Common Platform Services API Guide

Ivanti Neurons for MDM 94 and Ivanti EPMM 11.12.0.0

November 2023

Contents

Revision history

Date	Description
November 16, 2023	Updated for Ivanti EPMM 11.12.0.0.
August 21, 2023	Updated for Ivanti EPMM 11.11.0.0.
April 26, 2023	Added a sample request and deepened the documentation for the call, "Get device registration URI" on page 103.
November 7, 2022	Re-branded "Cloud" as "Ivanti Neurons for MDM," and "Core" as "Ivanti EPMM."
November 2, 2022	Added note that the CPS APIs only support usage in the default device space.
July 5, 2022	Changed instances of **MDM_IDENTIFIERS** to **MDM_IDENTIFIER** for occurrences in descriptions and examples of the **identifierType** parameter.
December 10, 2021	Added "Sign out " on page 44 event.
July 7, 2021	**Content changes for rebranding and distribution:** Product documentation has been rebranded to align with Ivanti standards and is now available on the Ivanti Product Documentation page.
January 22, 2021	Added the following iOS fields to the application object model: • adHocCodeSigned • appStoreVendable • betaApp • deviceBasedVPP • externalVersionIdentifier • updateAvailable

	• installing • validated • shortVersion • bundleVersion See "Get application inventory" on page 46 for an example response with these new fields.
October 28, 2020	Added the **managed** field to the return of "Get application inventory" on page 46.
August 31, 2020	The following calls are now supported in Ivanti EPMM 10.8.0.0, and were already supported in Ivanti Neurons for MDM: • "Get device groups associated with device list" on page 78 • "Search users by GUID" on page 91
August 3, 2020	Clarified that the direction is inbound for the port (8883) on which the messaging service listens.
July 21, 2020	Added the call, "Get device groups associated with device list" on page 78.
June 11, 2020	• Add documentation for the call, "Search users by GUID" on page 91. • Updated documentation for the call "Get devices by MDM or deviceGuid identifiers" on page 54 to describe that the call can also take the deviceGUID as a search identifier.
April 2, 2020	• Updated "Device" on page 38 object model to include deviceGuid. • Updated "Example Device Event messages" on page 26 to include deviceGuid.
July 31, 2019	• Clarified MQTT and API endpoints. • Updated Authentication methods supported for Ivanti EPMM and Ivanti Neurons for MDM to include basic auth and certificate-based authentication.

July 17, 2019	• Added ldapStandardAttributes parameter to the "Search users by user id" on page 93 call. • Added LDAP standard attributes information to the "Get user attributes metadata" on page 89 call. • Changed identifier_type parameter to identifierType for the "Get profile inventory (iOS only)" on page 50 and "Check-in for Threat Response Matrix update" on page 73, and "Get application inventory" on page 46 calls.
June 3, 2019	• Updated the API "Endpoint information" on page 31 to reflect the new format for certificate-based authenticated calls. • Updated the messaging "Endpoint information" on page 24. • Updated the "Resolved issues" on page 8 and "Limitations" on page 8 sections for the Ivanti Neurons for MDM 62 and Ivanti EPMM 10.3.0.0 version of the guide.
April 24, 2019	Added identifier_type parameter to the "Get profile inventory (iOS only)" on page 50 and "Check-in for Threat Response Matrix update" on page 73, and "Get application inventory" on page 46 calls.
April 22, 2019	Added documentation for these calls: • "Get device app metadata" on page 103 • "Get device certificate metadata" on page 104
April 10, 2019	Added documentation for the "Check-in for Threat Response Matrix update" on page 73 call.
January 2, 2018	• Added documentation for these new calls: - "Manage device settings - iOS only" on page 65 - "Get device settings - iOS only" on page 69 - "Get device location details " on page 61 • The device.report_initialized event is now supported on Ivanti EPMM for iOS and Android devices. Previously, it was only supported on Ivanti Neurons for MDM. See "Device event" on page 40.

Introduction

The Event Notification Service and Common Platform Services (CPS) API provide automated clients with a messaging push and an RPC pull interface for integration with third-party solutions, namely:

- Event Notification Service: an MQTT endpoint for notifications of activities within Ivanti Neurons for MDM and Ivanti EPMM. See "Using the Event Notification Service" on page 23.
- CPS API: an RPC (HTTP/JSON) API to retrieve detailed information of interest to integrators. The API is an RPC API using JSON over HTTP. See the API-related chapters, starting with "API characteristics" on page 31.

The Event Notification Service and CPS API comprise an integration platform between Ivanti's integration partners and the Ivanti customer base. Before this integration platform, integration partners had to write and maintain two separate integrations between Ivanti EPMM and Ivanti Neurons for MDM, and constantly poll the API for device state changes, increasing processing complexity and computational load.

The Event Notification Service notifies integration partners of events within a few seconds of event occurrence, and the CPS API provides standard responses across Ivanti EPMM and Ivanti Neurons for MDM, serving as the ideal integration platform between integration partners and the Ivanti EPMM and Ivanti Neurons for MDM customer base.

Resolved issues

This release includes the following resolved issues:

- **AW-32833:** Previously, the GET /msa/v1/cps/appcatalog/apps API returned multiple versions of the same app, whereas the App Catalog user interface showed only the latest of the two uploaded versions in the app catalog listing. This issue has been fixed.
- **AW-32746:** The WINDOWS parameter of the GET /msa/v1/cps/appcatalog/apps API now returns expected results.

Limitations

This release includes the following limitation:

- **AW-23186:** The GET /msa/v1/cps/user API call yields differing result formats on Ivanti Neurons for MDM and Ivanti EPMM for the following items:

Item	Ivanti Neurons for MDM	Ivanti EPMM

LDAP custom attribute keys	Lower case
	`objectguid`
	`userprincipalname`
	See the highlighted example in Ivanti Neurons for MDM response below.
		Camel Case
		`objectGUID`
		`userPrincipalName`
		See the highlighted example in Ivanti EPMM response below.

objectguid field	Hexadecimal:
	`\`
	`\3f\\21\\d9\\5b\\1c\\ab\\cf\\49\`
	`\81\\11\\7a\\fa\\b7\\d2\\ab\\6a`
	See the highlighted example in Ivanti Neurons for MDM response below.
		Unencoded string:
		`9a34fe08e8041c47`
		`991c5a387efff6bb`
		See the highlighted example in Ivanti EPMM response below.

Ivanti Neurons for MDM response

```
{
    "searchResults": [
        {
            "enabled": true,
            "createdAt": 1520419288985,
            "accountSource": "LDAP",
            "displayName": "cloud user",
            "emailAddress": "cloud.user@exchce.com",
            "firstName": "cloud",
            "lastName": "user",
            "userId": "cloud.user12@exchce.com",
            "userUuid": "a27c1b0e-96f6-4acc-af79-b3af881db1cd",
            "ldapCustomAttributes": {
                "objectguid":
"\\3f\\21\\d9\\5b\\1c\\ab\\cf\\49\\81\\11\\7a\\fa\\b7\\d2\\ab\\6a",
                "userprincipalname": "cloud.user12@exchce.com"
            }
        }
    ],
    "results": 1,
    "offset": 0,
    "limit": 50
}
```

Ivanti EPMM response

```
{
    "searchResults": [
        {
            "enabled": true,
            "createdAt": 1520553600000,
            "accountSource": "LDAP",
            "displayName": "testuser6771",
            "emailAddress": "testuser6771@auto8.mobileiron.com",
            "firstName": "testuser6771",
            "userId": "testuser6771",
            "userUuid": "bff6bf12-72ff-4524-97a0-064cc137f955",
            "ldapCustomAttributes": {
                "objectGUID": "9a34fe08e8041c47991c5a387efff6bb",
                "userPrincipalName": "testuser6771@auto8.mobileiron.com"
            }
        }
    ],
    "results": 1,
    "offset": 0,
    "limit": 50
}
```

Getting Started

Prerequisites

These sections describe the prerequisites for using the Event Notification Service and Common Platform Services API.

Common prerequisites

You need to have a working Ivanti EPMM or Ivanti Neurons for MDM environment with registered devices. See the Ivanti Neurons for MDM Administrator Guide and the Ivanti EPMM documentation set for information about configuring your environment to work with this integration platform.

Event Notification Service prerequisites

- Event notification requires Certificate based Authentication for each connection using client (identity) certificates issued by DigiCert, Inc. See "Authentication" below, "Certificate-based authentication workflows" on the next page,and "Example Ivanti Neurons for MDM certificate-based authentication cURL command" on page 32.
- Integrators can only subscribe to events if admin user associated with the certificate has been granted the CPS role. See "Assigning the CPS role to a user" on page 15.

CPS API prerequisites

Authentication

Ivanti supports the following authentication methods for CPS on Ivanti EPMM and Ivanti Neurons for MDM over an encrypted link between client and server using Transport Layer Security (TLS) v1.2:

- "Basic authentication" on the next page

- "Certificate-based authentication" below

Basic authentication

The credentials you use for basic authentication must correspond to a user with the required CPS role. See "Assigning the CPS role to a user" on page 15 for how to assign the CPS role to users.

Certificate-based authentication

The Common Platform Services API requires authentication for each API call, and supports certificate-based authentication using client (identity) certificates issued by DigiCert, Inc. When using certificate-based authentication, you must include the identity certificate in every API call and when establishing a connection to MQTT. See "Example Ivanti Neurons for MDM certificate-based authentication cURL command" on page 32 for how to include the certificate. Ivanti Neurons for MDM or Ivanti EPMM compares the username and email address (RFC 822) in the certificate SAN field against the registered username and email address. If these do not match, the system refuses the MQTT connection and API calls. See "Certificate-based authentication workflows" below for how to set up certificate-based authentication, and "Certificate-based authentication security" on page 15 for a discussion of how Ivanti enforces certificate security.

ⓘ The system does not renew certificates. Please keep track of your certificate expiration date and renew the certificate before expiration.

See "Certificate-based authentication workflows" below and "Example Ivanti Neurons for MDM certificate-based authentication cURL command" on page 32 for more details.

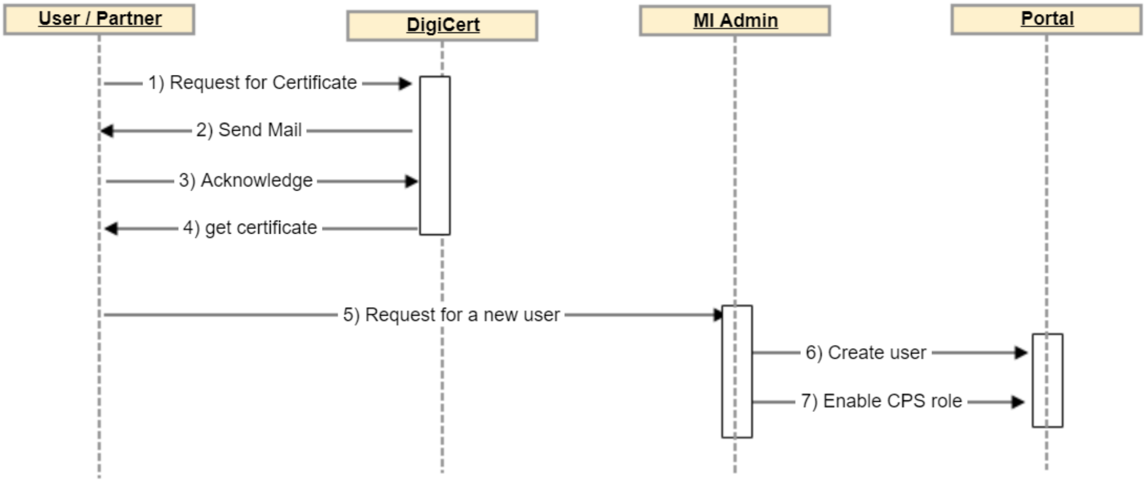
Certificate-based authentication workflows

Thus section describes the following workflows:

- Obtaining a certificate
- Using certificates in messaging
- Using certificate for API calls

Workflow: Obtaining a certificate

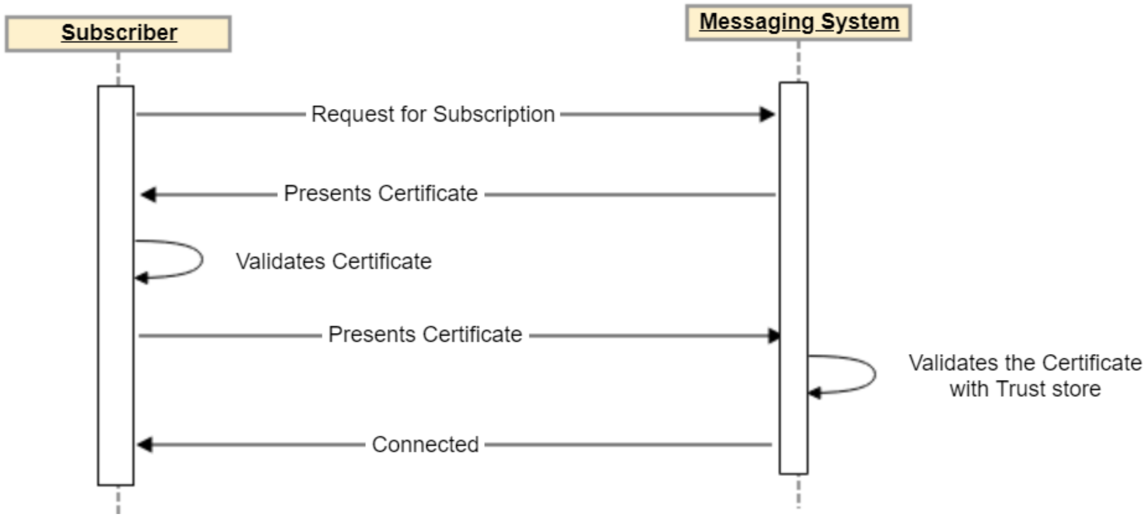
Follow this workflow to set up certificate-based authentication:



1. Request Certificate
   - User or partner requests a premium certificate from the DigiCert CA, https://www.digicert.com/secure/order.
   - The certificate issuer should be "DigiCert SHA2 Assured ID CA"
   - Signature Hash must be SHA2.
   - Common Name must be the Ivanti Neurons for MDM or Ivanti EPMM email of a user with the CPS role. See "Assigning the CPS role to a user" on page 15 for how to assign the CPS role to users.
   - Recipient Email must be a working email address of a username that is also registered with same email id in Ivanti Neurons for MDM or Ivanti EPMM.
2. Send Mail
   - DigiCert sends an email to the Recipient Email containing an URL at which to generate the certificate for the Common Name, in this case, the Ivanti Neurons for MDM or Ivanti EPMM email id associated with the CPS role.
3. Get Certificate
   - DigiCert sends a certificate in P12 format.
   - Ensure that certificate is created with the "RFC 822 Name" header.
   - The "RFC 822 Name" should be the email id of the user who will have the required CPS role. See "Assigning the CPS role to a user" on page 15 for how to assign the CPS role to users.
4. Acknowledge
   - User clicks the URL to generate the certificate.
5. Request a new user
   - User or partner requests the Ivanti admin to create a user.
6. Create user
   - The Ivanti admin creates a single user for the given email id. The email id must be the same as to which the certificate refers. See "Adding a User" in the Ivanti Neurons for MDM Administrator Guide and "Managing Users" in the Getting Started with Ivanti EPMM guide for information about creating users.
7. Enable CPS role
   - The Ivanti admin enables the CPS role for the email id. See "Assigning the CPS role to a user" on page 15 for how to assign the CPS role to users.

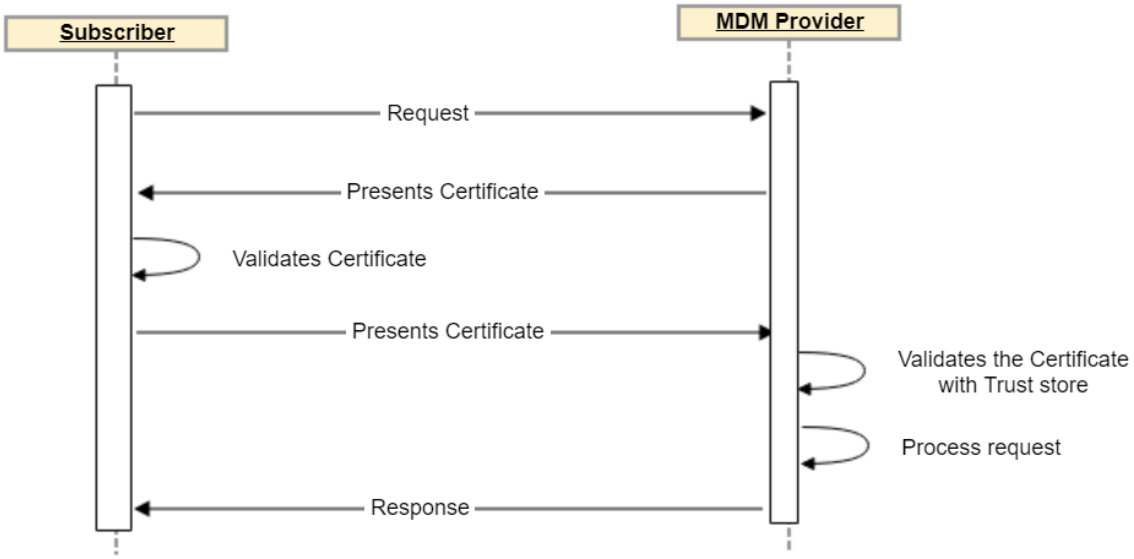
Workflow: Using certificates in messaging

Follow this workflow to use certificates in messaging:



Workflow: Using certificates for API calls

Follow this workflow to use certificates for API calls:



Certificate-based authentication security

CPS supports Certificate-based Mutual Authentication. Mutual Authentication requires that both the server and client present a certificate to prove their identity. This allows CPS to prevent unauthorized clients without valid certificates from connecting to CPS APIs and event notification service.

Certificate Validation

To be considered a valid certificate, a certificate must:

1. Chain to a Certificate Authority (CA) trusted by Ivanti Neurons for MDM and Ivanti EPMM. Currently, this is DigiCert only.
2. Be otherwise valid, for example, marked as usable for client authentication, and not expired.
3. After validating a certificate as valid, Ivanti extracts the email identity of a certificate using the SubjectAltName RFC822Name value. Ivanti enforces that only one RFC822Name value may exist in a given certificate, and rejects a certificate if there are multiple values.

X509v3 Subject Alternative Name format: email: <name>@<domain>.com

4. After extracting the email identity from the certificate, Ivanti performs a lookup for the user and ensures user has the CPS role for the Ivanti Neurons for MDM or Ivanti EPMM tenant.

Security of Certificate Procurement Process

The security of this system depends upon preventing unauthorized users from obtaining certificates that are valid for authorized users. To that end, DigiCert has the following processes and protections in place:

1. When a certificate for an email address is requested, an email is sent to that email address requesting confirmation of the certificate request. Note: it is not required that the requester has access to the email address. This allows admins to request (and pay for) certificates on behalf of users.
2. The certificate is only generated when the email recipient clicks the confirmation link. Prior to this, the certificate requester is prevented from seeing the certificate.
3. The certificate and private key are sent to the email address to which the certificate belongs. If the certificate is requested by person A who does not have access to mailbox B, then A will not be able to obtain the private key.
4. The certificate (public) is visible to the certificate requester on Digicert's website, but the private key is not. Certificate-based Mutual Authentication works by proving that each side has possession of their respective private key. Thus, by providing the private key only to the intended recipient and not the requester, the system prevents attackers from gaining unauthorized access to the APIs.

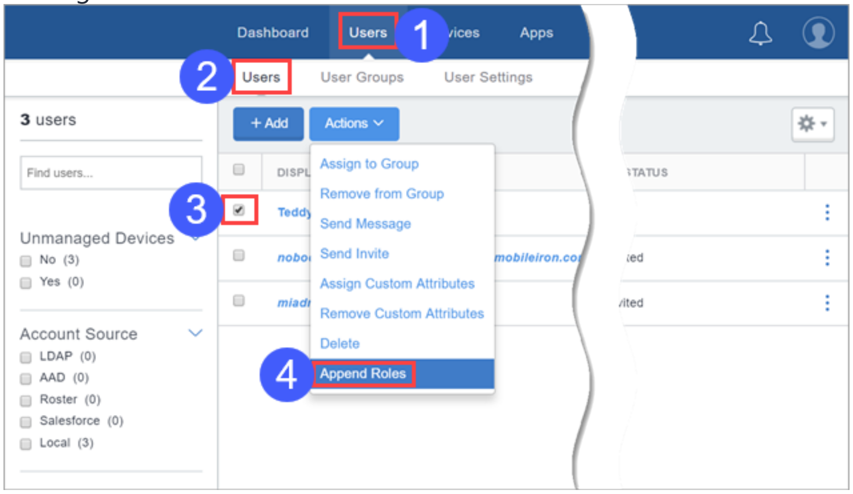
Assigning the CPS role to a user

The process differs between Ivanti Neurons for MDM and Ivanti EPMM.

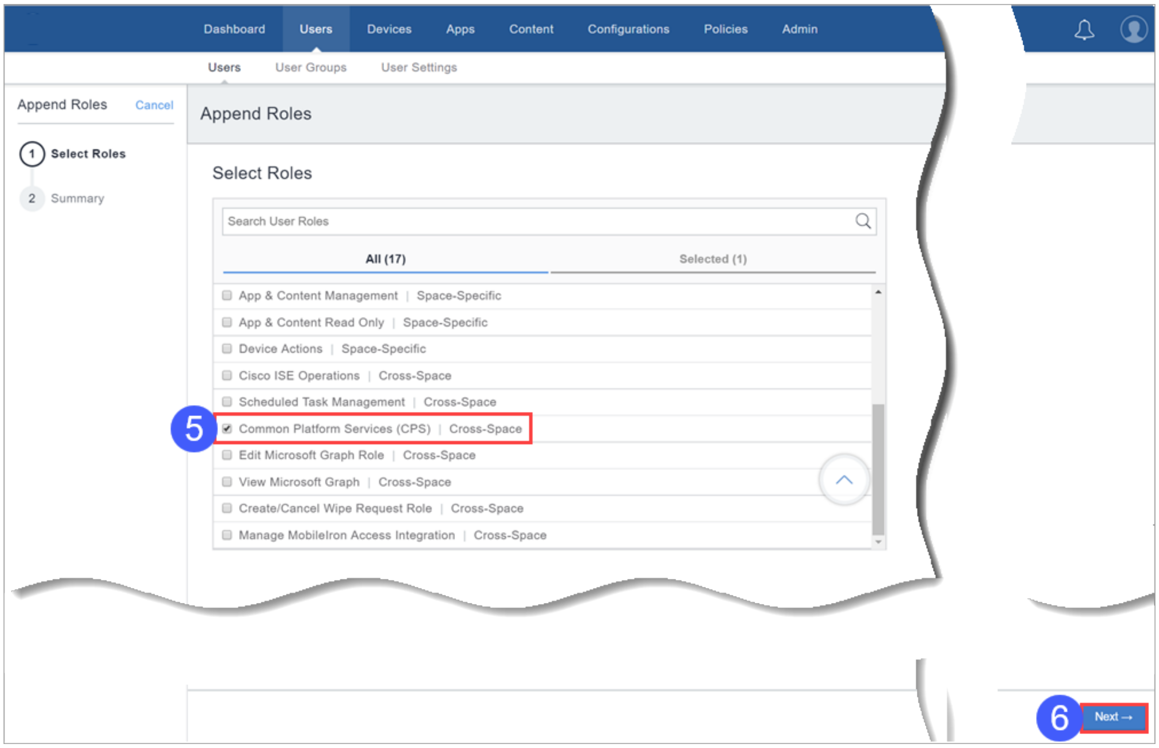
Assigning the CPS role to a Ivanti Neurons for MDM user

To assign the CPS role to a user:

1. Log in to Ivanti Neurons for MDM and then click **Users**.



2. Select **Users**.
3. Select a user.
4. Select **Append Roles**.

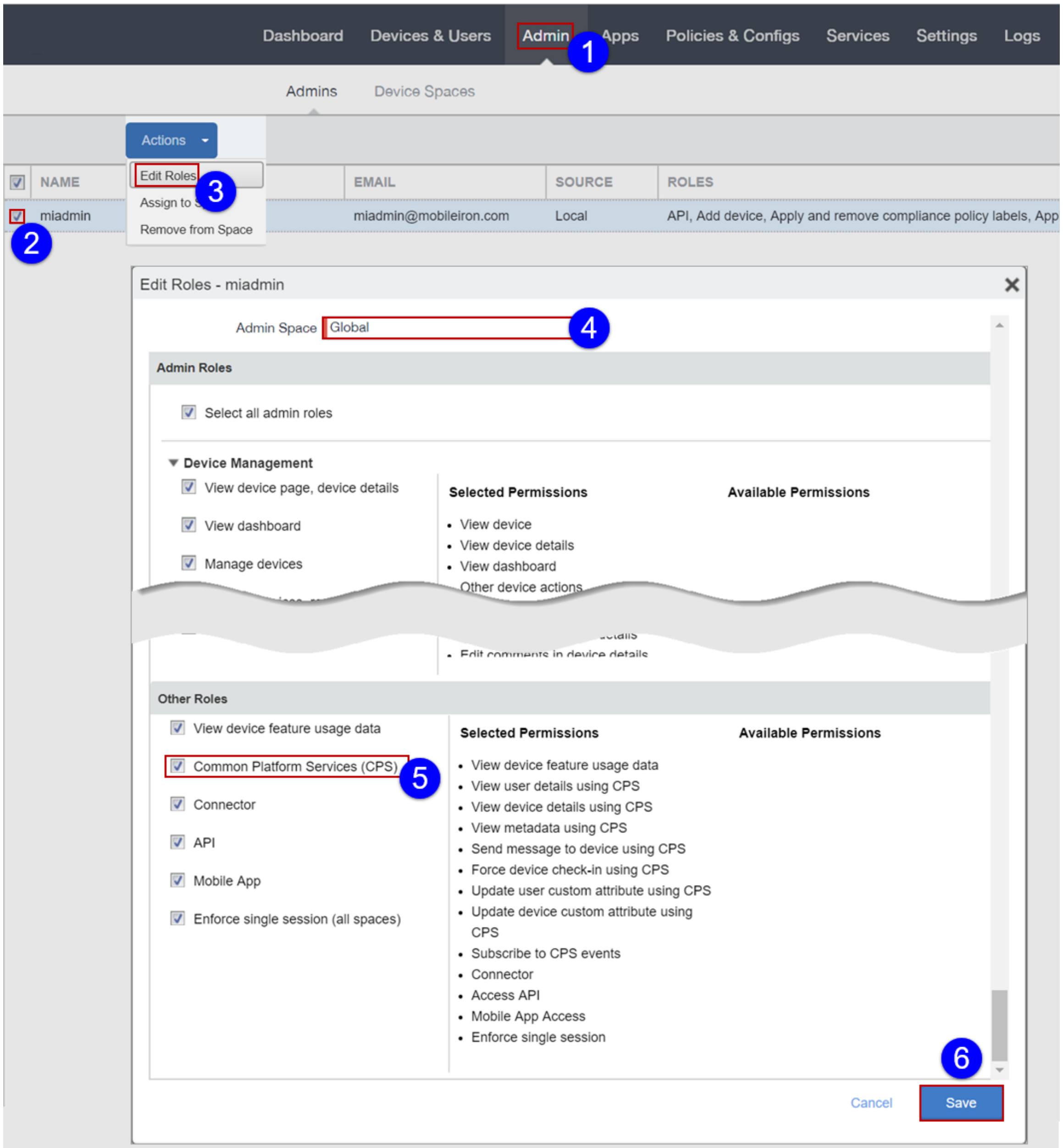

Dashboard
Users
Devices
Apps
Content
Configurations
Policies
Admin
Users
User Groups
User Settings
Append Roles
Cancel
1 Select Roles
2 Summary
Append Roles
Select Roles
Search User Roles
All (17)
Selected (1)
□ App & Content Management | Space-Specific
□ App & Content Read Only | Space-Specific
□ Device Actions | Space-Specific
□ Cisco ISE Operations | Cross-Space
□ Scheduled Task Management | Cross-Space
5
☑ Common Platform Services (CPS) | Cross-Space
□ Edit Microsoft Graph Role | Cross-Space
□ View Microsoft Graph | Cross-Space
□ Create/Cancel Wipe Request Role | Cross-Space
□ Manage MobileIron Access Integration | Cross-Space
6
Next →

5. Select the **Common Platform Services (CPS)** role.
6. Click **Next**.
7. Click **Done**.

Assigning the CPS role to a Ivanti EPMM user

To assign the CPS role to a user:

1. Log in to Ivanti EPMM and then click **Admin**.



NAME	EMAIL	SOURCE	ROLES
☑ miadmin	miadmin@mobileiron.com	Local	API, Add device, Apply and remove compliance policy labels, App

Edit Roles - miadmin

Admin Space: Global

Admin Roles

☑ Select all admin roles

Device Management

☑ View device page, device details

☑ View dashboard

☑ Manage devices

Selected Permissions

- View device
- View device details
- View dashboard
- Other device actions
- Edit comments in device details

Available Permissions

Other Roles

☑ View device feature usage data

☑ Common Platform Services (CPS)

☑ Connector

☑ API

☑ Mobile App

☑ Enforce single session (all spaces)

Selected Permissions

- View device feature usage data
- View user details using CPS
- View device details using CPS
- View metadata using CPS
- Send message to device using CPS
- Force device check-in using CPS
- Update user custom attribute using CPS
- Update device custom attribute using CPS
- Subscribe to CPS events
- Connector
- Access API
- Mobile App Access
- Enforce single session

Available Permissions

2. Select a user.
3. Select **Edit Roles**.
4. Assign the admin user to the global space.

The user will be unable to access Common Platform Services functionality if assigned to a device space other than global.

5. Select the **Common Platform Services (CPS)** role.
6. Click **Save**.

Enabling messaging

The process for enabling messaging differs between Ivanti Neurons for MDM and Ivanti EPMM. See the following sections for details.

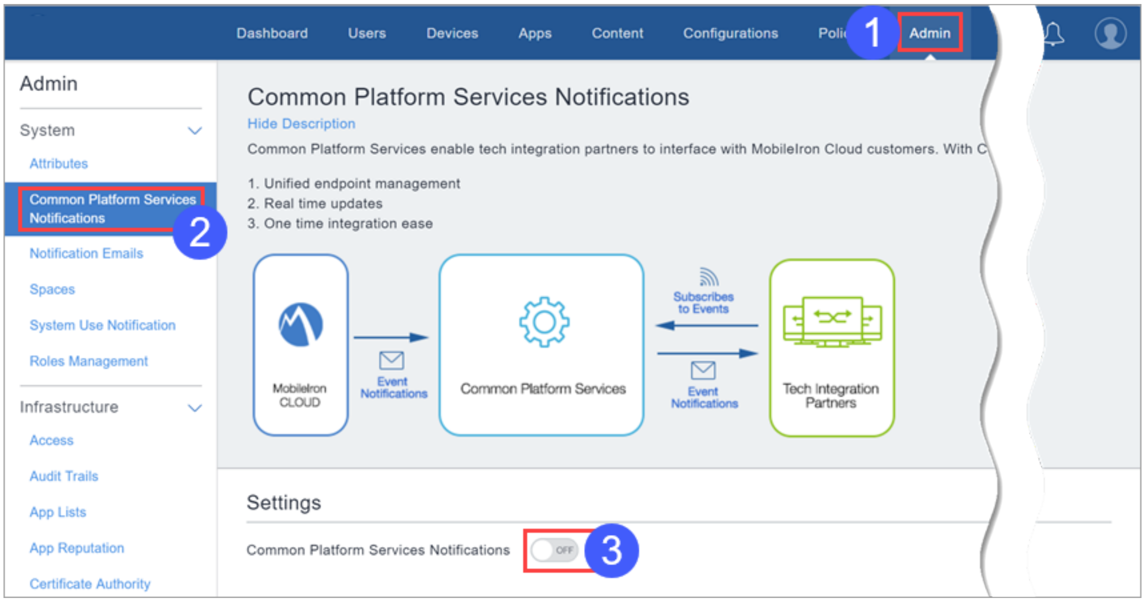
Ivanti recommends to allow HTTPS traffic on port 8443 from the corporate network, limited to Ivanti applications only. This service is intended for EPMM server management and must have strictly controlled access.

Enabling messaging on Ivanti Neurons for MDM

You can enable messaging using the Ivanti Neurons for MDM admin portal.

To enable messaging:

1. Login to Ivanti Neurons for MDM, and then select **Admin**.



Common Platform Services Notifications

Hide Description

Common Platform Services enable tech integration partners to interface with MobileIron Cloud customers. With C

1. Unified endpoint management
2. Real time updates
3. One time integration ease

Settings

Common Platform Services Notifications OFF

2. Select **Common Platform Services Notifications**.
3. Toggle the CPS notifications switch to enable or disable the service.

Enabling messaging on Ivanti EPMM

You run the Ivanti EPMM CLI program to enable messaging. This procedure invokes a message broker, enables the Event Notification Service event notification feature, restarts the Ivanti server, and restarts Apache Tomcat (on Ivanti EPMM) to reload configurations.

Considerations:

- The Messaging server listens to subscribing client requests over port 8883, and this port must be open inbound for the service to function.
- If Ivanti EPMM is running in a High Availability configuration, please enable messaging on both primary and secondary nodes.

To enable or disable messaging, run the Ivanti EPMM CLI program as shown below:

```
host:~ host$ ssh miadmin@hostname

miadmin@hostname's password:

Last login: Thu May  4 13:43:48 2017 from 10.101.10.191

************************************************************

*                   Ivanti EPMM CLI                    *

*                                                          *

*                                                          *

************************************************************

Welcome miadmin it is Thu May 4 13:49:39 UTC 2017

CORE(9.4.0.0-2388)@hostname> enable

Password:

CORE(9.4.0.0-2388)@hostname#configure terminal

Enter configuration commands, one per line.

CORE(9.4.0.0-2388)@hostname/config#activemq

Warning:Maintenance mode command.

Portal service will be stopped during this operation. Proceed? (y/n)y

Updating chkconfig...
Updating portal...

Starting ActiveMQ...

INFO: Loading '/mobileiron.com/programs/org.apache.activemq/bin/env'

INFO: Using java '/mobileiron.com/programs/com.mobileiron.platform.jre8/bin/java'

INFO: Starting - inspect logfiles specified in logging.properties and log4j.properties
to get details

INFO: pidfile created : '/mobileiron.com/programs/org.apache.activemq/data/activemq.pid'
(pid '22954')

Capturing tomcat metrics:                                   [  OK  ]

Stopping tomcat:                                            [  OK  ]

Starting tomcat: Using TOMCAT_ALLOCATION_MB=2048

Using JAVA_OPTS=-Xms128m -Xmx2048m -XX:PermSize=512m -XX:MaxPermSize=512m -
XX:+OptimizeStringConcat -server -Dvsp.branding=mobileiron -Dmi.hostname=hostname -
Dfile.encoding=utf-8 -Djavax.net.ssl.keyStore=/usr/java/default/jre/lib/security/cacerts
-Djava.security.auth.login.config=/mi/files/ldap/gsseg_jaas.conf -
Djava.security.krb5.conf=/mi/files/ldap/krb5.conf -Dspring.auth.method=form -
Dsun.net.inetaddr.ttl=60 -Dmi.tcp.port.system.manager.portal=8443 -
Djsse.enableSNIExtension=false -Dmi.version=9.4.0.0-2388 -Xms1024m -Xmx2048m -XX:-
HeapDumpOnOutOfMemoryError -XX:HeapDumpPath=/mi/files -verbose:gc -XX:+PrintGCDateStamps
-XX:+PrintGCTimeStamps -XX:+PrintGCDetails -Xloggc:/mi/tomcat/logs/gc.log -
XX:+UseConcMarkSweepGC -XX:+UseParNewGC -XX:MaxNewSize=480M -XX:NewRatio=7 -
XX:TargetSurvivorRatio=90 -XX:CMSInitiatingOccupancyFraction=60

                                                            [  OK  ]

Successfully enabled activemq

CORE(9.4.0.0-2388)@hostname/config#no activemq

Warning:Maintenance mode command.

Portal service will be stopped during this operation. Proceed? (y/n)y

Updating chkconfig...

Updating portal...
```

```
Stopping ActiveMQ...

INFO: Loading '/mobileiron.com/programs/org.apache.activemq/bin/env'

INFO: Using java '/mobileiron.com/programs/com.mobileiron.platform.jre8/bin/java'

ERROR: No or outdated process id in
'/mobileiron.com/programs/org.apache.activemq/data/activemq.pid'

INFO: Removing /mobileiron.com/programs/org.apache.activemq/data/activemq.pid

Capturing tomcat metrics:                                  [  OK  ]

Stopping tomcat:                                           [  OK  ]

Starting tomcat: Using TOMCAT_ALLOCATION_MB=2048

Using JAVA_OPTS=-Xms128m -Xmx2048m -XX:PermSize=512m -XX:MaxPermSize=512m -
XX:+OptimizeStringConcat -server -Dvsp.branding=mobileiron -Dmi.hostname=hostname -
Dfile.encoding=utf-8 -Djavax.net.ssl.keyStore=/usr/java/default/jre/lib/security/cacerts
-Djava.security.auth.login.config=/mi/files/ldap/gsseg_jaas.conf -
Djava.security.krb5.conf=/mi/files/ldap/krb5.conf -Dspring.auth.method=form -
Dsun.net.inetaddr.ttl=60 -Dmi.tcp.port.system.manager.portal=8443 -
Djsse.enableSNIExtension=false -Dmi.version=9.4.0.0-2388 -Xms1024m -Xmx2048m -XX:-
HeapDumpOnOutOfMemoryError -XX:HeapDumpPath=/mi/files -verbose:gc -XX:+PrintGCDateStamps
-XX:+PrintGCTimeStamps -XX:+PrintGCDetails -Xloggc:/mi/tomcat/logs/gc.log -
XX:+UseConcMarkSweepGC -XX:+UseParNewGC -XX:MaxNewSize=480M -XX:NewRatio=7 -
XX:TargetSurvivorRatio=90 -XX:CMSInitiatingOccupancyFraction=60

                                                           [  OK  ]

Successfully disabled activemq

CORE(9.4.0.0-2388)@hostname/config#
```

Validating Event Notification Service and CPS API functionality

To validate Event Notification Service functionality, see the section, "Sample client snippet for message subscriber" on page 27 for how to write a script that verifies that you can subscribe to events.

To validate that you can use the CPS API, run the call described in the section, " Getting event subscription metadata" on page 25.

Using the Event Notification Service

Overview

The Event Notification Service provides automated clients with an MQTT endpoint for notifications of activities within Ivanti Neurons for MDM and Ivanti EPMM.

Messages indicate:

- Device activity, such as registration, check-in, and retirement
- Device state activity, for example, transitions between compliant and non-compliant, and back
- Administrative changes

A variety of out-of-the-box MQTT implementations are available and this manual provides sample client code in the section, "Sample client snippet for message subscriber" on page 27. Messages are JSON objects, and any of the widely-available JSON implementations can de-serialize them. For more information about MQTT, visit http://mqtt.org/documentation.

TLS (Transport Layer Security) is the standard security technology for establishing an encrypted link between a server and a client. This link ensures that all data passed between the web server and browsers remain private and integral. Partner connection and communication with a message broker as a subscriber is TLS secured and encrypted.

Notes:

- MQTT protocol enabled by Ivanti EPMM for CPS events only supports TLS 1.2.
- Ivanti recommends a maximum of **fifteen** connections per tenant. The best practice is to use a single connection to connect and subscribe to all the required topics from a tenant or Ivanti EPMM.
- Every connection should have a unique MQTT client ID.
- Subscribers (MQTT/TLS clients) need to connect to broker over TLS encrypted port 8883. See "Endpoint information" on the next page for the broker URL to use for your environment.

- For Ivanti EPMM, during backup and restore on a secondary node, any cipher suite and protocol configuration changes for incoming TLS connections from Ivanti EPMM System Manager (MICS) has to be manually replicated on the secondary node because MICS configurations are not synced .
- All MQTT clients maintaining an active subscription when MQTT sends the message receive the message. The system purges the message after delivery, even if the message is consumed by only one active client that acknowledges receipt with a PUBACK command. MQTT delivers messages with QoS level 1, which means the sender can deliver the message more than once until the sender receives a PUBACK command from a receiving client.
- If no active consumers are available at the instant message is published to the messaging server, the message would be retained in the system for a maximum duration of 3 hours before it either gets consumed by a re-connecting durable subscriber or gets evacuated out of the system

Suggested messaging integration workflow

1. Obtain endpoint URLs and a login email address to make a connection. These are provided separately by your Ivanti account representative. You also need a client (identity) certificate. See "Authentication" on page 11 and "Certificate-based authentication workflows" on page 12.
2. Use the " Getting event subscription metadata" on the next page call to get list of available events and topics to which to subscribe.
3. Edit your login information into the sample client snippet for message subscriber, then execute one of the sample clients. This connects to the messaging system, registers for events, and begins printing message activity to the standard output. See "Sample client snippet for message subscriber" on page 27.
4. Register a device to generate events that will enable you to test your integration.
5. Verify that device registration and check-in actions are showing up on your console from your client.

Endpoint information

The endpoints differ between Ivanti Neurons for MDM and Ivanti EPMM.

The following table summarizes the endpoints:

Ivanti Neurons for MDM	Ivanti EPMM

`ssl://queue-<clustername>.mobileiron.com:8883`
For example, given the cluster NA1, then the endpoint address would be:
`ssl://queue-na1.mobileiron.com:8883`
	`ssl://<fully-qualified-hostname>:8883`
	For example, given the fully qualified hostname, acme.mobileiron.com, then the endpoint address would be:
	`ssl://acme.mobileiron.com:8883`

Getting event subscription metadata

This call returns available event types to which you can subscribe and the retention time for Event Notification Service events.

HTTP Method

GET

Request URI

/msa/v1/cps/event/$metadata

Example response

Ivanti Neurons for MDM

```
{
    "topics": {
        "device_profile_inventory_update": "18ccc64c-66e4-4e27-8bd6-
9acc9c8d0224/device/profile_inventory",
        "device_cert_inventory_update": "18ccc64c-66e4-4e27-8bd6-9acc9c8d0224/device/cert_
inventory",
        "device_compliant": "18ccc64c-66e4-4e27-8bd6-9acc9c8d0224/device/compliant",
        "device_check_in": "18ccc64c-66e4-4e27-8bd6-9acc9c8d0224/device/check_in",
        "device_app_inventory_update": "18ccc64c-66e4-4e27-8bd6-9acc9c8d0224/device/app_
inventory",
        "device_wiped": "18ccc64c-66e4-4e27-8bd6-9acc9c8d0224/device/wiped",
        "device_not_compliant": "18ccc64c-66e4-4e27-8bd6-9acc9c8d0224/device/not_
compliant",
        "device_enrolled": "18ccc64c-66e4-4e27-8bd6-9acc9c8d0224/device/enrolled",
        "device_retired": "18ccc64c-66e4-4e27-8bd6-9acc9c8d0224/device/retired"
    },
    "retentionPeriodMinutes": 180
}
```

Ivanti EPMM

```
{
    "topics": {
        "device_not_compliant": "device/not_compliant",
        "device_compliant": "device/compliant",
        "device_retired": "device/retired",
        "device_enrolled": "device/enrolled",
        "device_app_inventory_update": "device/app_inventory",
        "device_profile_inventory_update": "device/profile_inventory",
        "device_cert_inventory_update": "device/cert_inventory",
        "device_wiped": "device/wiped",
        "device_check_in": "device/check_in"
    },
    "retentionPeriodMinutes": 180
}
```

The key in the “topics” JSON defines the event types available and the corresponding values indicate the destination topic to which the partner should subscribe to get respective event notifications. The system retains messages for three hours and then purges them if they are not consumed and acknowledged with a PUBACK command.

Messages

There is currently one message format defined. See "Device" on page 38, and see the example messages below.

Example Device Event messages

App install

```
{
    "timestamp": 1582796173229,
    "eventType": "device.app_inventory",
    "devices": [
        {
            "compliant": true,
            "quarantined": false,
            "blocked": false,
            "compromised": false,
            "status": "ACTIVE",
            "lastCheckInTime": 1582796171062,
            "registrationTime": 1582608427870,
            "identifier": "f2e01873c6d96d87f1a34ff72e0e4143e42ee089",
            "macAddress": "74:81:14:d6:f5:b7",
            "manufacturer": "Apple Inc.",
            "model": "iPad5,3",
            "os": "IOS",
            "osVersion": "12.2",
            "serialNumber": "DMPPK6YVG5VT",
            "userId": "rc@mi.com",
            "userUuid": "5159964d-95a7-466f-9677-1b90877f045f",
            "iosUdid": "f2e01873c6d96d87f1a34ff72e0e4143e42ee089",
            "deviceGuid": "1d71e131-0487-4d1d-9a0d-89f2bd1a888d"
        }
    ]
}
```

Device retire

```
{
    "timestamp": 1582796293479,
    "eventType": "device.retired",
    "devices": [
        {
            "compliant": true,
            "quarantined": false,
            "compromised": false,
            "status": "RETIRED",
            "lastCheckInTime": 1582796291883,
            "registrationTime": 1582608427870,
            "identifier": "f2e01873c6d96d87f1a34ff72e0e4143e42ee089",
            "macAddress": "74:81:14:d6:f5:b7",
            "manufacturer": "Apple Inc.",
            "model": "iPad5,3",
            "os": "IOS",
            "osVersion": "12.2",
            "serialNumber": "DMPPK6YVG5VT",
            "userId": "rc@mi.com",
            "userUuid": "5159964d-95a7-466f-9677-1b90877f045f",
            "iosUdid": "f2e01873c6d96d87f1a34ff72e0e4143e42ee089",
            "deviceGuid": "1d71e131-0487-4d1d-9a0d-89f2bd1a888d"
        }
    ]
}
```

Sample client snippet for message subscriber

This example code snippet is for the Paho client. For more information, see https://eclipse.org/paho/clients/java/.

```
import java.io.BufferedInputStream;
import java.io.FileInputStream;
import java.security.KeyStore;
import java.security.Security;
import java.security.cert.CertificateFactory;
import java.security.cert.X509Certificate;

import javax.net.ssl.KeyManagerFactory;
import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManagerFactory;

import org.bouncycastle.jce.provider.BouncyCastleProvider;
import org.eclipse.paho.client.mqttv3.IMqttDeliveryToken;
import org.eclipse.paho.client.mqttv3.MqttCallback;
import org.eclipse.paho.client.mqttv3.MqttClient;
import org.eclipse.paho.client.mqttv3.MqttConnectOptions;
import org.eclipse.paho.client.mqttv3.MqttMessage;
import org.eclipse.paho.client.mqttv3.persist.MemoryPersistence;

public class SampleMQTTCBA implements MqttCallback {
    public static final int KEEP_ALIVE_INTERVAL = 30;

    public static void main(String[] args) {
        String topic = "45a10ad4-ada7-4698-bc2b-d8ce1e1eb5c6/device/profile_inventory";
        try {
            MqttClient client = new MqttClient("ssl://<hostname>:8883",
                    "clientId",
                    new MemoryPersistence());

            MqttConnectOptions options = getMqttConnectOptions();
            client.setCallback(new SampleMQTTCBA());

            client.connect(options);
            client.subscribe(topic, 0);

            if (client.isConnected()) {
                System.out.println("Client connected");
            }
            client.unsubscribe(topic);
            client.disconnect();
            client.close();
        } catch (Exception exception) {
            exception.printStackTrace();
        }
    }

    private static MqttConnectOptions getMqttConnectOptions() throws Exception {
        MqttConnectOptions options = new MqttConnectOptions();
        options.setKeepAliveInterval(KEEP_ALIVE_INTERVAL);
        options.setCleanSession(false);

        String caFile = "<ca-cert.pem file path>";
        String p12File = "<End user's P12 file path>";
        String p12Password = "<P12 password>";

        final SSLContext sslContext = getSslContext(caFile, p12File, p12Password);
        options.setSocketFactory(sslContext.getSocketFactory());
        return options;
    }


    private static SSLContext getSslContext(final String caCrtFile, final String p12File,
final String password) {
        Security.addProvider(new BouncyCastleProvider());
        try {
            KeyManagerFactory kmf = getKeyManagerFactory(p12File, password);
            TrustManagerFactory tmf = getTrustManagerFactory(caCrtFile);

            SSLContext context = SSLContext.getInstance("TLSv1.2");
            context.init(kmf.getKeyManagers(), tmf.getTrustManagers(), null);
            return context;
        } catch (Exception e) {
            e.printStackTrace();
            System.err.println("Could not create SSL context");
            throw new IllegalArgumentException();
        }
    }

    private static KeyManagerFactory getKeyManagerFactory(String p12File, String password)
throws Exception {
        KeyStore clientStore = KeyStore.getInstance("PKCS12");
        clientStore.load(new FileInputStream(p12File), password.toCharArray());

        KeyManagerFactory kmf = KeyManagerFactory.getInstance
(KeyManagerFactory.getDefaultAlgorithm());
        kmf.init(clientStore, password.toCharArray());
        return kmf;
    }

    private static TrustManagerFactory getTrustManagerFactory(String caCrtFile) throws
Exception {
        // load CA certificate
        X509Certificate caCert = null;

        FileInputStream fis = new FileInputStream(caCrtFile);
        BufferedInputStream bis = new BufferedInputStream(fis);
        CertificateFactory cf = CertificateFactory.getInstance("X.509");

        while (bis.available() > 0) {
            caCert = (X509Certificate)cf.generateCertificate(bis);
        }

        // CA certificate is used to authenticate server
        KeyStore caKs = KeyStore.getInstance(KeyStore.getDefaultType());
        caKs.load(null, null);
        caKs.setCertificateEntry("ca-certificate", caCert);
        TrustManagerFactory tmf = TrustManagerFactory.getInstance("X509");
        tmf.init(caKs);
        return tmf;
    }
    public void connectionLost(Throwable cause) {
        System.out.println("Demo.connectionLost()");
        cause.printStackTrace();
    }

    public void messageArrived(String topic, MqttMessage message) throws Exception {
        System.out.println("Demo.messageArrived()" + topic + "   " + message);
    }

    public void deliveryComplete(IMqttDeliveryToken token) {
        System.out.println("message id" + token.getMessageId());
    }
}
```

API characteristics

Endpoint information

The endpoints differ between Ivanti Neurons for MDM and Ivanti EPMM for certificate-based authentication, and between basic authentication and certificate authentication for Ivanti Neurons for MDM.

ⓘ The CPS APIs only support usage in the default device space.

The following table summarizes the endpoints:

Authentication	Ivanti Neurons for MDM	Ivanti EPMM

Basic	`https://<fully-qualified-hostname>/msa/v1/cps`
	For example, given the fully qualified hostname, na1.mobileiron.com, the endpoint address would be:
	`https://na1.mobileiron.com/msa/v1/cps`
		For both basic and certificate-based authentication:
		`https://<fully-qualified-hostname>/msa/v1/cps`
		For example, given the fully qualified hostname, acme.mobileiron.com, then the endpoint address would be:
		`https://acme.mobileiron.com/msa/v1/cps`

Certificate	`https://cps-<clustername>.mobileiron.com/msa/v1/cps`
	For example, given the cluster name NA1, the endpoint address would be:
	`https://cps-na1.mobileiron.com/msa/v1/cps`
	When using certificate-based authentication, you must include the identity certificate in every API call and when establishing a connection to MQTT. See "Example Ivanti Neurons for MDM certificate-based authentication cURL command" below for how to include the certificate. See "Certificate-based authentication workflows" on page 12 for how to set up certificate-based authentication.	

These endpoints listen on port 443.

Example Ivanti Neurons for MDM certificate-based authentication cURL command

```
curl -X GET 'https://cps-na1.mobileiron.com/msa/v1/cps/version'
--cert <End user's P12 file path>/<End user's P12 filename>.p12:<End user's P12 file
password>
--cert-type p12
--cacert <ca-cert.pem file path>
```

P12 in the snippet above refers to p12 authentication certificates. See "Certificate-based authentication workflows" on page 12 for how to set up certificate-based authentication.

Aggregate API rate limits per cluster

Ivanti recommends integrators manage and control the call rate of CPS APIs so as to not exceed the following maximum values of calls per minute. Aggregate API call rates per cluster that exceed the following limits may result in rate-limiting being invoked:

	Cluster					
Tenants	**NA1**	**NA2**	**NA3**	**EU1**	**AP1**	**AP2**
msa/v1/cps/device/application	300*	400	300	300	300	100
msa/v1/cps/device/mac	100	200	100	100	100	100
/msa/v1/cps/device?status=<DeviceStatus>&offset=<Integer>&limit=<Integer>&ruleId=<rule>	200	200	200	200	200	100
/msa/v1/cps/user?uid=<user-id>&offset=<offset>&limit=<limit>	100	100	100	100	100	100
/msa/v1/cps/device/forceCheckin	100	100	100	200	100	100
/msa/v1/cps/device/notification	200	200	100	200	200	200
/msa/v1/cps/device/email	200	400	200	400	200	200
Remaining APIs	400	600	600	600	200	200

* Values are calls/minute.

Batch ID recommendation

Ivanti recommends limiting to 100 the number of IDs sent with a single API call. The maximum number of IDs supported with any single API call is 200. API performance may degrade if more than 100 IDs are sent in any single API call and rate limiting may be invoked on MI Ivanti Neurons for MDM.

Pulling data

- API calls only return properties that have a value and where the parameter is supported in the data model. For the complete list of properties, see "Object models" on page 36.

- The API calls return all date fields as a long: number of milliseconds since the epoch - January 1, 1970 UTC.

Common items

The following data structures are used across multiple APIs.

List of identifiers

A list of strings, device UEM identifiers, deviceGuid identifiers, device mac addresses, user Globally, or Universally Unique identifiers.

List of devices format

```
{"
identifiers":["mdm-id-1", "mdm-id-2"]
}
```

Example list of devices

```
 {
      "compliant":true,
         "quarantined": false,
         "compromised": false,
      "status":"ACTIVE",
      "lastCheckInTime":1469176919585,
      "registrationTime":1466449519230,
      "identifier":"24f048d3ec2b1d9f97921588f114b7f1fea9da35b685750bd072b395ba716",
         "imei": "35 2342342342345 64 1"
      "macAddress":"08:d4:2b:1f:1b:2a",
      "manufacturer":"samsung",
      "model":"Nexus 10",
      "os":"ANDROID",
      "osVersion":"5.1.1",
         "serialNumber": "F123431WN123",
      "userId":"e@m.d",
      "userUuid":"96bd909c-25f8-4434-b243-470e5ebcfb6e",
      "custom attribute 1":"custom attribute value",
      "custom attribute 2":"custom attribute value"
   },
   {
      "compliant": true,
      "quarantined": false,
      "compromised": false,
      "status": "RETIRED",
      "lastCheckInTime": 1493723182655,
      "registrationTime": 1493712801136,
      "identifier": "b9a0d7d05fb4204f52264f805932526b848ca61d",
      "imei": "35 377808 345646 1",
      "manufacturer": "Apple Inc.",
      "model": "iPhone9,3",
      "os": "IOS",
      "osVersion": "10.2.1",
      "serialNumber": "F17ST4356HG7F",
      "userId": "sk@mi.com",
      "userUuid": "7d191915-28d1-444563-bec8-456",
      "iosUdid": "b9a0d7d05fb4563456f805932526b848ca61d"
   }
```

Number result

Number results are single integer responses whose meanings vary depending on the API call, for example, the count of updated entities or the response of a count API.

Example number result

```
{
    "result": 2
}
```

Update attributes

These handle both device and user attributes updates.

Update attributes request format

```
{
    "identifiers": [
        "id-1",
        "id-2"
    ],
    "attributes": {
        "key": "value"
    }
}
```

Enums

Enum	Values	Used for
AccountSource	LOCAL, LDAP	User # accountSource
DevicePlatform	ANDROID, IOS, OSX, WINDOWS, UNKNOWN	Device # os
DeviceStatus	ACTIVE, RETIRED, WIPED, WIPE_SENT, UNKNOWN, RETIRE_SENT, ENROLLMENT_PENDING	Device # status

Object models

The following sections describe the response fields you will encounter using the APIs.

Do not perform strict type checking against the API JSON responses because Ivanti may enhance the responses with additional fields in subsequent releases.

App Catalog

Field name	Description
name	Human readable name.
platformType	Type of OS platform.
version	Application version.
identifier	Application (unique) identifier.
buildNumber	Build number of the application. This is for in-house apps only.
createdAt	Time app was added.

Application

Field name	Description
identifier	Application (unique) identifier
name	Human readable name
version	Application version
managed	Is the application managed.
source	Source of the app, for example, PUBLIC.
adHocCodeSigned	true or false
appStoreVendable	true or false
betaApp	true or false
deviceBasedVPP	true or false
externalVersionIdentifier	The external version identifier, for example, 832990878.
updateAvailable	true or false
installing	true or false
validated	true or false
shortVersion	The short version, for example, 5.2.1.
bundleVersion	The bundle version, for example, 6733.

Certificate

Field name	Description
identity	Indicates the certificate is an identity certificate.
notAfter	Epoch time after which certificate expires.
notBefore	Epoch time before which certificate is not valid.
issuer	DName of the certificate issuer organization.
serialNumber	Unique identification number for the certificate.
subject	DName of the client to which the certificate belongs.
thumbprint	The thumbprint is a hexadecimal string uniquely identifying a certificate. A thumbprint algorithm calculates the thumbprint. CES accepts Secure Hash Algorithm 1 (SHA-1) thumbprints in the 40-digit hexadecimal string form without spaces.

Device

Note: imei, imsi and macAddress may not be present in all devices.

Field name	Description
compliant	"true" if the device meets the compliance rules established by the Ivanti enterprise administrator, else "false".
quarantined	"true" if the device is violating any policy with respect to quarantine action for device.
blocked	"true" if the device is violating a defined policy that has a compliance action.
compromised	"true" if the device OS is compromised as defined by rules established by the Ivanti enterprise administrator, for example, iOS device is jailbroken or an Android device is rooted.
status	Device registration state; one of: ACTIVE RETIRED WIPED WIPE_SENT UNKNOWN RETIRE_SENT ENROLLMENT_PENDING
lastCheckInTime	The timestamp of the last device check in (number of milliseconds from the epoch). This is the UEM check in time.
registrationTime	The timestamp of the device registration (number of milliseconds from the epoch).
identifier	The unique device UEM identifier for the device in question. UEM identifiers are consistent within an OS, but are not consistently formatted across OSs. This identifier is not present for MAM only and Auth only devices because MDM is required in order to provide this information. Use deviceGuid, described below, which is more reliably available.
macAddress	The device's MAC address, six groups of two hexadecimal digits separated by colons e.g. b0:65:bd:33:fe:2b. Available and valid only after the system fires the device.report_initialized event immediately after device enrollment.
manufacturer	Device manufacturer, e.g. **samsung** , **Apple Inc.** or **MicrosoftMDG**
model	Device model, e.g. **Nexus 10** , **iPad3,1** or **Lumia 950 Dual SIM**

Field name	Description
os	The devices operating system; one of: ANDROID IOS OSX WINDOWS UNKNOWN
osVersion	The OS version.
serialNumber	Device serial number.
userId	User name of the user the device belongs to.
userUuid	User unique identifier, can be used with user APIs like "Update user attributes" on page 97.
iosUdid	The UDID number for iOS device.
deviceGuid	The device GUID of the device.
customAttributes	All custom device attribute fields that have been set with a value for the device will be listed.
ownership	Indicates whether the device is company-owned or employee-owned, or the ownership type is unknown. Returns one of the following values: COMPANY, EMPLOYEE, UNKNOWN
imei	The device International Mobile Station Equipment Identity number. See http://www.imei.info/faq-what-is-IMEI/.
imsi	The IMSI number for the device.
phoneNumber	Phone number.

Device event

Field name	Description
timestamp	milliseconds since the epoch: January 1, 1970
eventType	one of the following: • device.report_initialized Fired only when the device enrolls and the UEM server receives the security status of the device. If the Security status changes, the UEM server processes the change, but does not fire this event. ⓘ For Android devices managed by Ivanti Neurons for MDM, to include the IMEI value in Android devices' device.report_initialized events, go to **Admin** > **Android** → **Registration** on the MobileIron Ivanti Neurons for MDM Admin portal and enable **Require Android device identifiers during registration (Work Profile & Device Admin)**.This is supported for Android 6.0 devices and above. • device.enrolled • device.check_in • device.not_compliant • device.compliant • device.retired • device.app_inventory • device.cert_inventory • device.profile_inventory • device.wiped
devices	A list of devices.

Profile

Field name	Description
encrypted	Indicates whether the profile payload content is encrypted or not
hasRemovalPasscode	Indicates whether the profile has a removal passcode
managed	Indicates whether the profile is managed by Ivanti
removalDisallowed	Supervised only. If present and set to true, the user cannot delete the profile (unless the profile has a removal password and the user provides it)
content	Array of payload dictionaries. Not present if 'encrypted' is true.
description	A description of the profile
displayName	Friendly name that identifies the entity.
identifier	A reverse-DNS style identifier (com.example.myprofile, for example) that identifies the profile
organization	A human-readable string containing the name of the organization that provided the profile
uuid	A globally unique identifier for the profile
version	The version number of the profile format

Rules

These are the rules used to create device groups on Ivanti Neurons for MDM and labels on Ivanti EPMM.

Field name	Description
id	Unique identifier for rule group
name	Rule name
description	Rule group description
definition	Filter criteria for rule

User

Field name	Description
enabled	Indicates whether or not the user is enabled in the system.
createdAt	Epoch time when the user was on-boarded into the system.
accountSource	AccountSource enum.
displayName	Display name of the user.
emailAddress	Registered email address of the user.
firstName	User first name.
lastName	User last name.
userId	User name. This is the field by which you can search or count users.
userUuid	User unique identifier, to be used for the API call, "Update user attributes" on page 97.
customAttributes	All custom user attribute fields that have been set with a value for the user will be listed.
ldapCustomAttributes	All LDAP custom attribute fields that have been set with a value for the LDAP user will be listed.

Sign out

Field name	Description
timestamp	
eventType	device.user_signout
devices	
compliant	
quarantined	
blocked	
compromised	
status	
lastCheckInTime	
registrationTime	
identifier	
imei	
macAddress	
manufacturer	
model os	
osVersion	
serialNumber	
userId	
userUuid	
iosUdid	
deviceGuid	

Handling Paged Responses

When making calls that return a large number of results as a list, it will often be of benefit to page the result set. By requesting smaller subsets of data, you will get a response much faster than when requesting the entire, potentially large, data set.

On calls that support result set paging, pass in the following parameters to control size and start point of the page:

Name	Description
offset	The index of the first item for which you want results.
limit	The maximum number of items you want to be included in the result set. Note that there may be less remaining items than the value you specify here. The default limit is 50.

To page through the results, begin with a start value of 0 and a count value of N. To get the next page, set start value to N, while the count value stays the same. Subsequent pages will start at 2N, 3N, 4N, et cetera.

Inventory APIs

Get application inventory

This API returns the application inventory for the device identifier(s) specified in the request. This API consists of a command and optional payload (URL) that you can use to work with GUIDs instead of mdm-identifiers. This call silently ignores invalid device identifiers in the request.

ⓘ The app inventory response depends on the device mode on Android and data privacy permissionson on iOS.

To enable this API call to retrieve Windows 10 device data on Ivanti EPMM, see "Enabling Windows 10 app inventory reporting on Ivanti EPMM" on page 106.

HTTP Method

POST

Request URI

/msa/v1/cps/device/application

Request parameters

Parameter	Description	Sample Value

identifiers	Required
	Parameter Type: Request body
	Data Type: String
	List of UEM identifiers in this format:
	{
	"identifiers":["mdm-id-1", "mdm-id-2"]
	}
	See "List of identifiers" on page 34.
		"identifiers": ["e428c618933372785b0641d053c562021ef9cd49"

identifierType	**Optional** **Parameter Type:** Request body **Data Type:** Enum **Possible Values:** One of : • MDM_IDENTIFIER • GUID If not specified, the server assumes MDM identifiers.	"identifierType":"MDM_IDENTIFIER"

Response fields

See "Application" on page 36.

Example request

```
curl --location --request POST 'https://[Ivanti Neurons for MDM or Ivanti
EPMM]/msa/v1/cps/device/application' \
--header 'Authorization: Basic <token>' \
--header 'Content-Type: application/json' \
--data-raw '{"identifiers": ["e428c618933372785b0641d053c562021ef9cd49"] }'
```

Example response

```
[
    {
        "deviceMdmId": "e428c618933372785b0641d053c562021ef9cd49",
        "applications": [
            {
                "identifier": "com.apple.Keynote",
                "name": "Keynote",
                "version": "5.2.1",
                "managed": false,
                "source": "UNKNOWN_APP_TYPE",
                "adHocCodeSigned": false,
                "appStoreVendable": true,
                "betaApp": false,
                "deviceBasedVPP": false,
                "externalVersionIdentifier": 832990878,
                "updateAvailable": false,
                "installing": false,
                "validated": true,
                "shortVersion": "5.2.1",
                "bundleVersion": "6733"
            },
            {
                "identifier": "com.apple.Numbers",
                "name": "Numbers",
                "version": "5.2.1",
                "managed": false,
                "source": "UNKNOWN_APP_TYPE",
                "adHocCodeSigned": false,
                "appStoreVendable": true,
                "betaApp": false,
                "deviceBasedVPP": false,
                "externalVersionIdentifier": 832990870,
                "updateAvailable": false,
                "installing": false,
                "validated": true,
                "shortVersion": "5.2.1",
                "bundleVersion": "6733"
            }
        ]
    }
]
```

Get certificate inventory (iOS only)

This call gets the certificate inventory for the iOS device identifiers specified in the request. This call silently ignores any invalid identifiers passed.

HTTP Method

POST

Request URI

/msa/v1/cps/device/certificate

Request parameters

Parameter	Description	Sample Value

identifiers	Required
	Parameter Type: Request body
	Data Type: String
	List of UEM identifiers in this format:
	{
	"identifiers":["mdm-id-1", "mdm-id-2"]
	}
	See "List of identifiers" on page 34.	

Response fields

See "Certificate" on page 37.

Example response

```
[
    {
        "deviceMdmId": "1cdacac686e39dd367a47f2286be436df0a37433",
        "certificates": [
            {
                "identity": true,
                "notAfter": 1812011724000,
                "notBefore": 1496740310000,
                "issuer": "CN=ppp819.auto.acme.com,OU=iOSMDMCA,UID=1496738424913",
                "serialNumber": "1017",
                "subject": "CN=1cdacac686e39dd367a47f2286be436df0a37433",
                "thumbPrint": "BF0670CFC5B548D5837E9D078F4617D296C9577E"
            },
            {
                "identity": false,
                "notAfter": 2035213580000,
                "notBefore": 1088528780000,
                "issuer": "C=US,O=The Go Daddy Group\\, Inc.,OU=Go Daddy Class 2
Certification Authority",
                "serialNumber": "0",
                "subject": "C=US,O=The Go Daddy Group\\, Inc.,OU=Go Daddy Class 2
Certification Authority",
                "thumbPrint": "2796BAE63F1801E277261BA0D77770028F20EEE4"
            },
            {
                "identity": false,
                "notAfter": 2442818116000,
                "notBefore": 1496738116000,
                "issuer": "CN=ppp819.auto.acme.com,OU=SystemRootCA,UID=1496738416965",
                "serialNumber": "750514842",
                "subject": "CN=ppp819.auto.acme.com,OU=SystemRootCA,UID=1496738416965",
                "thumbPrint": "C3EA70EB304FCD0500FF0369EC06103CF1C0260E"
            },
            {
                "identity": true,
                "notAfter": 1812011720000,
                "notBefore": 1496740332000,
                "issuer": "CN=ppp819.auto.acme.com,OU=AppStoreFrontCA,UID=1496738420411",
                "serialNumber": "1018",
                "subject": "CN=1cdacac686e39dd367a47f2286be436df0a37433",
                "thumbPrint": "C45966CC2222CCB5457D2C614B264ACE89EA683A"
            }
        ]
    }
]
```

Get profile inventory (iOS only)

This calls returns the profile inventory details for the device identifiers specified in the request. This API consists of a command and optional payload (URL) that you can use to work with GUIDs instead of mdm-identifiers.

Any invalid device identifier specified in the request is silently ignored.

HTTP Method

POST

Request URI

/msa/v1/cps/device/profile

Request parameters

Parameter	Description	Sample Value

identifiers	Required
	Parameter Type: Request body
	Data Type: String
	List of UEM identifiers in this format:
	{
	"identifiers":["mdm-id-1", "mdm-id-2"]
	}
	See "List of identifiers" on page 34.	

identifierType	**Optional** **Parameter Type:** Request body **Data Type:** Enum **Possible Values:** One of : • MDM_IDENTIFIER • GUID If not specified, the server assumes MDM identifiers.	GUID

Response fields

See "Profile" on page 42.

Example response

```
[
        {
```

```
        "deviceMdmId": "8bfb639c2676ed8c4d55c8962773698fe2e04cd4",
        "profiles": [
            {
                "encrypted": true,
                "hasRemovalPasscode": false,
                "managed": true,
                "removalDisallowed": true,
                "description": "Access the Apple App Catalog via this Web Clip.",
                "displayName": "Apple App Catalog",
                "identifier": "mi.webclip.35016.0",
                "organization": "acme, Inc.",
                "uuid": "e45473ff-bc98-4041-a220-94019fd2b9d9",
                "version": "1"
            },
            {
                "encrypted": true,
                "hasRemovalPasscode": false,
                "managed": true,
                "removalDisallowed": true,
                "description": "The Identity used by this device to access the Apple App
Catalog securely.",
                "displayName": "Identity for the App Catalog",
                "identifier": "mi.credentialidentitydg.35018.0",
                "organization": "acme, Inc.",
                "uuid": "5d299f56-6dbb-4318-9f53-39e8a78950f1",
                "version": "1"
            },
            {
                "encrypted": true,
                "hasRemovalPasscode": false,
                "managed": false,
                "removalDisallowed": false,
                "description": "The top-level MDM payload containing the MDM profile, the
identities and the trust certificates necessary to MDM-manage this device.",
                "displayName": "Root MDM Profile",
                "identifier": "com.acme.polaris.mdm",
                "organization": "acme, Inc.",
                "uuid": "3cdac85c-d56d-4d75-835c-dce062d66c9e",
                "version": "1"
            }
        ]
    }
]
```

Device APIs

Please note that for any of the device GET APIs, the value returned by the API corresponds to the device state at the last device check-in. If you would like to get the most up to date value, then include a "Force checkin" on page 73 call in your work-flow.

Get device attributes metadata

This call returns available device attributes and their types. Some fields such as **devicestatus** and **deviceplatform** will have a value from a finite known set. See "Enums" on page 35 for all such fields.

HTTP Method

GET

Request URI

/msa/v1/cps/device/$metadata

Example response

```
[
    {
        "name": "compliant",
        "dataType": "boolean",
        "readOnly": true
    },
    {
        "name": "status",
        "dataType": "devicestatus",
        "readOnly": true
    },
    {
        "name": "lastCheckInTime",
        "dataType": "long",
        "readOnly": true
    },
    {
        "name": "macAddress",
        "dataType": "string",
        "readOnly": true
    },
    {
        "name": "os",
        "dataType": "deviceplatform",
        "readOnly": true
    },
    {
        "name": "userId",
        "dataType": "string",
        "readOnly": true
    },
    {
        "name": "userUuid",
        "dataType": "string",
        "readOnly": true
    },
    {
        "name": "custom_device_attribute_1",
        "dataType": "string",
        "readOnly": false
    },
    {
        "name": "custom_device_attribute_2",
        "dataType": "string",
        "readOnly": false
    }
]
```

Get devices by MDM or deviceGuid identifiers

This API returns the details of devices with the MDM or deviceGuid identifiers specified in the request body. It silently ignores invalid MDM and deviceGuid identifiers. In the following device details resturn, the MDM identifier is in the field "identifier," and the deviceGuid is in the field, "deviceGuid." Those are the values, for example, "d0838fa7ba1ee023c9ece9192589088325d7aa08" and "dbccbe70-7499-4676-88b0-a6d5d298752e" that you could use in the body of this call.

```
{
    "compliant": true,
    "quarantined": false,
    "blocked": false,
    "compromised": false,
    "status": "ACTIVE",
    "lastCheckInTime": 1581229664468,
    "registrationTime": 1581229603492,
    "identifier": "d0838fa7ba1ee023c9ece9192589088325d7aa08",
    "macAddress": "8c:fe:57:b3:53:96",
    "manufacturer": "Apple Inc.",
    "model": "iPad7,5",
    "os": "IOS",
    "osVersion": "13.3",
    "serialNumber": "GG7XP285JMVT",
    "userId": "b2@m.com",
    "userUuid": "b8e7fea0-4730-4bc5-a089-cfd631078b9e",
    "iosUdid": "d0838fa7ba1ee023c9ece9192589088325d7aa08",
    "ownership": "UNKNOWN",
    "deviceGuid": "dbccbe70-7499-4676-88b0-a6d5d298752e"
}
```

HTTP Method

POST

Request URI

/msa/v1/cps/device/uuid

Request parameters

Parameter	Description	Sample Value

identifierType	Required
	Parameter Type: Request body
	Data Type: String
	Type of identifier.
	Valid values:
	• "identifierType":"MDM_IDENTIFIER"
	• "identifierType":"GUID"
		{
		"identifierType":"GUID",
		"identifiers":["deviceGuid-1", "deviceGuid-2"]
		}

identifiers	Required
	Parameter Type: Request body
	Data Type: String
	List of UEM identifiers in this format:
	{
	"identifierType":"MDM_IDENTIFIER",
	"identifiers":["mdm-id-1", "mdm-id-2"]
	}
	or
	{
	"identifierType":"GUID",
	"identifiers":["deviceGuid-1", "deviceGuid-2"]
	}
	See "List of identifiers" on page 34.
		{
		"identifierType":"GUID", "identifiers":["dbccbe70-7499-4676-88b0-a6d5d298752e"]
		}

Response fields

A list of devices.

See "List of identifiers" on page 34 and "User" on page 43.

Example request

```
curl --location --request POST 'https://<Ivanti Neurons for MDM>/msa/v1/cps/device/uuid'
\
--header 'Authorization: Basic <token>' \
--header 'Content-Type: application/json' \
--data-raw '{
"identifiers":["dbccbe70-7499-4676-88b0-a6d5d298752e"]
}'
```

Example response

```
[
    {
        "compliant": true,
        "quarantined": false,
        "blocked": false,
        "compromised": false,
        "status": "ACTIVE",
        "lastCheckInTime": 1581229664468,
        "registrationTime": 1581229603492,
        "identifier": "d0838fa7ba1ee023c9ece9192589088325d7aa08",
        "macAddress": "8c:fe:57:b3:53:96",
        "manufacturer": "Apple Inc.",
        "model": "iPad7,5",
        "os": "IOS",
        "osVersion": "13.3",
        "serialNumber": "GG7XP285JMVT",
        "userId": "b2@m.com",
        "userUuid": "b8e7fea0-4730-4bc5-a089-cfd631078b9e",
        "iosUdid": "d0838fa7ba1ee023c9ece9192589088325d7aa08",
        "ownership": "UNKNOWN",
        "deviceGuid": "dbccbe70-7499-4676-88b0-a6d5d298752e"
    }
]
```

Get devices by mac addresses

This API returns the details of devices with MAC addresses specified in the request. It silently ignores invalid MAC addresses.

HTTP Method

POST

Request URI

/msa/v1/cps/device/mac

Request parameters

Parameter	Description	Sample Value

identifiers	Required
	Parameter Type: Request body
	Data Type: String
	List of wifi MAC identifiers in this format:
	{
	"identifiers":["mac-id-1", "macm-id-2"]
	}
	See "List of identifiers" on page 34.	

Response fields

A list of devices.

See "List of identifiers" on page 34 and "User" on page 43.

Search devices by status

This API returns the device information for devices with status specified in the request. If no status is specified in the request, details of all the devices is returned.

HTTP Method

GET

Request URI

/msa/v1/cps/device?status=<DeviceStatus>&offset=<Integer>&limit=<Integer>

Request parameters

Parameter	Description	Sample Value
status	**Parameter Type:** Query **Data Type:** String	See "status" on page 39.
offset	**Parameter Type:** Query **Data Type:** Number **Default:** 0	0
limit	**Parameter Type:** Query **Data Type:** String **Default:** 50	50

Response fields

A list of devices.

See "List of identifiers" on page 34 and "User" on page 43.

Example response

```
{
   "searchResults": #List of devices,
   "results":2,
   "offset":0,
   "limit":50
}
```

Search devices by device group identifier

This API returns device membership details for device group identifier specified in the search request.

HTTP Method

GET

Request URI

/msa/v1/cps/device?ruleId=<id>&offset=<Offset>&limit=<Limit>

Request parameters

Parameter	Description	Sample Value
id	**Required** **Parameter Type:** Query **Data Type:** String ID of the device group whose device members to find.	38089
offset	**Parameter Type:** Query **Data Type:** Number **Default:** 0	5
limit	**Parameter Type:** Query **Data Type:** String **Default:** 50	50

Response fields

A list of devices.

See "List of identifiers" on page 34 and "User" on page 43.

Example response

```
{
    "searchResults": [
        {
            "compliant": true,
            "quarantined": false,
            "blocked": false,
            "compromised": false,
            "status": "ACTIVE",
            "lastCheckInTime": 1516619820303,
            "registrationTime": 1516619792690,
            "identifier": "f4aadce56d25edffggh4958484brr1c35f6b6d5",
            "imei": "6733284512443420",
            "macAddress": "fa:c6:7b:0f:eb:ff",
            "manufacturer": "Apple Inc.",
            "model": "iPhone6,1",
            "os": "IOS",
            "osVersion": "10.1",
            "phoneNumber": "7171046951",
            "serialNumber": "d028d4db2162",
            "userId": "testuser@auto.automation.com",
            "userUuid": "047rdee6-31c1-4e55a-898d-97de620d5f22",
            "iosUdid": "f4aadce56d25edffggh4958484brr1c35f6b6d5",
            "ownership": "UNKNOWN",
            "customAttributes": {
                "nacCompliant": "false"
            }
        }
    ],
    "results": 1,
    "offset": 0,
    "limit": 50
}
```

Get device location details

This API returns device location details of the target device.

HTTP Method

POST

Request URI

/msa/v1/cps/device/location

Request parameters

Parameter	Description	Sample Value

identifiers	Required
	Parameter Type: Request body
	Data Type: String
	List of UEM identifiers in this format:
	{
	"identifiers":["mdm-id-1", "mdm-id-2"]
	}
	See "List of identifiers" on page 34.	

Example response

```
[
    {
        "deviceMdmId": "id-1",
        "lastCaptured": 1525260287000,
        "location": {
            "latitude": "-6.446318",
            "longitude": "8.305312"
        }
    },
    {
        "deviceMdmId": "id-2",
        "lastCaptured": 1525260287000,
        "location": {
            "latitude": "5.178482",
            "longitude": "240.943423"
        }
    }
]
```

Count devices by status

This API returns the count of devices with status specified in the request. If no status is specified in the request, count of all the devices is returned.

HTTP Method

GET

Request URI

/msa/v1/cps/device/count?status=<DeviceStatus>

Request parameters

Parameter	Description	Sample Value
status	**Parameter Type:** Query **Data Type:** String	See "status" on page 39.

Response fields

A number result.

See "Number result" on page 35.

Count devices by device group identifier

This API returns device membership count for device group identifier specified in the search request.

HTTP Method

GET

Request URI

/msa/v1/cps/device/count?ruleId=<id>

Request parameters

Parameter	Description	Sample Value
id	**Required** **Parameter Type:** Query **Data Type:** String ID of the device group whose member devices to count.	38089

Response fields

A number result.

See "Number result" on page 35.

Example response

```
{
    "result": 1
}
```

Update device attributes by device UEM identifiers

This API is used to update attributes for the device identifiers specified in the request. It returns the count of devices for which the attributes were successfully updated.

HTTP Method

POST

Request URI

/msa/v1/cps/device

Request parameters

List of update attributes.

See "Update attributes" on page 35.

Response fields

API returns the count of devices for which attributes were successfully updated along with HTTP status 200.

See "Number result" on page 35.

Manage device settings - iOS only

This API updates specific iOS settings based on the device identifiers in the request. The supported settings are:

- Personal hotspot
- Data roaming

Mobile data needs to be switched on in the device to allow enabling of the hotspot. WiFi and Bluetooth are not required for enablement, but for other devices to connect to the hotspot.

The API returns the count of devices for which the settings was successfully updated. The call silently ignores any invalid device identifiers specified in the request.

HTTP Method

POST

Request URI

/msa/v1/cps/device/settings

Request parameters

Parameter	Description	Sample Value

identifiers	Required
	Parameter Type: Request body
	Data Type: String
	List of UEM identifiers in this format:
	{
	"identifiers":["mdm-id-1", "mdm-id-2"]
	}
	See "List of identifiers" on page 34.
		{
		"identifiers": [
		"b9a0d7d05fb4204f52264f805932526b848ca61d"
		],
		"settings": {
		"type": "PERSONAL_HOTSPOT",
		"value": {
		"enabled": true
		}
		}
		}

settings	Required
	Parameter Type: Request body
	Data Type: String
	JSON container for the type and value parameters.
		{
		"identifiers": [
		"b9a0d7d05fb4204f52264f805932526b848ca61d"
		],
		"settings":
		{
		"type": "PERSONAL_HOTSPOT",
		"value": {
		"enabled": true
		}
		}
		}

Parameter	Description	Sample Value

type	Required
	Parameter Type: Request body
	Data Type: String
	The settings to update. Can be:
	• PERSONAL_HOTSPOT
	• DATA_ROAMING
		{
		    "identifiers": [
		        "b9a0d7d05fb4204f52264f805932526b848ca61d"
		    ],
		    "settings": **{**
		        "type": "PERSONAL_HOTSPOT",
		        "value": {
		            "enabled": true
		        }
		    }
		}

value	Required
	Parameter Type: Request body
	Data Type: String
	The value to set for the settings. Can be:
	• true
	• false
		{
		    "identifiers": [
		        "b9a0d7d05fb4204f52264f805932526b848ca61d"
		    ],
		    "settings": {
		        "type": "PERSONAL_HOTSPOT",
		        "value": {
		            "enabled": true
		        }
		    }
		}

Response fields

API returns the count of devices for which attributes were successfully updated along with HTTP status 200.

See "Number result" on page 35.

Example requests

Turn on personal hotspot:

```
POST /msa/v1/cps/device/settings
{
    "identifiers": [
        "b9a0d7d05fb4204f52264f805932526b848ca61d"
    ],
    "settings": {
        "type": "PERSONAL_HOTSPOT",
        "value": {
            "enabled": true
        }
    }
}
```

Turn off data roaming:

```
POST /msa/v1/cps/device/settings
{
    "identifiers": [
        "b9a0d7d05fb4204f52264f805932526b848ca61d"
    ],
    "settings": {
        "type": "DATA_ROAMING",
        "value": {
            "enabled": false
        }
    }
}
```

Example response

```
{
    "result": 1
}
```

In case all the identifiers are invalid, then response code will still be 200, with result as 0.

Get device settings - iOS only

This API returns values for data roaming and personal hotspot received on the last device check in. If you would like the most up to date value, then include a "Force checkin" on page 73 call in your work-flow. The supported settings are:

- Personal hotspot
- Data roaming

HTTP Method

POST

Request URI

/msa/v1/cps/device/getDeviceSettings

Request parameters

Parameter	Description	Sample Value

identifiers	Required
	Parameter Type: Request body
	Data Type: String
	List of UEM identifiers in this format:
	{
	"identifiers":["mdm-id-1", "mdm-id-2"]
	}
	See "List of identifiers" on page 34.
		{
		"settingsType": "DATA_ROAMING",
		"identifiers": [
		"b9a0d7d05fb4204f52264f805932526b848ca61d",
		"a9a0d7d05fb4204f52264f805932526b848ca61d"
		]
		}

settingsType	Required
	Parameter Type: Request body
	Data Type: String
	The settings to retieve. Can be:
	• PERSONAL_HOTSPOT
	• DATA_ROAMING
		{
		"settingsType": "DATA_ROAMING",
		"identifiers": [
		"b9a0d7d05fb4204f52264f805932526b848ca61d",
		"a9a0d7d05fb4204f52264f805932526b848ca61d"
		]
		}

Response fields

API returns the count of devices for which attributes were successfully updated along with HTTP status 200.

See "Number result" on page 35.

Example request

```
POST /msa/v1/cps/device/settings
{
    "settingsType": "DATA_ROAMING",
    "identifiers": [
        "b9a0d7d05fb4204f52264f805932526b848ca61d",
        "a9a0d7d05fb4204f52264f805932526b848ca61d"
    ]
}
```

Example response

```
{
    "settings": [
        {
            "identifier": "b9a0d7d05fb4204f52264f805932526b848ca61d",
            "settings": {
                "type": "DATA_ROAMING",
                "value": {
                    "enabled": false
                }
            }
        },
        {
            "identifier": "a9a0d7d05fb4204f52264f805932526b848ca61d",
            "settings": {
                "type": "DATA_ROAMING",
                "value": {
                    "enabled": true
                }
            }
        }
    ]
}
```

In case all the identifiers are invalid, then response code will still be 200, with an empty response body.

Force checkin

This API is used to initiate force check-in from the server for the device identifiers specified in the request. Invalid device identifiers are silently ignored. The API returns the count of devices for which force check-in action was successfully initiated.

HTTP Method

POST

Request URI

/msa/v1/cps/device/forceCheckin

Request parameters

List of UEM identifiers.

See "List of identifiers" on page 34.

Response fields

Success: HTTP Status 200, and a number result with the count of successfully checked in devices.

See "Number result" on page 35

Check-in for Threat Response Matrix update

Mobile Threat Detection vendors use this API to notify their agent in our integrated client to immediately check-in with their management service to receive the latest Threat Response Matrix (TRM). This ensures that their agent is working with the most recent application, network, and device threat information, rather than waiting for the next regularly scheduled check-in event. This API consists of a command and optional payload (URL) that vendors can use to work with GUIDs instead of mdm-identifiers.The API returns the count of devices for which the push notification action successfully initiated. The API silently ignores invalid device identifiers specified in the request.

HTTP Method

POST

Request URI

/msa/v1/cps/device/mtd/notification

Request parameters

Parameter	Description
command	**Required** **Parameter Type:** Request body **Data Type:** String **Possible value:** CHECKIN Command to be forwarded to Mobile Treat Defense vendor.
identifierType	**Optional** **Parameter Type:** Request body **Data Type:** Enum **Possible Values:** One of : • MDM_IDENTIFIER • GUID If not specified, the server assumes MDM identifiers.
identifiers	**Required** **Parameter Type:** Request body **Data Type:** String List of mdm identifiers in this format: { "identifiers":["mdm-id-1", "mdm-id-2"] } See "List of identifiers" on page 34.
url	**Optional** **Parameter Type:** Request body **Data Type:** String URL to be forwarded to Mobile Treat Defense vendor.

Response fields

Success: HTTP Status 200, and a number result with the count of successfully checked in devices.

See "Number result" on page 35

Example request payload

```
{
  "command": "CHECKIN",
  "URL": "http://example.com/example",
  "identifiers": [
    "mdm-id-1",
    "mdm-id-2"
  ]
}
```

Example response

```
{
    "result": 2
}
```

In case all the identifiers are invalid, then response code will still be 200, with result as 0.

Device Group APIs

Device groups are containers for logically grouping devices for the ease of device management. Device Groups are “device groups” on Ivanti Neurons for MDM and “labels” on Ivanti EPMM. See "Example device groups API scenario" on page 83 for an example of how you can use these calls to calculate specific group device characteristics.

Get all Device Groups

This call returns device groups information for device groups existing in the system and includes system-defined, statically or manually managed, and dynamically managed device group information. Device groups correspond to rule IDs.

HTTP Method

GET

Request URI

/msa/v1/cps/rule?offset=<Offset>&limit=<Limit>

Request parameters

Parameter	Description	Sample Value
offset	**Parameter Type:** Query **Data Type:** String **Default:** 0 Breaks results into chunks. The syntax is offset=value.	5
limit	**Parameter Type:** Query **Data Type:** String **Default:** 50 Limits the return to the specified number of rows.	25

Response fields

A list of device groups.

See "Rules" on page 42.

Example response

```
{
  "searchResults": [
    {
      "id": 38089,
      "name": "rule1",
      "description": "Rule 1 Description",
      "definition": "PLATFORMTYPE EQ 'IOS'"
    },
    {
      "id": 38090,
      "name": "rule2",
      "description": "Rule 2 Description",
      "definition": "PLATFORMTYPE EQ 'IOS'"
    }
  ],
  "results": 2,
  "offset": 0,
  "limit": 10
}
```

Get device groups associated with device list

This call returns a list of device groups and related information associated with a list of devices. Device groups correspond to rule IDs.

HTTP Method

POST

Request URI

/msa/v1/cps/rule/deviceGuid

Request parameters

Parameter	Description	Sample Value

identifiers	**Required**
	Parameter Type: Request body
	Data Type: String
	List of device GUIDs identifiers in this format:
	`{`
	`    "identifiers": [`
	`        "deviceGuid"`
	`    ]`
	`}`
	See "List of identifiers" on page 34.
		`{`
		`"identifiers": [`
		`"6b8df48e-7778-4a33-97e4-1cc6b39de61c"`
		`]`
		`}`

Response fields

A list of device groups.

See "Rules" on page 42.

Example request

```
curl --location --request POST 'https://[mobileiron-Ivanti Neurons for MDM-or-Ivanti
EPMM]/msa/v1/cps/rule/deviceGuid' \

--header 'Accept: application/json, text/plain, */*' \

--header 'Content-Type: application/json' \

--header 'Authorization: Basic <token>' \
--data-raw {
    "identifiers": [
        "6b8df48e-7778-4a33-97e4-1cc6b39de61c"
    ]
}
```

Example response

```
{
    "errors": null,
    "result": [
        {
            "deviceGuid": "6b8df48e-7778-4a33-97e4-1cc6b39de61c",
            "rules": [
                {
                    "id": 51173,
                    "name": "jk Sandbox User Group",
                    "definition": "DISPLAYNAME EQ 'jk Sandbox User'"
                },
                {
                    "id": 390113,
                    "name": "test#2",
                    "definition": "CURRENTMCC NEQ 'test#'"
                },
                {
                    "id": 361609,
                    "name": "Bozo",
                    "definition": "UNLOCK_TOKEN EQ 'TRUE'"
                },
                {
                    "id": 40165,
                    "name": "All Devices",
                    "description": "Targets any type of device."
                },
                {
                    "id": 118493,
                    "name": "current country",
                    "definition": "CURRENTCOUNTRYNAME EQ 'United States'"
                },
                {
                    "id": 40167,
                    "name": "iOS Devices",
                    "description": "Targets all iOS devices",
                    "definition": "PLATFORMTYPE EQ 'IOS'"
                }
            ]
        }
    ]
}
```

Search device groups by device group name

This call returns device group information for device groups whose name starts with the string specified in the search request.

HTTP Method

GET

Request URI

/msa/v1/cps/rule?name=<name>&offset=<Offset>&limit=<Limit>

Request parameters

Parameter	Description	Sample Value
name	**Required** **Parameter Type:** Query **Data Type:** String The prefix string of the device group name on which to perform the search. Not context sensitive. No records are returned if this parameter is blank.	iOS

Response fields

A list of device groups.

See "Rules" on page 42.

Example response

```
{
    "searchResults": [
        {
            "id": 38089,
            "name": "rule1",
            "description": "Rule 1 Description",
            "definition": "PLATFORMTYPE EQ 'IOS'"
        },
        {
            "id": 38090,
            "name": "rule2",
            "description": "Rule 2 Description",
            "definition": "CFDEVICE[nacCompliant] EQ 'false'"
        }
    ],
    "results": 2,
    "offset": 0,
    "limit": 10
}
```

Get device group by device group identifier

This call returns device group details for the device group identifier specified in the request.

HTTP Method

GET

Request URI

/msa/v1/cps/rule/<id>

Request parameters

Parameter	Description	Sample Value
id	**Required** **Parameter Type:** Query **Data Type:** String The device group ID on which to search. You can use the call, "Get all Device Groups" on page 77, or the call, "Search device groups by device group name" on the previous page, to find the device group id.	38089

Response fields

Rule information for the device group identifier specified.

See "Rules" on page 42.

Example response

```
{
    "id": 38089,
    "name": "rule1",
    "description": "Rule 1 Description",
    "definition": "PLATFORMTYPE EQ 'IOS'"
}
```

Example device groups API scenario

You can use the device group APIs to calculate device group characteristics, such as the percentage of Android Enterprise-enrolled devices. Remember that device groups are “device groups” on Ivanti Neurons for MDM, and “labels” on Ivanti EPMM.

Example

1. First, use the "Get all Device Groups" on page 77 call to get a list of all the device groups, so you know which device groups you can use for your calculations.

Call:

GET msa/v1/cps/rule

Response:

```
{
    "searchResults": [
        {
            "id": 38000,
            "name": "All Devices",
            "description": "Targets any type of device."
        },
        {
            "id": 38001,
            "name": "Android Devices",
            "description": "Targets all Android devices",
            "definition": "PLATFORMTYPE EQ 'ANDROID'"
        },
        {
            "id": 38002,
            "name": "Android Enterprise Devices",
            "description": "Targets all Android Enterprise devices",
            "definition": "PLATFORMTYPE EQ 'ANDROID' AND ANDROIDWORKENABLED EQ 'true'"
        },
        {
            "id": 38003,
            "name": "iOS Devices",
            "description": "Targets all iOS devices",
            "definition": "PLATFORMTYPE EQ 'IOS'"
        },
        {
            "id": 38004,
            "name": "Windows Devices",
            "description": "Targets all Windows devices",
            "definition": "PLATFORMTYPE EQ 'WINDOWS_PHONE'"
        },
        {
            "id": 38005,
            "name": "macOS Devices",
            "description": "Targets all OSX devices",
            "definition": "PLATFORMTYPE EQ 'OSX'"
        },
        {
            "id": 38006,
            "name": "tvOS Devices",
            "description": "Targets all tvOS devices.",
            "definition": "PLATFORMTYPE EQ 'IOS' AND MODEL STARTS_WITH 'AppleTV')"
        }
    ],
    "results": 7,
    "offset": 0,
    "limit": 50
}
```

2. Use the "Count devices by device group identifier" on page 63 call to find the total android device count.

Call:

GET : /msa/v1/cps/device?ruleId=38001

Note that ruleId 38001 targets all Android devices.

3. Use the "Count devices by device group identifier" on page 63 and then the number of Android Enterprise-enrolled devices.

Call:

GET : /msa/v1/cps/device?ruleId=38002

Note that ruleId 38002 targets all Android Enterprise-enrolled devices.

4. Using the two values returned by the preceding two calls, you can calculate the percentage of Android devices that are also Android Enterprise-enrolled devices.

Message APIs

Send email to device owners

This API is used to send email notification to device owners by specifying device identifiers in the request. The API returns the count of devices for which the send email action was successfully initiated. Any invalid device identifier specified in the request is silently ignored.

HTTP Method

POST

Request URI

/msa/v1/cps/device/email

Request parameters

Parameter	Description	Sample Value
subject	**Parameter Type:** Request body **Data Type:** String Subject of the email. Character limit is 100.	Please backup your device.
body	**Parameter Type:** Request body **Data Type:** String Body of the email. Character limit is 3000.	Please backup your device to ensure against data loss.
html	**Parameter Type:** Request body **Data Type:** Boolean Specifies whether the email is in HTML format.	true

identifiers	Required
	Parameter Type: Request body
	Data Type: String
	List of UEM identifiers in this format:
	{
	"identifiers":["mdm-id-1", "mdm-id-2"]
	}
	See "List of identifiers" on page 34.	

See "List of identifiers" on page 34.

Response fields

Success: HTTP Status 200

See "Number result" on page 35.

Send push notification to devices

This API is used to send notifications by specifying device identifiers in the request. The API returns the count of devices for which the push notification action was successfully initiated. Any invalid device identifier specified in the request is silently ignored.

HTTP Method

POST

Request URI

/msa/v1/cps/device/notification

Request parameters

Parameter	Description	Sample Value
message	**Parameter Type:** Request body **Data Type:** String Content of push message. Character limit is 140.	Please backup your device.

identifiers	Required
	Parameter Type: Request body
	Data Type: String
	List of UEM identifiers in this format:
	{
	"identifiers":["mdm-id-1", "mdm-id-2"]
	}
	See "List of identifiers" on page 34.	

Response fields

Success: HTTP Status 200

See "Number result" on page 35.

User APIs

This chapter describes the Common Platform Services API calls you use to query and manipulate Ivanti Neurons for MDM and Ivanti EPMM users.

Get user attributes metadata

Get user attributes metadata

This call returns available user attributes and their types.

HTTP Method

GET

Request URI

/msa/v1/cps/user/$metadata

Example response

```
[
    {
        "name": "enabled",
        "dataType": "boolean",
        "readOnly": true
    },
    {
        "name": "createdAt",
        "dataType": "long",
        "readOnly": true
    },
    {
        "name": "accountSource",
        "dataType": "string",
        "readOnly": true
    },
    {
        "name": "displayName",
            "dataType": "string",
            "readOnly": true
        },
        {
            "name": "emailAddress",
            "dataType": "string",
            "readOnly": true
        },
        {
            "name": "firstName",
            "dataType": "string",
            "readOnly": true
        },
        {
            "name": "lastName",
            "dataType": "string",
            "readOnly": true
        },
        {
            "name": "userId",
            "dataType": "string",
            "readOnly": true
        },
        {
            "name": "userUuid",
            "dataType": "string",
            "readOnly": true
        },
        {
            "name": "customAttributes",
            "dataType": "map",
            "readOnly": true
        },
        {
            "name": "ldapCustomAttributes",
            "dataType": "map",
            "readOnly": true
        },
        {
            "dataType": "map",
            "name": "ldapStandardAttributes",
            "readOnly": true
        },

        {
            "name": "ios",
            "dataType": "string",
            "readOnly": false
        },
        {
            "name": "site",
            "dataType": "string",
            "readOnly": false
        }
]
```

Search users by GUID

This call searches for users by GUID. See "User" on page 43. This call returns custom LDAP attributes along with other user information, unless issued with the **fields="ldapStandardAttributes"** parameter, which adds LDAP standard attributes to the response.

HTTP Method

GET

Request URI

/msa/v1/cps/user?guid=<string>&fields="ldapStandardAttributes"

Request parameters

Parameter	Description	Sample Value
guid	**Parameter Type:** Query **Data Type:** String	dd8b7006-2780-49c1-9b03-f85aa57c9da4
fields="ldapStandardAttributes"	**Optional** **Parameter Type:** Query **Data Type:** String If this parameter is included, the return includes the following standard LDAP attributes: • distinguishedName • samAccountName • userPrincipalName	fields="ldapStandardAttributes"

Response fields

A list of users.

See "List of identifiers" on page 34 and "User" on page 43.

Example responses

Local accountSource

```
{
  "searchResults": [
    {
      "enabled": true,
      "createdAt": 1580812767902,
      "accountSource": "LOCAL",
      "displayName": "John Doe",
      "emailAddress": "admin@acme.com",
      "firstName": "John",
      "lastName": "Doe",
      "userId": "admin@acme.com",
      "userUuid": "dd8b7006-2780-49c1-9b03-f85aa57c9da4",
      "guid": "dd8b7006-2780-49c1-9b03-f85aa57c9da4"
    }
  ],
  "results": 1,
  "offset": 0,
  "limit": 50
}
```

LDAP accountSource with fields="ldapStandardAttributes" parameter included in the call

```
{
  "limit": 50,
  "offset": 0,
  "results": 1,
  "searchResults": [
    {
      "accountSource": "LDAP",
      "createdAt": 1500444347689,
      "displayName": "Fred A",
      "emailAddress": "Freda@auto8.mobileiron.com",
      "enabled": true,
      "firstName": "Fred",
      "ldapCustomAttributes": {
        "lastlogontimestamp": "131516814255658414",
        "mailnickname": "Fred",
        "mdbusedefaults": "TRUE",
        "msexchmailboxguid": "k\ufffd\ufffdVt\ufffdN\ufffd\ufffdS)\u289e=",
        "msexchrecipientdisplaytype": "1073741824",
        "msexchwhenmailboxcreated": "20140407185418.0Z"
      },
      "ldapStandardAttributes": {
        "distinguishedName": "foo",
        "samAccountName": "bar",
        "userPrincipalName": "baz"
      }
      "userId": "Freda@auto8.mobileiron.com",
      "userUuid": "a3343fd0-c1f2-424f-a993-83b951c94246",
      "guid": "dd8b7006-2780-49c1-9b03-f85aa57c9da4"
    }
  ]
}
```

Search users by user id

This call searches for users by user IDs by searching for users whose user ID starts with the given string. See "User" on page 43. This call returns custom LDAP attributes along with other user information, unless issued with the **fields="ldapStandardAttributes"** parameter, which adds LDAP standard attributes to the response. Note that the call searches only on the email address as the user ID.

HTTP Method

GET

Request URI

/msa/v1/cps/user?uid=<String>&fields="ldapStandardAttributes"&offset=<Offset>&limit=<Limit>

Request parameters

Parameter	Description	Sample Value
uid	**Parameter Type:** Query **Data Type:** String	The email prefix, for example, the “btarris” in “:btarris@mobileiron.com.”
fields="ldapStandardAttributes"	**Optional** **Parameter Type:** Query **Data Type:** String If this parameter is included, the return includes the following standard LDAP attributes: • distinguishedName • samAccountName • userPrincipalName	fields="ldapStandardAttributes"
offset	**Parameter Type:** Query **Data Type:** Number **Default:** 0	0
limit	**Parameter Type:** Query **Data Type:** String **Default:** 50	50

Response fields

A list of users.

See "List of identifiers" on page 34 and "User" on page 43.

Example responses

Local accountSource

```
{
    "searchResults": [
        {
            "enabled": true,
            "createdAt": 1466202626423,
            "accountSource": "LOCAL",
            "displayName": "JGB",
            "emailAddress": "abc@mobileiron.com",
            "firstName": "Josh",
            "lastName": "Brit",
            "userId": "abc@mobileiron.com",
            "userUuid": "0aed8d85-6dfd-40ac-b3b5-82f629e3c168"
        },
        {
            "enabled": true,
            "createdAt": 1466525975950,
            "accountSource": "LOCAL",
            "displayName": "Bob Tarris",
            "emailAddress": "btarris@mobileiron.com",
            "firstName": "Bob",
            "lastName": "Tarris",
            "userId": "btarris@mobileiron.com",
            "userUuid": "b7149c70-30ef-4b88-b8f3-18ec273e8c80"
        },
        {
            "enabled": true,
            "createdAt": 1467161380058,
            "accountSource": "LOCAL",
            "displayName": "test user",
            "emailAddress": "test@mob.dev",
            "firstName": "test",
            "lastName": "user",
            "userId": "test@mob.dev",
            "userUuid": "e0ad3de5-2390-4bb3-afdf-acfa9838503b",
            "custom_user": "from rest 2"
        }
    ],
    "results": 3,
    "offset": 0,
    "limit": 10
}
```

LDAP accountSource with fields="ldapStandardAttributes" parameter included in the call

```
{
  "limit": 50,
  "offset": 0,
  "results": 1,
  "searchResults": [
    {
      "accountSource": "LDAP",
      "createdAt": 1500444347689,
      "displayName": "Fred A",
      "emailAddress": "Freda@auto8.mobileiron.com",
      "enabled": true,
      "firstName": "Fred",
      "ldapCustomAttributes": {
        "lastlogontimestamp": "131516814255658414",
        "mailnickname": "Fred",
        "mdbusedefaults": "TRUE",
        "msexchmailboxguid": "k\ufffd\ufffdVt\ufffdN\ufffd\ufffdS)\u289e=",
        "msexchrecipientdisplaytype": "1073741824",
        "msexchwhenmailboxcreated": "20140407185418.0Z"
      },
      "ldapStandardAttributes": {
        "distinguishedName": "foo",
        "samAccountName": "bar",
        "userPrincipalName": "baz"
      }
      "userId": "Freda@auto8.mobileiron.com",
      "userUuid": "a3343fd0-c1f2-424f-a993-83b951c94246"
    }
  ]
}
```

Count users by user id

This call counts users by user IDs by searching for users whose user ID starts with the given string. See "User" on page 43. Note that the call searches only on the email address as the user ID.

HTTP Method

GET

Request URI

/msa/v1/cps/user/count?uid=<String>

Request parameters

Parameter	Description	Sample Value
uid	**Parameter Type:** Query **Data Type:** String	jilldoe@acme.com

Response fields

A number result of users whose user id starts with the given string.

See "Number result" on page 35.

Update user attributes

This API is used to update attributes for the user identifiers specified in the request. It returns the count of users for which the attributes were successfully updated. The API silently ignores any invalid user identifiers specified in the request body.

HTTP Method

POST

Request URI

/msa/v1/cps/user

Request parameters

List of update attributes.

See "Update attributes" on page 35.

Response fields

Count of users for which attributes were successfully updated.

See "Number result" on page 35.

App Catalog APIs

Get application inventory

This API returns information about the apps in the app catalog.

HTTP Method

GET

Request URI

/msa/v1/cps/appcatalog/apps

Request parameters

Parameter	Description	Sample Value
identifier	**Parameter Type:** Query **Data Type:** String Bundle id of the application (exact match).	com.facebook.Facebook
name	**Parameter Type:** Query **Data Type:** String Name of the app (matches starts with) Note: This parameter is case sensitive. See this known issue.	Facebook
platformType	**Parameter Type:** Query **Data Type:** String Type of the os platform (exact match). Case sensitive. Must be one of these values: • IOS • OSX • ANDROID • WINDOWS • WEBAPPS WEBAPPS is only supported for Ivanti EPMM. WINDOWS is not working in this release. See this known issue.	IOS

Response fields

See "App Catalog" on page 36.

Example request

/msa/v1/cps/appstore/apps

Example responses

```
{
    "searchResults": [
        {
            "name": "Google Photos",
            "platformType": "Android",
            "version": "2019.02.19",
            "identifier": "com.google.android.apps.photos",
            "createdAt": 1550659389308
        },
        {
            "name": "Facebook",
            "platformType": "iOS",
            "version": "208.0",
            "identifier": "com.facebook.Facebook",
            "createdAt": 1550659372336
        },
        {
            "name": "Windows Scan",
            "platformType": "Windows",
            "identifier": "Microsoft.WindowsScan_8wekyb3d8bbwe",
            "createdAt": 1550659437232
        },
        {
            "name": "MAC Address Scan",
            "platformType": "Mac OS X",
            "version": "2.6.9",
            "identifier": "com.Dandelion.MACAddressScan",
            "createdAt": 1550659454097
        }
    ],
    "results": 4,
    "offset": 0,
    "limit": 50
}
```

In-house app response

Note that this response contains the buildNumber field, available only for in-house apps.

```
{
    "searchResults": [
        {
            "name": "HAC-Extension",
            "platformType": "iOS",
            "version": "1.4.2.0.0.17",
            "identifier": "com.mobileiron.enterprise.HAC-Extension",
            "buildNumber": "1.4.2.0.0.17",
            "createdAt": 1550660326185
        }
    ],
    "results": 11,
    "offset": 0,
    "limit": 50
}
```

Metadata APIs

Get current minor version of the API

This call gets the current minor version of the API.

HTTP Method

GET

Request URI

/msa/v1/cps/version

Response fields

Field	Description
minorVersion	A string representing the minor version of the target API.

.

Example response

{

"minorVersion": "1.0.0"

}

Get device registration URI

This API returns the URL that users visit to register devices. This is the URL to which you would direct users to register their devices.

HTTP Method

GET

Request URI

/msa/v1/cps/device/registration

Example request

```
curl --location 'https://<EPMM or MDM host>/msa/v1/cps/device/registration' \
--header 'Accept: application/json, text/plain, */*' \
--header 'Content-Type: application/json' \
--header 'Authorization: Basic <token>'
```

Example responses

Ivanti Neurons for MDM

```
{
    "registrationUrl": "https:<Ivanti Neurons for MDM host>/ireg/index.html"
}
```

Ivanti EPMM

```
{
    "registrationUrl": "https:<Ivanti EPMM host>/go"
}
```

Get device app metadata

This API returns the device app metadata.

HTTP method

GET

Request URI

/msa/v1/cps/device/application/$metadata

Example response

```
[
    {
        "name": "identifier",
        "dataType": "string",
        "readOnly": true
    },
    {
        "name": "name",
        "dataType": "string",
        "readOnly": true
    },
    {
        "name": "version",
        "dataType": "string",
        "readOnly": true
    }
]
```

Get device certificate metadata

This API returns the device certificate metadata.

HTTP method

GET

Request URI

/msa/v1/cps/device/certificate/$metadata

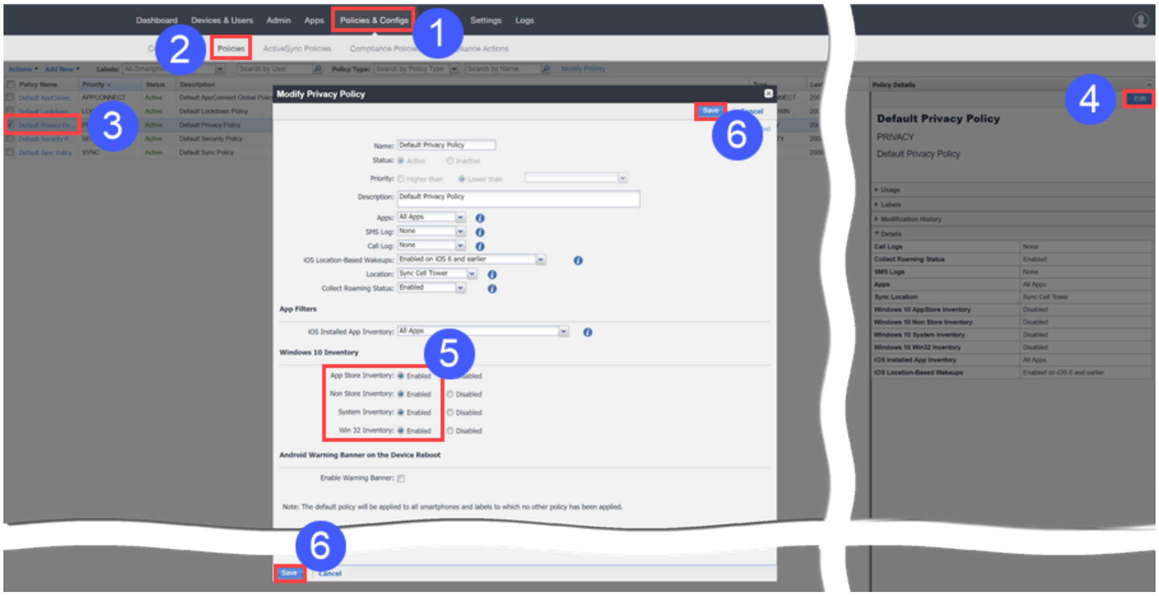
Example response

```
[
    {
        "name": "identity",
        "dataType": "boolean",
        "readOnly": true
    },
    {
        "name": "notAfter",
        "dataType": "long",
        "readOnly": true
    },
    {
        "name": "notBefore",
        "dataType": "long",
        "readOnly": true
    },
    {
        "name": "issuer",
        "dataType": "string",
        "readOnly": true
    },
    {
        "name": "serialNumber",
        "dataType": "string",
        "readOnly": true
    },
    {
        "name": "subject",
        "dataType": "string",
        "readOnly": true
    },
    {
        "name": "thumbPrint",
        "dataType": "string",
        "readOnly": true
    }
]
```

Enabling Windows 10 app inventory reporting on Ivanti EPMM

To enable app inventory reporting for Windows 10:

1. Log into Ivanti EPMM, and then select **Policies and Configs**.



Policy Name	Priority ▴	Status	Description
Default AppConne...	APPCONNECT	Active	Default AppConnect Global Polic
Default Lockdown...	LO	Active	Default Lockdown Policy
Default Privacy Po...	P	Active	Default Privacy Policy
Default Security P...	SE	Active	Default Security Policy
Default Sync Policy	SYNC	Active	Default Sync Policy

Modify Privacy Policy

Name: Default Privacy Policy

Status: ● Active ○ Inactive

Priority: ○ Higher than ● Lower than

Description: Default Privacy Policy

Apps: All Apps

SMS Log: None

Call Log: None

iOS Location-Based Wakeups: Enabled on iOS 6 and earlier

Location: Sync Cell Tower

Collect Roaming Status: Enabled

App Filters

iOS Installed App Inventory: All Apps

Windows 10 Inventory

App Store Inventory: ● Enabled ○ Disabled

Non Store Inventory: ● Enabled ○ Disabled

System Inventory: ● Enabled ○ Disabled

Win 32 Inventory: ● Enabled ○ Disabled

Android Warning Banner on the Device Reboot

Enable Warning Banner: □

Note: The default policy will be applied to all smartphones and labels to which no other policy has been applied.

Policy Details

Default Privacy Policy

PRIVACY

Default Privacy Policy

▸ Usage

▸ Labels

▸ Modification History

▾ Details

Call Logs	None
Collect Roaming Status	Enabled
SMS Logs	None
Apps	All Apps
Sync Location	Sync Cell Tower
Windows 10 AppStore Inventory	Disabled
Windows 10 Non Store Inventory	Disabled
Windows 10 System Inventory	Disabled
Windows 10 Win32 Inventory	Disabled
iOS Installed App Inventory	All Apps
iOS Location-Based Wakeups	Enabled on iOS 6 and earlier

2. Select **Policies**.
3. Select **Privacy Policy**.
4. Click **Edit**.
5. Enable the following Windows 10 inventory settings:
   - App Store Inventory
   - Non Store Inventory
   - System Inventory
   - Win 32 Inventory
6. Click one of the **Save** buttons.