName*
GetAllHostingNeNodeAddresses	*<none>,* or *private (deprecated),* or *broadworksOnly,* or *private, broadworksOnly,* or *callPRequest* *disregardPartitions* *clusterMode* *enterpriseName*

Servlet Name	Required URL Parameters
GetDeviceFileServingAS	*<none>,* or *url,* or *mac,* or *username,* or *fileFormatId, or* *fileName*

3.1.1 URL Parameter Description

URL Parameter Name	Description
address	A string identifying one address of a hosting network element (NE) node.
broadworksOnly	A *true* or *false* flag. ▪ *true* means that addresses are requested for "broadworks" hosting network element (hosting NE of type BroadWorks in Network Server). ▪ *false* means that addresses are requested for any type of hosting network element (hosting NE of any type in Network Server).
Dn	A unique directory number associated with one and only one group within the Network Server. The dn should be an E.164 number, including the country code (leading + is assumed if not specified). The dn should not contain any visual separator.
enterpriseName	A string identifying the name of an enterprise within the Network Server.
Password	The password of an administrator provisioned on the Network Server.
Private	A *true* or *false* flag: ▪ *true* means that private addresses are requested (type Signaling and DualRouting in Network Server). ▪ *false* means that public addresses are requested (type Access and DualRouting in Network Server).
Token	A string that is temporarily mapped to a user ID and password. The client application should ensure the uniqueness of the given string.
url	A unique URL (alias or login ID) associated with one and only one group within the Network Server. The URL is case sensitive.
userId	The user ID of an administrator provisioned on the Network Server.
Version	An alphanumeric string identifying the version of the Application Server or Web Server. The format of this field is as follows: <type>_<release>_<id> where: ▪ "type" is either Application Server (AS) or Web Server (WS), ▪ "release" is the major release number (for example, Rel_14.0), and ▪ "id" is the minor release number (for example, 1.122).
webImpactingPatches	List of semi-comma separated patch numbers that affect the communication between Application Server and Web Server.

URL Parameter Name	Description
url (getDeviceFileServingAS)	The name and path of the file being requested by an access device in the context of device management.
Mac	The access device MAC address.
Username	The access device user name.
fileFormatId	A unique identifier that identifies an access device file format on the Network Server. This ID is mainly used to speed up the URL mapping for the second request for the same URL.
filename	The name of the file being requested by an access device in the context of device management (without the path).
clusterMode	A *true* or *false* flag: ▪ *true* means that addresses are returned by cluster, one address for the primary, one address for the secondary. If there is no address defined for the secondary, an empty string is returned. ▪ *false* indicates that all addresses of a cluster are returned.
returnCompatibleXSP	A *true* or *false* flag: ▪ *true* means that the response message contains an xspAddressArray block of data, which consists of a list of Xtended Services Platform IP addresses corresponding to the Application Server where the user is located. ▪ *false* indicates that the xspAddressArray block of data is not included in the response.
callPRequest	A *true* or *false* flag: This is an optional parameter that specifies the list of servers of interest and indicates whether the list of call processing servers or the list of provisioning servers is returned in the portal API response. If omitted, the list of provisioning servers is returned. In a hosting model composed only of Application Servers in primary-secondary mode, the portal API response is the same whether the request is for the list of call processing servers or provisioning servers. Therefore, in this case, the parameter value is irrelevant. ▪ *true* indicates the list of call processing servers is returned in the portal API response with the exception described above. ▪ *false* indicates the list of provisioning servers is returned in the portal API response.

3.2 Network Server Portal API Response

The Cisco BroadWorks web portal on the Network Server responds to an HTTP-based portal API request by returning an HTTP *200 OK* response containing a content of type text/html. The response content is an XML-encoded string. The Network Server portal API XML definition provides the schematic description of all portal API responses.

3.2.1 Document Type Definition (DTD)

```
<?xml version="1.0" encoding="UTF-8"?>

<!ELEMENT com.broadsoft.protocols.nsportal.WebServerRequest
(webServerArray)>
<!ATTLIST com.broadsoft.protocols.nsportal.WebServerRequest
version               CDATA  #IMPLIED
webImpactingPatches   CDATA  #IMPLIED
>

<!ELEMENT webServerArray
(com.broadsoft.protocols.nsportal.WebServerData+)>

<!ELEMENT com.broadsoft.protocols.nsportal.WebServerData    EMPTY>
<!ATTLIST com.broadsoft.protocols.nsportal.WebServerData
address                 CDATA         #REQUIRED
isBrandingMaster        (false|true)  #REQUIRED
type                    CDATA         #REQUIRED
version                 CDATA         #IMPLIED
webImpactingPatches     CDATA         #IMPLIED
>

<!ELEMENT com.broadsoft.protocols.nsportal.HostingNEInfoRequest
(applicationServerArray)>
<!ATTLIST com.broadsoft.protocols.nsportal.HostingNEInfoRequest
hostingNe  CDATA  #REQUIRED
url        CDATA  #REQUIRED
>

<!ELEMENT applicationServerArray
(com.broadsoft.protocols.nsportal.ApplicationServerData+)>

<!ELEMENT com.broadsoft.protocols.nsportal.ApplicationServerData
EMPTY>
<!ATTLIST com.broadsoft.protocols.nsportal.ApplicationServerData
address                 CDATA  #REQUIRED
webImpactingPatches     CDATA  #IMPLIED
type                    CDATA  #REQUIRED
version                 CDATA  #IMPLIED
>
<!ELEMENT com.broadsoft.protocols.nsportal.AuthorizationRequest    EMPTY>
<!ATTLIST com.broadsoft.protocols.nsportal.AuthorizationRequest
password  CDATA  #IMPLIED
token     CDATA  #REQUIRED
userId    CDATA  #REQUIRED
>

<!ELEMENT com.broadsoft.protocols.nsportal.PortalRequest
(addressArray, xspAddressArray? )>

<!ATTLIST com.broadsoft.protocols.nsportal.PortalRequest
dn          CDATA         #IMPLIED
isPrivate   (false|true)  #REQUIRED
```

```
url          CDATA          #IMPLIED
clusterMode (false|true)  #IMPLIED
>
<!ELEMENT addressArray    (string*)>
<!ELEMENT xspAddressArray (string*)>

<!ELEMENT com.broadsoft.protocols.nsportal.DeviceFileRequest
(clusterArray)>
<!ATTLIST com.broadsoft.protocols.nsportal.DeviceFileRequest
url                          CDATA         #REQUIRED
mac                          CDATA         #IMPLIED
username                     CDATA         #IMPLIED
fileFormatId                 CDATA         #IMPLIED
filename                     CDATA         #IMPLIED
deviceFileAsLookupEnabled  (false|true)  #REQUIRED
needMoreInfo               (false|true)  #REQUIRED
macFormatInNonRequestURI     CDATA         #IMPLIED
challenge                    CDATA         #IMPLIED
>

<!ELEMENT clusterArray (cluster*)>
<!ELEMENT cluster    EMPTY>
<!ATTLIST cluster
hostingNe CDATA  #REQUIRED
primary   CDATA  #REQUIRED
secondary CDATA  #IMPLIED
version   CDATA  #IMPLIED
>

<!ELEMENT com.broadsoft.protocols.nsportal.EnterpriseHostsRequest
(hostsArray)>
<!ATTLIST com.broadsoft.protocols.nsportal.EnterpriseHostsRequest
enterpriseName  CDATA         #IMPLIED
>

<!ELEMENT hostsArray    (string*)>
<!ELEMENT string    EMPTY>
<!ATTLIST string
value  CDATA  #IMPLIED
>

<!ELEMENT com.broadsoft.protocols.nsportal.Error    EMPTY>
<!ATTLIST com.broadsoft.protocols.nsportal.Error
detail   CDATA  #IMPLIED
id       CDATA  #IMPLIED
summary  CDATA  #IMPLIED>
```

3.2.2 Document Type Definition Description

Attribute or Element	Description
Address	A string identifying one address of a hosting NE node or a Web Server Farm node.
addressArray	An element containing ordered addresses for a hosting network element.
xspAddressArray	An element containing ordered Xtended Services Platform IP addresses.
com.broadsoft.protocols.nsportal.Error	An element that is a container for describing problems and issues. It has three attributes: ID, summary, and detail.
challenge	Indicate whether HTTP Basic or Digest authentication should be used to get the access device user name.
cluster	The name of the Application Server and its corresponding primary and secondary IP address.
clusterArray	An element containing the Application Servers that host a given access device file format.
clusterMode	A *true* or *false* flag: ▪ *true* means that addresses are returned by cluster, one address for the primary, one address for the secondary. If there is no address defined for the secondary, an empty string is returned. ▪ *false* indicates that all addresses of a cluster are returned
Detail	The detailed description of an error.
deviceFileAsLookupEnabled	Indicate whether the access device file lookup is enabled on the Network Server.
Dn	A unique directory number associated with one and only one group within the Network Server.
fileFormatId	A unique identifier that identifies an access device file format on the Network Server. This ID is mainly used to speed up the URL mapping for the second request (for the same URL).
filename	The name of the file being requested by an access device in the context of device management (without the path).
groupId	A unique group identifier associated with one and only one group and Application Server within the Network Server.
hostsArray	An element containing ordered hosting network elements that host a given enterprise.
hosted	The IP address or host name for an Application Server.
hostingNe	A unique alphanumeric string identifying an Application Server or other hosting NE within the Network Server.
Id	A numeric identifier for error messages within the Network Server.
isBrandingMaster	An indicator of whether a Web Server node is a branding master or not. A branding master is the only Web Server capable of changing the branding on an Application Server.
isPrivate	A *true* or *false* attribute: ▪ *true* means that private addresses are included in the response (type Signaling and DualRouting in Network Server). ▪ *false* means that public addresses are included in the response (type Access and DualRouting in Network Server).

Attribute or Element	Description
mac	The access device MAC address.
macFormatInNonRequestURI	The format of the MAC address when present in an *HTTP* header.
needMoreInfo	A *true* or *false* attribute: ▪ *true* means that the Network Server has found the access device file format but needs the access device MAC address or the access device user name. ▪ *false* means that the Network Server has finished processing a request and the result can be found in the *clusterArray* element.
Password	The password of an administrator provisioned on the Network Server. In responses, this attribute is empty.
primary	The primary IP address of the Application Server or the address of the call processing or provisioning server.
secondary	The secondary IP address of the Application Server.
Summary	The summary text of an error.
Token	A string that is temporarily mapped to a user ID and password.
Type	The type of hosting NE node address or type of Web Server Farm node address.
url	A unique URL (alias or login ID) associated with one and only one group within the Network Server.
url (getDeviceFileServingAS)	The name and path of the file being requested by an access device in the context of device management.
userId	The user ID of an administrator provisioned on the Network Server.
username	The access device user name.
Value	String value in a string element.
Version	An alphanumeric string identifying the version of the Application Server or Web Server. The format of this field is as follows: <type>_<release>_<id> where: ▪ "type" is either Application Server (AS), Execution Server (XS), Provisioning Server (PS), or Web Server (WS) ▪ "release" is the major release number (for example, Rel_14.0), and ▪ "id" is the minor release number (for example, 1.122).
webImpactingPatches	An alphanumeric string identifying the list of web-impacting patches installed on a server.

4 Network Server Portal API Commands

4.1 GetServingAS (deprecated)

This command returns the list of public or private addresses (IP addresses or host names) for the serving hosting NE server associated with a given directory number or URL The returned list can contain addresses other than the serving hosting NE server, for instance when the serving hosting NE server is part of a replicated cluster; however, the addresses for the serving hosting NE server are listed first.

Public addresses are also known as “Access” addresses, while private addresses are also known as “Signaling” addresses. “DualRouting” addresses are included in the resulting list regardless of the value of the *private* parameter, because these addresses are public and private at the same time.

This command includes an optional parameter, *returnCompatibleXSP,* which when set to “true”, queries the Network Server for the list of valid Xtended Services Platforms corresponding to the hosting NE server where the user is located.

This command includes an optional parameter, *callPRequest,* which specifies the list of servers of interest and indicates whether the list of call processing servers or the list of provisioning servers is returned in the portal API response. If omitted, the list of provisioning servers is returned. In a hosting model composed only of Application Servers in primary-secondary mode, the portal API response is the same whether the request is for the list of call processing servers or provisioning servers. Therefore, in this case, the parameter value is irrelevant.

Note that the following examples assume that the dual-homed hosting NE server with the public IP address, 169.254.61.198, and private IP address, 192.168.8.215, exists on the Network Server system and is associated with the directory number (DN) “15146987500”, with the URL “user11@broadsoft.com”.

4.1.1 Directory Number Example – Public, Parameter returnCompatibleXSP Set to “True”

```
HTTP GET
http://mtlns03/servlet/GetServingAS?dn=%2B15146987500&returnCompatibleXSP
=true

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.PortalRequest
 dn="+15146987500"
 isPrivate="false"
 url=""
>
  <addressArray>
    <string value="dualrouting.broadsoft.com"></string>
    <string value="169.254.61.198"></string>
    <string value="access.broadsoft.com"></string>
  </addressArray>
  <xspServerArray>
    <string value="1.1.1.1"></string>
    <string value="2.2.2.2"></string>
  </xspServerArray>

</com.broadsoft.protocols.nsportal.PortalRequest>
```

4.1.2 Uniform Resource Locator (URL) Example – Private

```
HTTP GET
http://mtlns03/servlet/GetServingAS?url=user11%40broadsoft.com&private=true

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.PortalRequest
 dn=""
 isPrivate="true"
 url="user11@broadsoft.com"
>
  <addressArray>
    <string value="dualrouting.broadsoft.com"></string>
    <string value="signaling.broadsoft.com"></string>
    <string value="192.168.8.215"></string>
  </addressArray>
</com.broadsoft.protocols.nsportal.PortalRequest>
```

4.1.3 Error Example – Invalid DN

```
HTTP GET
http://mtlns03/servlet/GetServingAS?dn=%2B151469875009

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.Error
  detail="Unknown DN "
  id="0"
  summary="Cannot complete request"
>
</com.broadsoft.protocols.nsportal.Error>
```

4.2 AuthorizeToken

This command prepares a short-lived token that is used instead of a user ID/password, when redirecting to the Network Server login servlet.

This command is used to prepare a pre-authenticated login to the Network Server, for instance for an enterprise administrator. To prepare a pre-authenticated login to an Application Server, such as for an end user or group administrator, the equivalent command in the Application Server portal API can be used. Alternatively, the Application Server can be set up to use an external authentication client (this applies to the Release 10 External Authentication Support functionality).

NOTE 1: The client application must ensure the token is unique.

NOTE 2: If the transaction is successful, the token is valid for 60 seconds or until it is used, whichever occurs first. The transaction fails only if data is missing. All other validation (invalid user ID, invalid password, and so on) is performed when the token is used to open a web session on the Network Server.

4.2.1 Valid Request Example

```
HTTP GET
http://mtlns03/servlet/AuthorizeToken?userId=admin&password=TgFu7Rii!&token=867984373487345945783845386754 6

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.AuthorizationRequest
 password=""
 token="867984373487345945783845386754 6"
 userId="admin"
>
</com.broadsoft.protocols.nsportal.AuthorizationRequest>
```

4.2.2 Invalid Request Example

```
HTTP GET
http://mtlns03/servlet/AuthorizeToken?userId=admin&password=TgFu7Rii!

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.Error
  detail="Must specify a token "
  id="0"
  summary="Cannot complete request"
>
</com.broadsoft.protocols.nsportal.Error>
```

4.3 GetHostNodeAddresses

This command returns the list of public or private addresses (IP addresses or host names) for the serving Application Server nodes associated with a given Application Server address or alias. Public addresses are also known as “Access” addresses, while private addresses are also known as “Signaling” addresses. “DualRouting” addresses are included in the resulting list regardless of the value of the *private* parameter, because those addresses are public and private at the same time.

If the Application Server has an External Web Server (this applies to Release 10 External Web Server Support functionality), then the “public” address is the address of the external Web Server, that is, the server name of the External Web Server. In this same scenario, the “private” address is the address (server name) of the collocated Web Server on the Application Server host.

4.3.1 Public Example

```
HTTP GET
http://mtlns03/servlet/GetHostNodeAddresses?address=192.168.8.215

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.PortalRequest
 dn=""
 hostId="192.168.8.215"
 isPrivate="false"
 url=""
```

```
>
  <addressArray>
    <string value="dualrouting.broadsoft.com"></string>
    <string value="access.broadsoft.com"></string>
    <string value="169.254.61.198"></string>
  </addressArray>
</com.broadsoft.protocols.nsportal.PortalRequest>
```

4.3.2 Private Example

```
HTTP GET
http://mtlns03/servlet/GetHostNodeAddresses?address=192.168.8.215&private
=true

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.PortalRequest
 dn=""
 hostId="192.168.8.215"
 isPrivate="true"
 url=""
>
  <addressArray>
    <string value="dualrouting.broadsoft.com"></string>
    <string value="signaling.broadsoft.com"></string>
    <string value="192.168.8.215"></string>
  </addressArray>
</com.broadsoft.protocols.nsportal.PortalRequest>
```

4.3.3 Error Example – Invalid Address

```
HTTP GET
http://mtlns03/servlet/GetHostNodeAddresses?address=192.168.8.214&private
=false

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.Error
  detail="Invalid host name "
  id="0"
  summary="Cannot complete request"
>
</com.broadsoft.protocols.nsportal.Error>
```

4.4 GetHostingNEInfo

This command returns the complete hosting NE information for a given user specified by its complete user ID (URL). This includes the hosting cluster fully qualified domain name (FQDN) as well as the list of public addresses (IP addresses or host names) for each hosting NE server in the replicated cluster. For each hosting NE server, its running software version is returned, as registered.

This command includes an optional parameter, *callPRequest,* which specifies the list of servers of interest and indicates whether the list of call processing servers or the list of provisioning servers is returned in the portal API response. If omitted, the list of provisioning servers is returned.

In a hosting model composed only of Application Servers in primary-secondary mode, the portal API response is the same whether the request is for the list of call processing servers or provisioning servers. Therefore, in this case, the parameter value is irrelevant. In this model, the type of server (primary or secondary) is returned.

4.4.1 Command Example

```
HTTP GET
http://mtlns03/servlet/GetHostingNEInfo?url=user11%40broadsoft.com

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.HostingNEInfoRequest
 hostingNe="AppServer1"
 url="user11@broadsoft.com"
>
  <applicationServerArray>
    <com.broadsoft.protocols.nsportal.ApplicationServerData
      address="169.254.61.198"
      webImpactingPatches="32487;36813"
      type="primary"
      version="AS_Rel_14.0_1.600">
    </com.broadsoft.protocols.nsportal.ApplicationServerData>
    <com.broadsoft.protocols.nsportal.ApplicationServerData
      address="dualrouting.broadsoft.com"
      webImpactingPatches="32487;36813"
      type="primary"
      version=" AS_Rel_14.0_1.600">
    </com.broadsoft.protocols.nsportal.ApplicationServerData>
    <com.broadsoft.protocols.nsportal.ApplicationServerData
      address="access.broadsoft.com"
      webImpactingPatches="32487;36813"
      type="primary"
      version=" AS_Rel_14.0_1.600">
    </com.broadsoft.protocols.nsportal.ApplicationServerData>
    <com.broadsoft.protocols.nsportal.ApplicationServerData
      address="sec.public.broadsoft.com"
      webImpactingPatches="32487;36813"
      type="secondary"
      version=" AS_Rel_14.0_1.600">
    </com.broadsoft.protocols.nsportal.ApplicationServerData>
    <com.broadsoft.protocols.nsportal.ApplicationServerData
      address="sec.dual.broadsoft.com"
      webImpactingPatches="32487;36813"
      type="secondary"
      version=" AS_Rel_14.0_1.600">
    </com.broadsoft.protocols.nsportal.ApplicationServerData>
  </applicationServerArray>
</com.broadsoft.protocols.nsportal.HostingNEInfoRequest>
```

4.4.2 Error Example

```
HTTP GET
http://mtlns03/servlet/GetHostingNEInfo?url=user12%40broadsoft.com

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.Error
```

```
  detail="Unknown URL "
  id="0"
  summary="Cannot complete request"
>
</com.broadsoft.protocols.nsportal.Error>
```

4.5 GetWebServerPortal

This command returns a list of Web Servers running the specified software version.

4.5.1 Command Example – All Web Servers

```
HTTP GET
http://mtlns03/servlet/GetWebServerPortal

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.WebServerRequest
 webImpactingPatches=""
 version=""
>
<webServerArray>
  <com.broadsoft.protocols.nsportal.WebServerData
    address="wsf2.access.2"
    isBrandingMaster="false"
    type="Access"
    version="WS_Rel_14.0_1.487"
    webImpactingPatches="32487;36813">
  </com.broadsoft.protocols.nsportal.WebServerData>
  <com.broadsoft.protocols.nsportal.WebServerData
    address="wsf.access.1"
    isBrandingMaster="true"
    type="Access"
    version="WS_Rel_14.0_1.487"
    webImpactingPatches="32487;36813">
  </com.broadsoft.protocols.nsportal.WebServerData>
  <com.broadsoft.protocols.nsportal.WebServerData
    address="wsf2.access.1"
    isBrandingMaster="false"
    type="Access"
    version="WS_Rel_14.0_1.487"
    webImpactingPatches="32487;36813">
  </com.broadsoft.protocols.nsportal.WebServerData>
  <com.broadsoft.protocols.nsportal.WebServerData
    address="wsf.access.2"
    isBrandingMaster="true"
    type="Access"
    version="WS_Rel_14.0_1.487"
    webImpactingPatches="32487;36813">
  </com.broadsoft.protocols.nsportal.WebServerData>
</webServerArray>
</com.broadsoft.protocols.nsportal.WebServerRequest>
```

4.5.2 Command Example – Specific Web Impacting Patches

```
HTTP GET
http://mtlns03/servlet/GetWebServerPortal?version=WS_Rel_14.0_1.487&patch
Level=mp2
```

```
HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.WebServerRequest
 minimumWebPatchLevel="32487;36813"
 version="WS_Rel_14.0_1.487"
>
<webServerArray>
  <com.broadsoft.protocols.nsportal.WebServerData
    address="wsf2.access.2"
    isBrandingMaster="false"
    type="Access"
    version="WS_Rel_14.0_1.487"
    webImpactingPatches="32487;36813">
  </com.broadsoft.protocols.nsportal.WebServerData>
  <com.broadsoft.protocols.nsportal.WebServerData
    address="wsf2.access.1"
    isBrandingMaster="false"
    type="Access"
    version="WS_Rel_14.0_1.487"
    webImpactingPatches="32487;36813">
  </com.broadsoft.protocols.nsportal.WebServerData>
</webServerArray>
</com.broadsoft.protocols.nsportal.WebServerRequest>
```

4.5.3 Command Example – No Match Found

```
HTTP GET
http://mtlns03/servlet/GetWebServerPortal?version=WS_Rel_14.0_1.487&patch
Level=mp3

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.Error
  detail="No web server found for specified version/patch-level "
  id="0"
  summary="Cannot complete request"
>
</com.broadsoft.protocols.nsportal.Error>
```

4.6 GetHostsForEnterprise

This command returns a list of hosting NEs that hosts a given enterprise.

4.6.1 Command Example – All Hosting Network Elements of Enterprise

```
HTTP GET
http://mtlns03/servlet/GetHostsForEnterprise?enterpriseName=Broadsoft

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.EnterpriseHostsRequest
   enterpriseName=”Broadsoft”
>
<hostsArray>
  <string value=”ASsanity”/>
  <string value=”ASaus”/>
</ hostsArray >
```

```
</com.broadsoft.protocols.nsportal.EnterpriseHostsRequest >
```

4.6.2 Command Example – No Match Found

```
HTTP GET
http://mtlns03/servlet/GetHostsForEnterprise?enterpriseName=unknownEntepr
ise

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.Error
  detail="Enterprise does not exist"
  id="0"
  summary="Cannot complete request"
>
</com.broadsoft.protocols.nsportal.Error>
```

4.7 GetAllHostingNeNodeAddresses

This command returns the list of addresses (IP addresses or host names) for the serving hosting NE server nodes. This command is capable of returning all nodes addresses from all hosting NEs at once within the selected NE maintenance partition, either the provisioning capable ones or the call processing capable ones. The requesting authority making this HTTP request has a major impact over the list of the returned nodes: only those sharing the same NE maintenance partition from the requester will be returned. A requesting entity (the one from which originates the HTTP request) should only be interested in the nodes pertaining to his NE maintenance partition. However, if the requester's NE maintenance partition cannot be determined, then the returned list will contain the addresses as if the network was partition-less based.

This command includes the following optional parameters:

1) *callPRequest,* which specifies the list of servers of interest and indicates whether the list of call processing capable servers or the list of provisioning-capable servers is returned in the portal API response. If omitted, the list of provisioning-capable servers is returned. In a hosting model composed only of Application Servers in primary-secondary mode, the portal API response is the same whether the request is for the list of call processing servers or provisioning servers. Therefore, in this case, the parameter value is irrelevant.

2) *private* (deprecated), which is a synonym for *callPRequest*. Any of the 2 parameters set to "true" forces call processing-capable servers to be returned in the response.

3) *broadworksOnly*, which specifies the list of servers of type "broadworks" only. If omitted, all serving hosting NE server nodes are returned.

4) *clusterMode,* which formats the output of the servlet (as shown in the example in the next section). When the returned hosting NE is in an HSS hosting model, and the parameter *clusterMode* is true, each address returned is followed by an empty address. This parameter is supported in legacy systems and shall no longer be used.

5) *disregardPartitions,* which prevents the requester from being the default authority that selects the self-assigned NE maintenance partition, later determining the returning hosting NE(s) node(s) address(es) pertaining to this NE maintenance partition. Setting this parameter to true will work as if the system had no partitions. Under normal conditions, a requesting entity (that is, the requester) should only be interested

in the nodes pertaining to their NE maintenance partition, and as result, this parameter should not be used.

6) *enterpriseName*, an optional parameter that returns all node addresses serving an enterprise. Note that if an enterprise is served via its groups, and one or more groups are served by a different NE maintenance partition than the one serving the requesting entity, only the nodes sharing the same NE maintenance partitions will be returned.

NOTE: Using the parameter *enterpriseName* with *disregardPartitions* set to "true" will guaranty to return all node addresses serving the enterprise. However, there is no way to know which addresses are part of the NE maintenance partition shared with the requester. In other words, there might be addresses in the response that the requester will not be able deal with as those might be in a different NE maintenance partition.

In the event that the NE maintenance partition of the requester cannot be determined, it shall be impossible to return a list of all node addresses, unless the parameter *disregardPartitions* is set to "true".

4.7.1 Command Example – All Provisioning Capable Servers

```
HTTP GET
http://mtlns03/servlet/GetAllHostingNeNodeAddresses?callPRequest=false

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.PortalRequest
 isPrivate="false"
 url=""
>
  <addressArray>
    <string value="dualrouting.broadsoft.com"></string>
    <string value="access.broadsoft.com"></string>
    <string value="169.254.61.198"></string>
    <string value="dualrouting2.broadsoft.com"></string>
    <string value="access2.broadsoft.com"></string>
    <string value="169.254.61.199"></string>
    <string value="dualrouting2.other.com"></string>
    <string value="access2.other.com"></string>
    <string value="169.254.61.200"></string>
  </addressArray>
</com.broadsoft.protocols.nsportal.PortalRequest>
```

4.7.2 Command Example – All Call Processing Capable Servers

```
HTTP GET
http://mtlns03/servlet/GetAllHostingNeNodeAddresses?callPRequest=true

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.PortalRequest
isPrivate="true"
 url=""
```

```
>
  <addressArray>
    <string value="dualrouting.broadsoft.com"></string>
    <string value="signaling.broadsoft.com"></string>
    <string value="192.168.8.215"></string>
    <string value="dualrouting2.broadsoft.com"></string>
    <string value="signaling2.broadsoft.com"></string>
    <string value="192.168.8.216"></string>
    <string value="dualrouting2.other.com"></string>
    <string value="signaling2.other.com"></string>
    <string value="192.168.8.217"></string>

  </addressArray>
</com.broadsoft.protocols.nsportal.PortalRequest>
```

4.7.3 Command Example – All Provisioning Capable Servers in an HSS HostingModel with clusterMode=true

```
HTTP GET
http://mtlns03/servlet/GetAllHostingNeNodeAddresses?clusterMode=true

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.PortalRequest
 isPrivate="false"
 url=""
>
  <addressArray>
    <string value="dualrouting.broadsoft.com"></string>
    <string value=""> </string>
    <string value="access.broadsoft.com"></string>
    <string value=""> </string>
    <string value="169.254.61.198"></string>
    <string value=""> </string>
    <string value="dualrouting2.broadsoft.com"></string>
    <string value=""> </string>
    <string value="access2.broadsoft.com"></string>
    <string value=""> </string>
    <string value="169.254.61.199"></string>
    <string value=""> </string>
    <string value="dualrouting2.other.com"></string>
    <string value=""> </string>
    <string value="access2.other.com"></string>
    <string value=""> </string>
    <string value="169.254.61.200"></string>
    <string value=""> </string>
  </addressArray>
</com.broadsoft.protocols.nsportal.PortalRequest>
```

4.8 GetDeviceFileServingAS

This command returns the hosting NE server(s) where a device file can be located. The command takes the following parameters:

- *URL (optional)*
- *MAC (optional)*
- *Username (optional)*

- *File format unique ID (optional)*
- *File name (optional)*

This command may return an intermediate response requesting more information, such as the MAC address or the user name. When this occurs, the following piece of information is carried by the response:

- File format unique ID
- *MAC HTTP* header and MAC pattern when requesting the MAC address
- Challenge (basic or digest) when requesting the user name

The file format unique ID uniquely identifies a device file format on the Network Server. This ID is used to speed up the URL mapping for the second request for the same URL.

The final response returns the address and the version of all Cisco BroadWorks Application or Provisioning Server clusters where the device file may be located. Among the IP addresses defined for a specific hosting NE node, the Network Server chooses the first "Access" address. If none is present, it takes the first "DualRouting" address. Otherwise, none is returned. In a hosting model composed only of Application Servers in primary-secondary mode, the primary and secondary addresses are returned with the version, and not only the address with the version.

When the device file hosting NE server lookup is disabled on the Network Server or the URL is not specified in the request, the response returned by the *GetDeviceFileServingAS* contains all BroadWorks Application or Provisioning Servers. An indication that the lookup is disabled is also included. A client, for example, the Device Management web application on the Xtended Services Platform can then decide to cache the list of Application Servers to speed up subsequent requests.

In a hosting model composed of call processing servers and provisioning servers, the portal API response always returns the provisioning servers.

4.8.1 Command Example – No Access Device File Format Indicated

```
HTTP GET
http://mtlns03/servlet/GetDeviceFileServingAS

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.DeviceFileRequest
 url="/polycom/firmware"
 deviceFileAsLookupEnabled = "false"
 needMoreInfo="false"
>
  <clusterArray>
    <cluster hostingNe="cluster1" primary="as1" secondary="as2"
version="AS_Rel_17.0_1.200"></cluster>
    <cluster hostingNe="cluster2" primary="as3" secondary="as4"
version="AS_Rel_17.0_1.200"></cluster>
    <cluster hostingNe="cluster3" primary="as5" secondary="as6"
version="AS_Rel_17.0_1.200"></cluster>
    <cluster hostingNe="cluster4" primary="as7" secondary="as8"
version="AS_Rel_17.0_1.200"></cluster>

  </clusterArray >
</com.broadsoft.protocols.nsportal.PortalRequest>
```

4.8.2 Command Example – Device Type Static File, Application Server Location Lookup Disabled on Network Server

```
HTTP GET
http://mtlns03/servlet/GetDeviceFileServingAS?url=/polycom/firmware

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.DeviceFileRequest
 url="/polycom/firmware"
 deviceFileAsLookupEnabled = "false"
 needMoreInfo="false"
>
  <clusterArray>
    <cluster hostingNe="cluster1" primary="as1" secondary="as2"
version="AS_Rel_17.0_1.200"></cluster>
    <cluster hostingNe="cluster2" primary="as3" secondary="as4"
version="AS_Rel_17.0_1.200"></cluster>
    <cluster hostingNe="cluster3" primary="as5" secondary="as6"
version="AS_Rel_17.0_1.200"></cluster>
    <cluster hostingNe="cluster4" primary="as7" secondary="as8"
version="AS_Rel_17.0_1.200"></cluster>

  </clusterArray >
</com.broadsoft.protocols.nsportal.PortalRequest>
```

4.8.3 Command Example – Device Type Static File Hosted on Multiple Application Servers

```
HTTP GET
http://mtlns03/servlet/GetDeviceFileServingAS?url=/polycom/firmware

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.DeviceFileRequest
 url="/polycom/firmware"
 deviceFileAsLookupEnabled = "true"
 needMoreInfo="false"
>
  <clusterArray>
    <cluster hostingNe="cluster1" primary="as1" secondary="as2"
version="AS_Rel_17.0_1.200"></cluster>
    <cluster hostingNe="cluster2" primary="as3" secondary="as4"
version="AS_Rel_17.0_1.200"></cluster>
    <cluster hostingNe="cluster3" primary="as5" secondary="as6"
version="AS_Rel_17.0_1.200"></cluster>
  </clusterArray>
</com.broadsoft.protocols.nsportal.DeviceFileRequest >
```

4.8.4 Command Example – Device File Identified with MAC Address, MAC Part of URL

```
HTTP GET
http://mtlns03/servlet/GetDeviceFileServingAS?url=/polycom/ABCDABCDABCD.cfg

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.DeviceFileRequest
 url="/polycom/ABCDABCDABCD.cfg"
 mac=”abcdabcdabcd”
 deviceFileAsLookupEnabled = "true"
 needMoreInfo="false"
>
  <clusterArray>
    <cluster hostingNe="cluster1" primary="as1" secondary="as2"
version="AS_Rel_17.0_1.200"></cluster>
  </clusterArray>
</com.broadsoft.protocols.nsportal.DeviceFileRequest >
```

4.8.5 Command Example – Device File Identified with MAC Address, MAC Part of HTTP Header

```
HTTP GET
http://mtlns03/servlet/GetDeviceFileServingAS?url=/polycom/device.cfg

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.DeviceFileRequest
 url="/polycom/device.cfg"
 deviceFileAsLookupEnabled = "true"
 needMoreInfo="true"
 fileFormatId="123456"
 macFormatInNonRequestURI="User-Agent: device (.*)"
>
  <clusterArray/>
</com.broadsoft.protocols.nsportal.DeviceFileRequest>

HTTP GET
http://mtlns03/servlet/GetDeviceFileServingAS?url=/polycom/device.cfg&
fileFormatId=123456&mac=ABCDABCDABCD

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.DeviceFileRequest
 url="/polycom/device.cfg"
 mac="ABCDABCDABCD"
 fileFormatId="123456"
 deviceFileAsLookupEnabled = "true"
 needMoreInfo="false"
>
  <clusterArray>
    <cluster hostingNe="cluster1" primary="as1" secondary="as2"
version="AS_Rel_17.0_1.200"></cluster>
  </clusterArray>
</com.broadsoft.protocols.nsportal.DeviceFileRequest >
```

4.8.6 Command Example – Device File Identified with MAC Address, MAC passed as parameter with fileName

```
HTTP GET
http://mtlns03/servlet/GetDeviceFileServingAS?url=/polycom/device.cfg&ma
c=ABCDABCDABCD&fileName=device.cfg

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.DeviceFileRequest
 url="/polycom/device.cfg"
 deviceFileAsLookupEnabled = "true"
 needMoreInfo="false"
 fileName="device.cfg"
 mac="ABCDABCDABCD"
>

  <clusterArray>
    <cluster hostingNe="cluster1" primary="as1" secondary="as2"
version="AS_Rel_17.0_1.200"></cluster>
  </clusterArray>
</com.broadsoft.protocols.nsportal.DeviceFileRequest >
```

4.8.7 Command Example – Device File Identified with User Name

```
HTTP GET
http://mtlns03/servlet/GetDeviceFileServingAS?url=/polycom/device.cfg

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.DeviceFileRequest
 url="/polycom/device.cfg"
 deviceFileAsLookupEnabled = "true"
 needMoreInfo="true"
 fileFormatId="123456"
 challenge="digest"
>
  <clusterArray/>
</com.broadsoft.protocols.nsportal.DeviceFileRequest >

HTTP GET
http://mtlns03/servlet/GetDeviceFileServingAS?url=/polycom/device.cfg&
fileFormatId=123456&username=admin

HTTP 200 OK text/html
<?xml version="1.0" encoding="UTF-8"?>

<com.broadsoft.protocols.nsportal.DeviceFileRequest
 url="/polycom/device.cfg"
 username="admin"
 fileFormatId="123456"
 deviceFileAsLookupEnabled = "true"
 needMoreInfo="false"
>
  <clusterArray>
    <cluster hostingNe="cluster1" primary="as1" secondary="as2"
```

```
version="AS_Rel_17.0_1.200"></cluster>
  </clusterArray>
</com.broadsoft.protocols.nsportal.DeviceFileRequest >
```

5 Validation Requirements

The portal API is designed as a machine-to-machine interface with the validation done before the interface is activated. Data input through this interface, which does not meet the validation requirements, can have unpredictable results within the system. The following table specifies the data requirements for fields used in this interface.

URL Parameter	Maximum Length	Validation Rules
Dn	21	Only characters 0 through 9 are allowed. The number is an E.164 number. A leading + is also allowed.
url	186	Must be a valid URL, with at most one <@> character.
private	Only true or false are allowed. The default is "false".	Public addresses are also known as "access" addresses, while private addresses are also known as "signaling" addresses. "DualRouting" addresses are public and private at the same time. If an External Web Server is used on the Application Server, the public address is the server name of the External Web Server, while the private address is the server name of the collocated Web Server.
enterpriseName	32	Any characters except %, ", ', \ and the characters sequence **::** are valid. The enterprise name must start with a letter or a digit.
userId	50	Must be an existing administrator user ID on the Network Server.
password	20	The password for the administrator with the above user ID.
token	No maximum length is enforced.	After preparation, the token is used as a parameter in the redirection URL and therefore, must satisfy validation rules for *Common Gateway Interface (CGI)* parameters.
address	255	An existing hosting NE server address. Must be a valid IP address or host name.
version	30	The format of this field is as follows: <type>_<release>_<id>, where: ▪ "type" is either Application Server (AS), Execution Server (XS), Provisioning Server (PS), or Web Server (WS), ▪ "release" is the major release number (for example, Rel_14.0), and "id" is the minor release number (for example, 1.122).
webImpactingPatches	2048	▪ List of semi-comma separated patch numbers that affect the communication between a hosting NE server and a Web Server.
url (getDeviceFileServingAS)	384	Must be a valid URL. If this format is not respected, the entry is not found.
mac	12	Only characters 0 through 9, a through f, and A through F are allowed. If this format is not respected, the entry is not found.

URL Parameter	Maximum Length	Validation Rules
username	161	Must be a valid URL, with at most one <@> character. If this format is not respected, the entry is not found.
fileFormatId	No maximum length is enforced.	Should correspond to the *fileFormatId* passed back by the Network Server. If the *fileFormatId* does not exist, it is not used.
fileName	No maximum length is enforced.	Must be a valid filename.
clusterMode	Only true or false are allowed. The default is "false".	Any value other than "true" or "false" is interpreted as "false". In an HSS hosting model, an empty string value is inserted between each returned address.
callPRequest	True or false	Any value other than "true" or "false" is interpreted as "false". Returns either the call processing capable servers or the provisioning capable servers.