6MSPCG0002-29A
May 2015

Configuration Guide

Using APIs with n-Command MSP

This configuration guide describes the use of an application programming interface (API) with ADTRAN's n-Command MSP products. The API can be used to create custom applications to configure, manage, and monitor devices through n-Command MSP. This guide includes an overview of the API, parameters for use of the API, and sample API call configurations.

This guide includes the following sections:

n-Command MSP API Overview

The n-Command MSP API is a web-based service used to configure, manage, and monitor operations of the n-Command MSP system and any managed devices. The API is used to create custom applications to interface with the n-Command MSP system. The API provides a set of Hypertext Transfer Protocol (HTTP) requests that can be used between ADTRAN and third-party products to manage and configure those products.

The n-Command MSP API is a resource-based representational state transfer (RESTful) API that uses HTTP requests sent to uniform resource locators (URLs). Hypertext Transfer Protocol Secure (HTTPS) should be used for API calls to prevent malevolent attackers from snooping the traffic.

The n-Command MSP API supports extensible markup language (XML) response formats. The response type is determined by the parameters in the HTTP Accept header of the request. For example, to specify XML as the response type, **application/xml** is placed in the header.

API requests use the HTTP methods GET and POST to perform actions on the specified n-Command MSP resource. The HTTP methods perform the following operations:

- GET Used to read resources,
- POST Used to create resources.

These operations are performed on URLs composed of the n-Command MSP server IP address, universal resource identifier (URI) for the resource on which the operation will take place, and a query parameter identifying the name of the domain in which to perform the operation. Care must be taken to encode all components of the URL as described in RFC 1738 if the URL contains characters outside of the US_ASCII character set or any reserved characters.

API Request Parameters

For correct API functionality, additional parameters must be added to the HTTP header of the API request. When using the n-Command MSP API, every request requires a valid user name and password. Every request is autonomous and not dependent on any previous requests. ADTRAN recommends using POST commands rather than GET commands. GET commands pass parameters in the URI which are often logged, exposing the user name and password.

GET Command URI

http://*{server}***/api/device/***{id}***?username=***{username}***&password=***{password}*

POST Command URI

http://*{server}***/api/device/***{id}*

POST Command Parameters

{username}	Name of the user.
{password}	User password.

Hardware and Software Requirements and Limitations

The API described in this document is supported on n-Command MSP version 6.1 or later.

This guide assumes you already have a basic general understanding of n-Command MSP and API functionality.

Because the n-Command MSP API uses HTTP verbs for its functionality, standard HTTP error codes apply.

Detailed n-Command MSP API Parameters

The n-Command MSP API consists of parameters and variables to compose your POST/GET request. The following table provides a detailed list of the available API parameters. Included in this list are a number of variables identified by *{ }*. These variables can be learned by issuing a POST/GET command that will report the required information. For example, to learn the device ID to force check-in of a specific device, use the API parameter **/devices**. Record the ID of the device to force check-in and issue the **/devices/***{id}***/checkin** where *{id}* is replaced with the actual device ID.

API	Description	Action
/devices	Lists all the devices.	POST/GET
/devices/installConfig	Creates an install configuration job on the specified device.	POST
{serialnumber}	The serial number of the device.	
{file}	The configuration file.	
Returns the job ID if successful or error information upon failure.		
/devices/groups	Lists all the device groups.	POST/GET
/devices/types	Lists all the device types.	POST/GET
/device/*{id}*	Displays the details of a device. The *{id}* is the device identification.	POST/GET
/device/*{id}***/checkin**	Forces check-in of a device. The *{id}* is the device identification.	POST/GET
/firmwares/new	Uploads a new firmware file to the device.	POST
{fwFile}	The filename to be uploaded.	POST
/jobs	Lists all jobs (active, pending, canceled, failed).	POST/GET
/scheduledJobs	Lists all the scheduled jobs (defined jobs). A scheduled job could have multiple jobs associated if it is a recurring job.	POST/GET
/scheduledJob/*{id}*	The list of jobs tied to the schedule (job definition). The *{id}* is the schedule identification.	POST/GET
/scheduledJob/*{id}***/delete**	Deletes the schedule (job definition). The *{id}* is the schedule identification.	POST/GET

API	Description	Action
/scheduledJobs/new	Schedules a new job.	POST
{name}	The name of the job.	
{description}	The job description (optional).	
{config}	The configuration commands to run on the device (for a push config job).	
{reboot}	Set if the box needs to be rebooted.	
{grpId}	The device group ID of the firmware (for a firmware upgrade job).	
{fwId}	The firmware file ID (for a firmware upgrade job).	
{devices}	Comma separated IDs of the devices.	
{duration}	The duration of the job (in minutes), defaults to 180 minutes if not specified.	
{startTime}	The start time for the job. Defaults to the time the job was created if not specified. The sample format for the time is **2014-08-20T17:00:00-00:00** for August 20, 2014, 12 p.m. Currently there is no validation to check the format.	

API Examples

The following examples show the API command URL, request, and results as they may appear. Results will vary depending on the specific devices and commands that are issued.

The following examples are provided:

- *Using the API to Get Details About a Specific Device on page 5*
- *Using the API to List Devices Registered with n-Command MSP Server on page 6*
- *Using the API to Create a Scheduled Job on the n-Command MSP Server on page 8*

Using the API to Get Details About a Specific Device

The following API lists information about a specific device registered with the n-Command MSP server.

Command URL

http://*{server}*/api/device/*{id}*

Example Request

http://*{server}*/api/device/15

Result

```
<?xml version=”1.0”?>
<device id="15">
<name>4.10</name>
<systemName>4.10</systemName>
<ip>10.14.4.10</ip>
<dns>10.14.4.10</dns>
<description>NetVanta 1234, Version: R11.5.0, Date: Thu Feb 24 07:34:54 2015</description>
<contact></contact>
<firmware>R11.5.0</firmware>
<hardwareVersion>UNKNOWN</hardwareVersion>
<primaryFirmware>NONVOL:/N123Xv2A-R11-5-0.biz</primaryFirmware>
<backupFirmware></backupFirmware>
<uptime>582405</uptime>
<location></location>
<lastUpdate>2014-08-03T10:09:06.621-05:00</lastUpdate>
<insertDate>2014-07-27T12:16:38.558-05:00</insertDate>
<serial>NV1234DVT007</serial>
<partNumber>1700594G1</partNumber>
<executingImage>Unknown</executingImage>
<availableNonvol>2109632</availableNonvol>
<totalNonvol>6697320</totalNonvol>
<availableCompactFlash>0</availableCompactFlash>
<totalCompactFlash>-1</totalCompactFlash>
<availableUSB0>0</availableUSB0>
<totalUSB0>-1</totalUSB0>
<primaryImageSize>4575502</primaryImageSize>
<backupImageSize>0</backupImageSize>
<lastBackup>2015-02-03T10:09:09.094-05:00</lastBackup>
<contactURL>10.14.1.205/al/DiscoveryProcessor?action=devinfo</contactURL>
<contactInterval>300</contactInterval>
<nextContact>2015-02-03T10:14:06.621-05:00</nextContact>
<platform>NetVanta 1234</platform>
<nCommandSupported>true</nCommandSupported>
<nCommandOn>true</nCommandOn>
<availableHeap>42888176</availableHeap>
```

```
<totalHeap>47840240</totalHeap>
<percentHeapUsed>10.351253</percentHeapUsed>
<heapChange>0.0</heapChange>
<systemLoadCurrent>31</systemLoadCurrent>
<systemLoad1MinAvg>26</systemLoad1MinAvg>
<systemLoad5MinAvg>27</systemLoad5MinAvg>
<systemLoadMinimum>0</systemLoadMinimum>
<systemLoadMaximum>100</systemLoadMaximum>
<vqmSupport>false</vqmSupport>
<unsavedConfig>true</unsavedConfig>
<secureConnection>false</secureConnection>
<lateContactingServer>false</lateContactingServer>
<customerDefined1></customerDefined1>
<customerDefined2></customerDefined2>
<customerDefined3></customerDefined3>
<customerDefined4></customerDefined4>
<customerDefined5></customerDefined5>
<type>1234</type>
</device>
```

Using the API to List Devices Registered with n-Command MSP Server

The following API example lists all devices registered with the MSP server, sorted by device name. The base URL returns the first 25 devices of an unfiltered list, sorted by device name. Additional parameters can be added to the base URL to provide paging, sorting, and filtering options.

Command URL

http://*{server}*/api/devices?page={page}

Arguments

Paging - Paging is provided with a combination of two fields; a **page** field and a **page_size** field. If either are omitted, the page defaults to **1** and **page_size** defaults to **25**.

Sorting - Sorting is provided by passing a comma delineated list of fields to the **sort_by** parameter. Optionally, a matching comma delineated list of **ASC/DESC** can be sent to **sort_how** which is paired with its corresponding **sort_by** element. A missing **sort_how** is sorted ascending (ASC).

Filtering - Appending field/value pairs to the URL results in a filtered list. Filtering can be used to search the device list for a specific device, for example, by filtering for an IP address or device name. Field Names mirrors the returned fields. The API is be aware of saved filters.

Example Request

http://*{server}*/api/devices
http://*{server}*/api/devices?page=2&sort_by=ip,type
http://*{server}*/api/devices?name=2.103
http://*{server}*/api/devices?page=2&ip=205

Result

```
<?xml version=”1.0”?>
<devices page="1" page_size="2" total="29">
<device id="2">
<name>ethel</name>
<ip>10.14.2.81<ip>
<serial>LBADTN0727AF262</serial>
<type>916 Gen2</type>
<dateAdded>2012-07-27T12:09:39.184-05:00</dateAdded>
<nextContact>2012-08-02T15:52:02.123-05:00</nextContact>
<location></location>
<dnsName>10.14.2.81</dnsName>
<firmwareVersion>R11.5.0.E</firmwareVersion>
<primaryFirmware>NONVOL:/T900G2A-R11-5-0-E.biz</primaryFirmware>
<backupfirmware>NONVOL:/T900G2A-R11-4-0-E.biz</backupfirmware>
<uptime>1709453</uptime>
<contact></contact>
<customerDefined1></customerDefined1>
<customerDefined2></customerDefined2>
<customerDefined3></customerDefined3>
<customerDefined4></customerDefined4>
<customerDefined5></customerDefined5>
</device>
<device id="4">
<name>gate2acl</name>
<ip>10.14.1.98<ip>
<serial>LBADTN0542AC125</serial>
<type>4305 Enhanced</type>
<dateAdded>2012-07-27T12:10:29.714-05:00</dateAdded>
<nextContact>2015-02-02T15:54:39.723-05:00</nextContact>
<location>NMS Lab</location>
<dnsName>10.14.1.98</dnsName>
<firmwareVersion>R11.5.0.E</firmwareVersion>
<primaryFirmware>NONVOL:/NV4305A-R11-5-0-E.biz</primaryFirmware>
<backupfirmware></backupfirmware>
<uptime>1708783</uptime>
<contact>barney</contact>
<customerDefined1></customerDefined1>
<customerDefined2></customerDefined2>
<customerDefined3></customerDefined3>
<customerDefined4></customerDefined4>
<customerDefined5></customerDefined5>
</device>
</devices>
```

Using the API to Create a Scheduled Job on the n-Command MSP Server

The following API example creates a job to be executed on the server. The job consists of two potential tasks, pushing a configuration and rebooting the device. The reboot task is always scheduled last.

Command URL

http://{server}**/api/jobs/new**

Arguments

{name} Name of the job.

{description} Job description.

{startTime} Time to start job using W3C UTC format: YYYY-MM-DDThh:mmTZD.

{devices} A comma delineated list of device IDs to target.

{config} Configuration commands.

{reboot} True or False flag to reboot the device after the task is completed.

Example Request

http://*{server}***/api/jobs/new**

Result:

```
<?xml version=”1.0”?>
<job id=”####” />
```