Credential Escrow Gateway

VMware Workspace ONE UEM 2111

vmware®

You can find the most up-to-date technical documentation on the VMware website at:

https://docs.vmware.com/

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

Contents

Overview of Credential Escrow Gateway

1

Utilizing Credential Escrow Gateway (CEG) through VMware Workspace ONE UEM simplifies the distribution of SMIME certificates to iOS and Android devices by not uploading the SMIME certificate to Workspace ONE Unified Endpoint Management. It provides automation of the SMIME delivery, with end-to-end public key encryption for consumption using native, and 3rd party email clients.

Note: To use the Credential Escrow Gateway, you must contact your VMware account representative. Your account representative will engage the VMware Professional Services Office and Product Management.

Prerequisites

The following prerequisites must be met to use the Credential Escrow Gateway with VMware Workspace ONE UEM.

- Credential Escrow Gateway 1.3.0 or later
- For single account
  - Workspace ONE UEM 2007 or later
  - Android or iOS Boxer 5.19 or later
- For multi-managed account
  - Workspace ONE UEM 2008 or later
  - Android of iOS Boxer 5.21 or later
- Windows 10 devices should be on 1909 build 8363.693 or later
- A webhook that subscribes to event notification and update certificate provider with DeviceUUID and EnrollmentUserUUID.

- A Certificate Provider receives Event Notifications from webhook and forwarding the information to Certificate Authority to generate SMIME cert for a specific user and send that certificate to escrow gateway. For more details, see Certificate Provider design specification. You are responsible for building a webhook and a certificate provider.

Note

- Workspace ONE UEM 2010 or later and Credential Escrow Gateway 1.4.0 supports event-based driven certificate checks by delivering escrow profiles in a faster manner based on enrollment date of the device.
- If you are on Workspace ONE UEM 2009 and below, then the cert status check runs every four hours, with a maximum retry count of 75. If the cert provider fails to upload after 12.5 days, then the profile install fails and requires a manual re-install.
- Turn on encryption using a Smart Group and a profile for existing devices ahead of time.

Requirements and Architecture 2

Workspace ONE UEM utilizes ACC and the Credential Escrow Gateway feature to send encrypted skeleton profiles back and forth between the SMIME certificate and Workspace ONE UEM. Learn more about the requirements and architecture behind the Credential Escrow Gateway feature.

System Requirements

The following system requirements are recommended for supporting 100,000 devices in Credential Escrow Gateway.

Escrow Gateway	Up to 100,000 devices	Notes
System Requirements	CPU - 4 cores	Per 8,000 devices, up to a maximum of 32,000 devices (8 CPU/ 16 GB RAM) per application server.
Memory	16 GB	Escrow Gateway uses Redis as it's primary data store. Given Redis is an in-memory database, it is essential to allocate sufficient memory. By default, EG sets the upper bound on the total memory used by Redis to be 8 GB. The remaining memory allocation is available for use by other system processes and applications.
Disk Space	40 GB	
Load	*Continuous load of 40 requests per second.	*If you adhere to the recommendations in this table.

Required Network Configurations

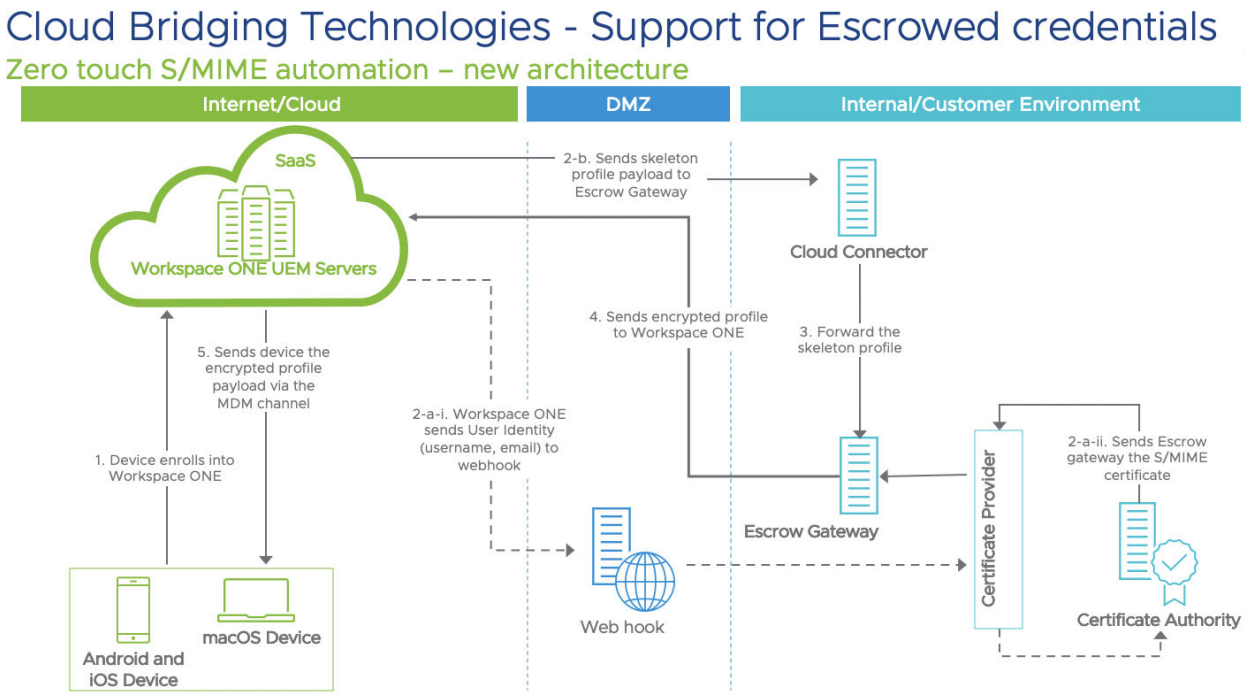
Credential Escrow Gateway uses port 443 for inbound API requests from ACC and an outbound connection to send completed and encrypted profiles (skeleton profile from UEM hydrated with Customer uploaded SMIME certificate) back to WS1 UEM.

Source Component	Destination Component	Protocol	Port
ACC	AWCM	HTTPS (inbound to AMCW)	2001 - OnPrem
ACC	AWCM	HTTPS (inbound to AWCM)	443 - SaaS
ACC	Escrow Gateway	HTTPS (inbound to Escrow Gateway)	443
AWCM	ACC	HTTPS (in-bound to ACC)	443
Escrow Gateway	Workspace ONE UEM API	HTTPS (outbound from Escrow Gateway)	443
PKI/Certificate Authority	Escrow Gateway	Open Firewall	

Note Since the request is going through ACC, please make sure you enable "All Other Components" under Groups & Settings > All Settings > System > Enterprise Integration > Cloud Connector > Advanced

Escrow Gateway Architecture Diagram

The following diagram illustrates the Credential Escrow Gateway components and how those components work with your environment.



Installing Credential Escrow Gateway

3

Installing Credential Escrow Gateway simplifies the distribution of SMIME certificates to iOS and Android devices by not uploading the SMIME certificate to Workspace ONE Unified Endpoint Management. Learn more about installing Workspace ONE UEM Credential Escrow Gateway.

The VA is delivered as an OVA file, which is used to create a dedicated virtual machine (VM) with Credential Escrow Gateway pre-installed. The creation process of the VM also asks for certain configuration parameters, such as the root user's password, network interface card (NIC) configuration, and more.

Create Credential Escrow Gateway VA

1 Download the VA using the provided link. This is a file with .ova extension.

2 Create a new VM using a VM tool such as VMware vSphere or Fusion.

   a In Fusion, use menu **new > import > choose file**

   b In vSphere, use menu **Deploy OVF Template > Local file > Choose file**

3 The VM initialization process asks you to validate and customize some parameters, such as the root user's password, how many NIC cards to create, the IP configuration of the NIC cards, whether to enable root user's SSH access, etc.

4 During the testing phase, it is recommended to turn on SSH access for the root user because SSH access is easier to use than native console or web console.

Book and Log Into VM

1 Power on the VM after installation (Credential Escrow Gateway VA is a server with a text-based console).

2 The first boot pauses a few seconds during the agent initialization process to unpack and start all packages used by Credential Escrow Gateway.

3 Log in using root credentials

Validate Credential Escrow Gateway (CEG)

Initial health checks use the `-k` flag to accept CEG's built-in TLS server certificate. Learn more about customizing the TLS server certificate in the Chapter 4 Configuring Credential Escrow Gateway section later in this guide.

1 Local CEG health check

   a Log into the VA using a console.

   Issue the following command from the console:

   ```
   curl -k https://localhost/v1/hc
   ```

   The output should look something like:

   ```
   {"host":"6aca8416535f"}
   ```

   This is the hash id of the container running CEG API.

2 Remote CEG health check

   a Get the IP address of CEG VA from the console

      1 use the `ifconfig` command

   b Run a health check from a different machine using the following command:

   ```
   curl -k https://{CEG VA's ip address}/v1/hc
   ```

   c You should see the same result as in the local health check.

Configuring Credential Escrow Gateway

4

Configuring the Credential Escrow Gateway creates a trust and secure communication channel between client and host. Learn more about configuring Workspace ONE UEM Credential Escrow Gateway.

While following the steps to configure Credential Escrow Gateway, there are several general considerations to keep in mind.

- The CEG API requires mutual TLS authentication (mTLS).
- The CEG API server must present to the client a proper server-side certificate for
- TLS handshake.
- API requests to Credential Escrow Gateway must present a client certificate whose thumbprint is allowed by CEG.

Note To generate certificate trust between Credential Escrow Gateway and ACC, the certificate needs to be generated via OpenSSL. Certificate generated by windows PowerShell for trust between Credential Escrow Gateway and Certificate Authority works fine.

Configure CEG server's TLS certificate

Credential Escrow Gateway needs a server certificate for API calls over HTTPS. The CA of this certificate needs to be trusted by ACC in order to establish TLS communication between ACC and CEG. The following are steps to installing the TLS certificate on CEG.

Configuration certificates used in Credential Escrow Gateway (CEG):

- Client Certs: These certificates are authorized certs that client can use toinvoke CEG APIs
- Thumbprints of client certs are pinned on CEG.
- CEG uses the list of pinned certificate thumbprints to authorize API access.
- SSL cert: This certificate is used for binding https (443) port. The API application runs on port 443.
- SSL certificate will be presented to client when it accesses CEG APIs over https.
- CEG virtual appliance is shipped with self-signed SSL cert for localhost.

The Credential Escrow Gateway exposes a set of configuration APIs (on port 5002), which will allow a VA admin to query and override existing configuration certificates. The next steps will need to be performed from a terminal logged into the VA since the 5002 port is blocked by the Virtual Appliance's firewall for incoming requests. Please keep the localhost:5002 as seen in the examples below.

1 Health check: This allows you to query health check of the configuration API endpoint.

```
curl -k -L -X GET 'https://localhost:5002/v1/hc'
```

2 Get configuration: This allows you to query existing certificate configuration for the CEG API endpoint.

```
curl -k -L -X GET 'https://localhost:5002/v1/configuration'
```

3 Put configuration: This allows you to override existing certificate configuration for the CEG API endpoint.

```
curl -k -L -X PUT 'https://localhost:5002/v1/configuration' \ -H 'Content-
Type: application/json' \ --data-raw '{ "client_certificate_thumbprints":
[ "{client certificate thumbprints} -> This should match up with
the client cert you installed on your ACC" ], "server_certificate":
{ "pkcs12": "{base64-encoded cert string of your tls cert}", "password":
"{certificate password1}" } }'
```

4 Afterwards you need to run the following script to restart docker services for the configuration changes to take effect

```
docker stack rm ceg
```

```
docker stack deploy -c /opt/vmware/docker/ceg/docker-compose.yml ceg
```

Configure client certificate for mTLS

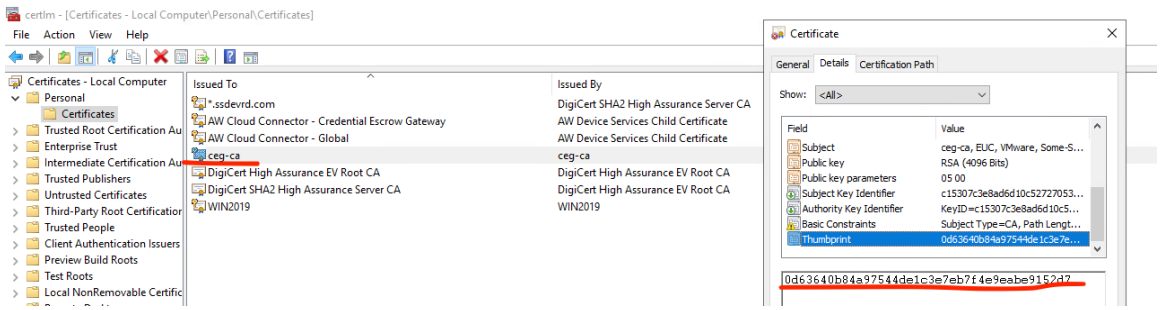
Because the API requests issued by Workspace ONE UEM to Credential Escrow Gateway (CEG) are proxied by ACC, ACC needs to have the client certificate used for mTLS. Workspace ONE UEM specifies which certificate to use, and the specified certificate must also be allowed by Credential Escrow Gateway.

1 Choose or install a client certificate on ACC

   a Log onto ACC's Windows desktop.

   b Launch **Manage Computer Certificates** by typing in the text after clicking the Windows key.

   c In the **Personal Certificate** store, you can select or install a certificate to be used for mTLS client authentication.



d The requirements for this certificate include:

1 Must have a private key

2 The key usages are appropriate for client authentication

3 It must be a root certificate because EG cannot obtain the revocation list for a non-root certificate.

Note Since Credential Escrow Gateway validates the certificate only by its thumbprint, self-signed certificates do work.

With this certificate in place, write down its thumbprint. Keep in mind that copying the thumbprint from this dialog can contain the NUL '\0' character at the beginning or end, causing problems if you paste it elsewhere.

Configure WS1 UEM to use the selected client certificate

This configuration is done with UEM API's.

1 Use the following API to get the Organization Group's UUID. This UUID is required to make the configuration change.

```
curl -i -X GET \-H 'Content-Type: application/json'\-H 'aw-tenant-code: {API access
key provisioned from Settings->System->Advanced->API} \-H 'Authorization: Basic {base64
encoding of admin-username:password}'\'https://{WS1 UEM hostname}/api/v1/system/groups/
{the OG's integer id}'
```

The Organization Group's integer id is displayed in the web browser's address bar when you view the Organization Group's details in UEM Console from **Groups & Settings > Organizations Groups > Details**.

2 The response has a UUID field:

```
"Uuid":"6eea71da-d8ef-4e51-b407-d22a0e41336a"
```

Copy the UUID value and use it in the following configuration query:

```
curl -i -X GET \-H 'Content-Type: application/json'\-H 'aw-tenant-code: {API access
key provisioned from Settings->System->Advanced->API}'\-H 'Authorization: Basic {base64
encoding of admin-username:password}'\'https://{WS1 UEM hostname}/api/v1/system/groups/{OG
UUID}/escrow-gateway-settings'
```

3 To make the configuration change:

```
curl -i -X PUT \-H 'Content-Type: application/json'\-H 'aw-tenant-code: {API access
key provisioned from Settings->System->Advanced->API}'\-H 'Authorization: Basic {base64
encoding of admin-username:password}'\--data-raw '{"gateway_url":"https://{CEG hostname}",
"client_cert_thumbprint":"{the client certificate's thumbprint"}' \'https://{WS1 UEM
hostname}/api/v1/system/groups/{OG UUID}/escrow-gateway-settings'
```

Allow the client certificate in Credential Escrow Gateway

1 Log into Credential Escrow Gateway VA using SSH or VM console

2 Use an editor such as vim to edit the file:

`/opt/vmware/docker/ceg/compose-config/authorized-client-certs.env`

Here is an example of adding an allowed certificate thumbprint:

`AuthorizedClientCertThumbprints__0=0D63640B84A97544DE1C3E7EB7F4E9EABE9152D7`

3 Save the changes and reboot Credential Escrow Gateway VA

Test client certificate

1 To test client certificate configuration:

a Log onto ACC Windows Desktop

b Export the client certificate to a pfx file

c From a bash terminal (e.g., Git Windows' bash terminal), run this command:

```
curl -iv 'https://{CEG hostname}/v1/hc'\--cert path-to-exported-client-certificate-
file-in-p12-format:cert-password \--cert-type p12
```

The result should be success (200 OK) and should contain Credential Escrow Gateway's host name in hash format.

Logs for Credential Escrow Gateway

1 Application logs are in `/var/log/vmware/docker/ceg/`

Configuring logging level, Encryption, and Certificate retention period

All configurations for Escrow Gateway can be updated through the .env files located in `/opt/vmware/docker/ceg/compose-config` directory. The following are the file names and the configurations available through them.

File Name	Configuration Name	Services requiring docker-compose.yml change	Description	Allowed Values	Default
logging.settings.env	Serilog__MinimumLevel__Default	api-0 api-1 skeleton-profile-consumer completed-profile-consumer	Change the application logging level.	Verbose Debug Information Warning Error Fatal	Information
redis.encryption.settings.env	EncryptionConfiguration__EnableEncryption	api-0 api-1 skeleton-profile-consumer completed-profile-consumer	If true, encrypts SMIME certificates before storing them into Redis. By default, this is disabled. NOTE: Needs to be set at application start. Will lead to data corruption otherwise.	True False	False
redis.encryption.settings.env	EncryptionConfiguration__EncryptionKey	api-0 api-1 skeleton-profile-consumer completed-profile-consumer	If EncryptionConfiguration__EnableEncryption property is set to true, provide a base64 encoded key to use for encryption. NOTE: This property should be set only once at application start along with the EncryptionConfiguration__EnableEncryption property. Changes in this value in future will lead to errors/data corruption.	tbase64 string representation of an encryption key	Empty. Please uncomment the line and add the desired value.

File Name	Configuration Name	Services requiring docker-compose.yml change	Description	Allowed Values	Default
redis.retention.settings.env NOTE: If this setting is changed, it will only be applied for future uploads. All existing data will use the retention setting from when you actually uploaded the data.	smimeCertificateRetention__UseCertificateExpiryAsRetention	api-0 api-1 skeleton-profile-consumer completed-profile-consumer	If true, Certificate expiration date is used as retention period. If false, the value set at redisKeyRetention__DefaultKeyExpiryTimeInDays is used.	true false	true
redis.retention.settings.env	redisKeyRetention__DefaultKeyExpiryTimeInDays	api-0 api-1 skeleton-profile-consumer completed-profile-consumer	The default retention period used for smime_certificates if smimeCertificateRetention__UseCertificateExpiryAsRetention is set to false.	±5.0 × 10−324 to ±1.7 × 10308	3
redis.retention.settings.env	smimeCertificateRetention__DeleteCertificateAfterConsumption	api-0 api-1 skeleton-profile-consumer completed-profile-consumer	If true, overrides the above mentioned retention period settings and deletes certificates as soon as it is used to complete a profile request.	true false	false

For any of the above changes to take effect, the following steps need to be executed after updating the configurations.

```
docker stack rm ceg
docker stack deploy -c /opt/vmware/docker/ceg/docker-compose.yml ceg
```

Configuring Disaster Recovery

5

Restore the Credential Escrow Gateway on another location with minimal steps through a unique disaster recovery procedure. When enabling Disaster Recovery in Escrow Gateway, we use an Active Passive setup with a common network file store. At any time, both active and passive servers cannot be turned on and be mounted to the NFS mountpoint (passive node must be in an off state). Other configurations lead to a loss of requests and possibly a corruption of data in the shared store. Currently disaster recovery is only supported from Credential Escrow Gateway 1.2.0.

Server type	OS	Version	IP	DNS	Mount points
NFS Server	Ubuntu	18.04.4 LTS (Bionic Beaver)	172.16.84.234		/home/eg/redis/data /home/eg/composeconfig
EG-Active Server	Photon	EUC Credential Escrow Gateway 1.2.0	172.16.70.52	https://beta1eg.ssdevrd.com/	/opt/vmware/docker/ceg/redis/data /opt/vmware/docker/ceg/compose-config
EG- Passive Server	Photon	EUC Credential Escrow Gateway 1.2.0	172.16.70.128	https://beta2eg.ssdevrd.com/	/opt/vmware/docker/ceg/redis/data /opt/vmware/docker/ceg/compose-config
Workspace ONE UEM environment	Windows	2008	172.16.99.159	https://egmma2007.ssdevrd.com/	N/A

Active Server Setup

NFS Server Setup

1 Create mount directories with appropriate permissions (Redis and other services running on EG requires read and write permission for user 1001 to read/write to the mounted folders).

```
mkdir -p /home/eg/redis/data
chown -R 1001:1001/home/eg/redis/data
chmod -R 755/home/eg/redis/data

mkdir -p /home/eg/composeconfig
chown -R 1001:1001/home/eg/composeconfig
chmod -R 755/home/eg/composeconfig
```

2 Copy the initial configuration information from EG's active server to NFS server. Only copy once for a given NFS server. The following example uses SCP for the copy.

```
scp -r /opt/vmware/docker/ceg/compose-config/* admin@172.16.84.234:/home/eg/composeconfig
```

3 Update /etc/exports

```
vi /etc/exports
```

Add the following lines to the end of the file by specifying the IP of Active EG server

```
/home/eg/redis/data  172.16.70.52(rw,sync,no_subtree_check)
/home/eg/composeconfig  172.16.70.52(rw,sync,no_subtree_check)
```

4 Restart NFS to apply changes

```
service nfs-kernel-server restart
```

Credential Escrow Gateway Active Server Setup:

1 Install nfs utils.

```
Tdnf install nfs-utils
```

2 Stop docker services.

```
docker stack rm ceg
```

3 Mount the file system to the NFS server by specifying the IP of the NFS server.

```
mount -t nfs 172.16.84.234:/home/eg/redis/data /opt/vmware/docker/ceg/redis/data
mount -t nfs 172.16.84.234:/home/eg/composeconfig /opt/vmware/docker/ceg/compose-config
```

4 Optional: Configure Encryption*. Refer to the escrow gateway configuration guide. With encryption enabled, the encryption configuration must be done on application startup and only once in the lifetime of a given EG setup. Configure both active and passive nodes to use the same encryption configuration which happens automatically when this setup guide is followed.

5 Start docker services.

```
docker stack deploy -c /opt/vmware/docker/ceg/docker-compose.yml ceg
```

6 Configure the Workspace ONE UEM instance to point to the Active Server by calling the Escrow Gateway Configuration API with the url of Active EG server.

```
curl --location --request PUT 'https://egmma2007.ssdevrd.com/api/system/groups/
96429a7f-6f42-4a17-a451-d487633d2336/escrow-gateway-settings'\
--header 'Content-Type: application/json'\
--header 'aw-tenant-code: XfGiwT8DxsMOopVdtJztHKc8b4DjiSknHF4cpdCQ9EU='\
--header 'Authorization: Basic YWRtaW5pc=='\
--data-raw '{
"gateway_url": "https://beta1eg.ssdevrd.com",
 "client_cert_thumbprint": <thumbprint>
}'
```

Fail-over Setup

NFS Server Setup

1 Update /etc/exports.

```
vi /etc/exports
```

Change the IP to point to Passive EG server.

```
/home/eg/redis/data  172.16.70.128(rw,sync,no_subtree_check)
/home/eg/composeconfig  172.16.70.128(rw,sync,no_subtree_check)
```

2 Restart nfs to apply changes.

```
service nfs-kernel-server restart
```

Credential Escrow Gateway Passive Server Setup:

1 Install nfs utils.

```
tdnf install nfs-utils
```

2 Stop docker services.

```
docker stack rm ceg
```

3 Mount the file system to the NFS server by specifying the IP of the NFS server.

```
mount -t nfs 172.16.84.234:/home/eg/redis/data /opt/vmware/docker/ceg/redis/data
mount -t nfs 172.16.84.234:/home/eg/composeconfig /opt/vmware/docker/ceg/compose-config
```

4 Start docker services.

```
docker stack deploy -c /opt/vmware/docker/ceg/docker-compose.yml ceg
```

5 Configure the Workspace ONE UEM instance to point to the Passive Server by calling the Escrow Gateway Configuration API by specifying the url of Passive EG server.

```
curl --location --request PUT 'https://egmma2007.ssdevrd.com/api/system/groups/
96429a7f-6f42-4a17-a451-d487633d2336/escrow-gateway-settings' \
--header 'Content-Type: application/json' \
--header 'aw-tenant-code: XfGiwT8DxsMOopVdtJztHKc8b4DjiSknHF4cpdCQ9EU=' \
--header 'Authorization: Basic YWRtaW5pc==' \
--data-raw '{
"gateway_url": "https://beta2eg.ssdevrd.com",
"client_cert_thumbprint": <thumbprint>
}'
```

Escrow Gateway Outbound Proxy Configuration

Credential Escrow Gateway (CEG) makes outbound API calls to Workspace ONE UEM to deliver encrypted payloads for S/MIME certificates or to inform UEM about the availability of S/MIME certificates. These outbound calls are made over HTTPS directly to UEM SaaS endpoint. When you have a proxy server that is in the call path from EG to SaaS UEM API, you need to use the following configuration to enable using proxy server in EG.

The following steps need to be done from a terminal off the EG VA.

1 Create an env file with this path `/opt/vmware/docker/ceg/compose-config/proxy_settings.env`. The content of this file looks like the following. Be sure to modify the values according to your environment.

```
# Proxy configuration for containers
HTTP_PROXY=http://127.0.0.1:3001
HTTPS_PROXY=https://127.0.0.1:3001
NO_PROXY=*.example.com
```

2 Modify `/opt/vmware/docker/ceg/docker-compose.yml` by inserting one new line for the new env file ONLY to this section

```
completed-profile-consumer: image:
'vmware.uem.escrowgateway.completedprofileconsumer:latest' env_file: - /opt/vmware/
docker/ceg/compose-config/redis.settings.env - /opt/vmware/docker/ceg/compose-config/
statsd.settings.env - /opt/vmware/docker/ceg/compose-config/proxy_settings.env networks: -
cegnet
```

Note Only the completed profile consumer needs this change.

3 Stop and restart the CEG service.

```
docker stack rm ceg
docker stack deploy -c /opt/vmware/docker/ceg/docker-compose.yml ceg
```

Configuring Boxer with Credential Escrow Gateway

6

Credential Escrow Gateway is supported for single account for Boxer 5.19 on iOS and Android. For multiple managed account, Boxer 5.21 on iOS and Android is supported by Credential Escrow Gateway. Learn more about how to enable Credential Escrow Gateway for Boxer with VMware Workspace ONE UEM.

Utilizing Boxer with Credential Escrow Gateway

Ensure that Credential Escrow Gateway is configured at the console level for the account so Boxer can get the certificates from Credential Escrow Gateway. To utilize Credential Escrow Gateway with Boxer 5.19, enable the Escrow Gateway switch by following the path below within your UEM console for iOS or Android.

1 Access your Boxer assignment within the UEM Console

2 Navigate to **Email Settings >> Configure S/MIME >> Escrow Gateway switch**

3 Turn on the Escrow Gateway switch

Creating a Credential Escrow Gateway Profile

7

Creating Credential Escrow Gateway profile Workspace ONE UEM gives the ability to send the skeleton profile to Workspace ONE UEM Credential Escrow Gateway to encrypt. This profile is encrypted, and only end-user devices can decrypt it using the private key. Learn more about how easy it is to create a profile with Credential Escrow Gateway.

Creating a Profile with Escrow Credentials

1 From the Profile page, click **Add Profile** and the desired platform

2 On the General Payload page give the profile a name and assign it to a smartgroup

3 On the Credentials Payload page click on **Escrow Gateway** and select **Signing Certificate**

4 To Add an Encryption Cert, click on the **+** button to add an additional credential payload and select **Escrow Gateway** and then **Encryption Certificate**.

5 Add Exchange ActiveSync details if needed and click **save and publish** to create the profile and assign to device.

6 During device enrollment, UEM sends a skeleton profile with placeholders for escrowed credentials to CEG via ACC.

This skeleton profile is identified by a combination of the user and device UUID. In the meantime, UEM fires a device enrollment event to the webhook defined by customer. The event's payload contains the user and device UUID.

7 The webhook should trigger customer's Certificate Provider to upload the required S/MIME certificates for the newly created profile.

8 See the next section about how to upload certificates.

Note Credential Escrow Gateway does not demand a particular order of profile creation and certificate upload; any of them can be done first.

Certificate Provider Design Specification

8

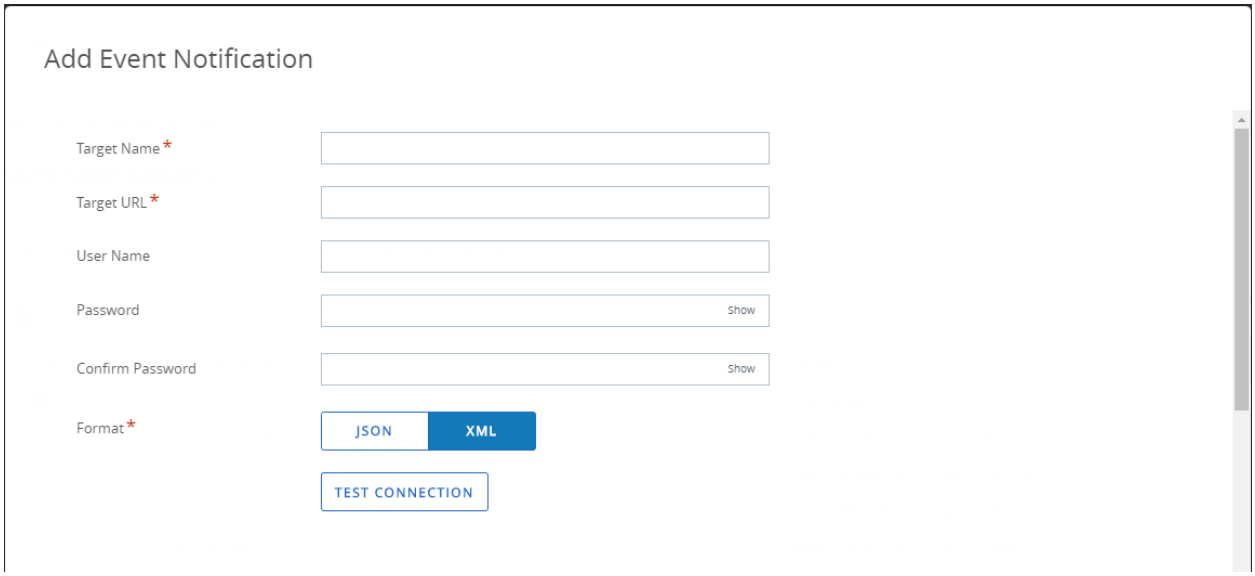
The Certificate Provider is responsible for listening to Event Notifications from UEM and forwarding the information to Certificate Authority to generate SMIME cert for a specific user and send that certificate to escrow gateway.

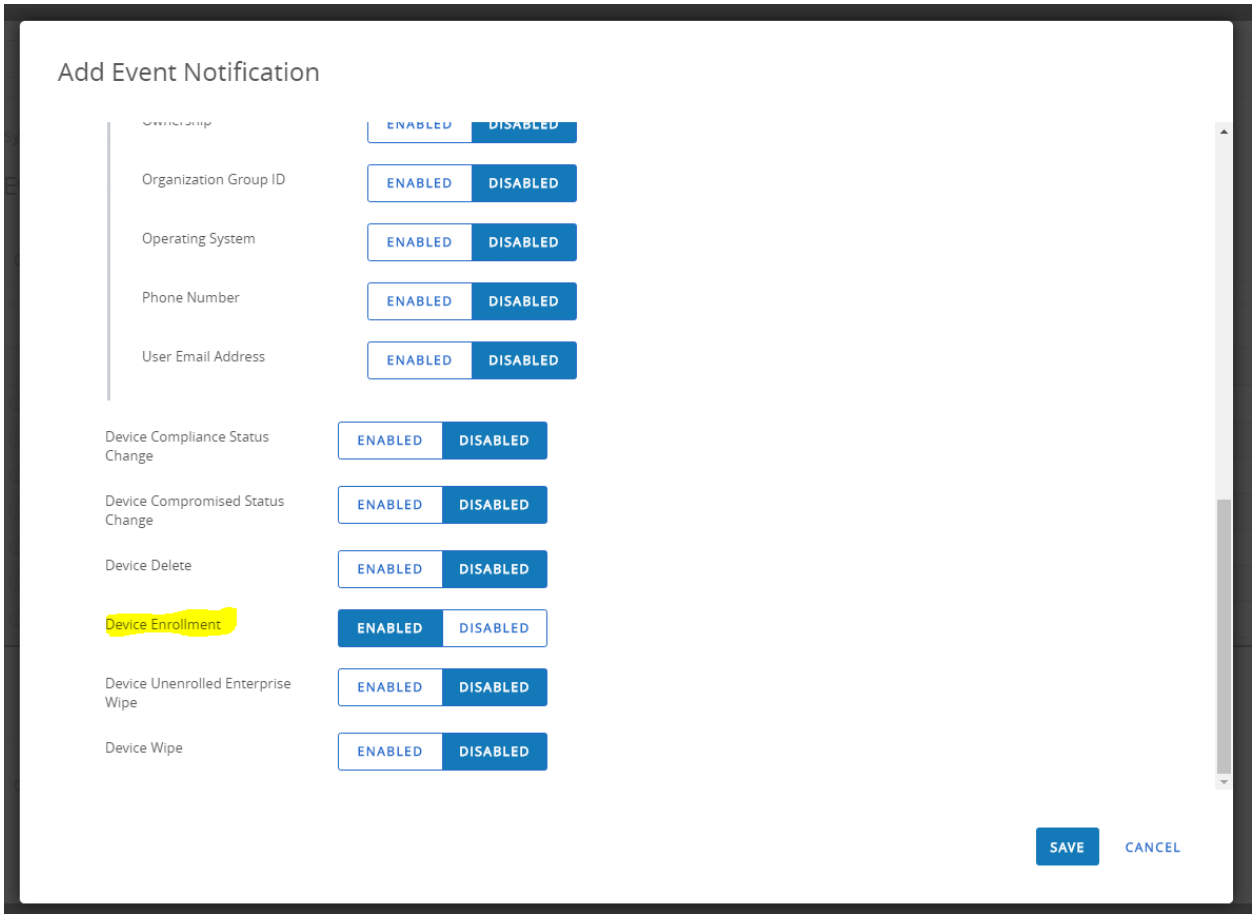
Event Notification Configuration

Customer must subscribe to the following events from UEM by configuring them in Admin Console.

Device Enrollment - this event will be fired when a new device enrolls.

The following example shows how to add event notification for Device Enrollment





The following is an example payload for an Enrollment event.

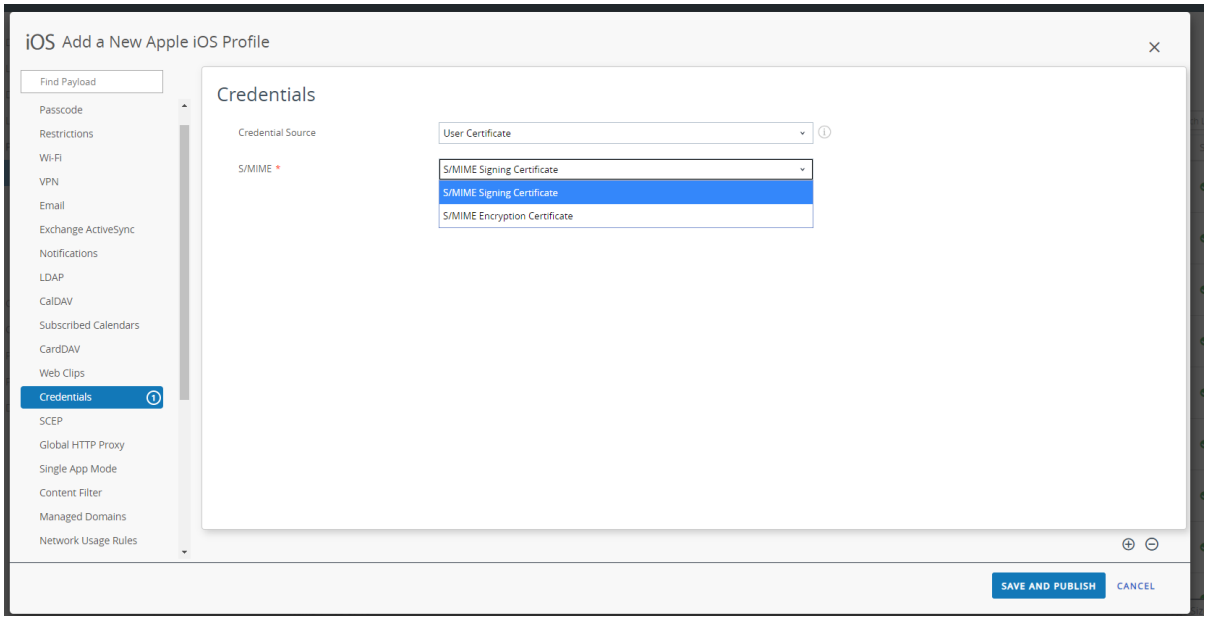

```
{ "EventId": 148,
"EventType": "MDM Enrollment Complete",
"DeviceId": 15,
"DeviceFriendlyName": "zenny iPod Touch Apple 12.4.5 GGK6", "EnrollmentEmailAddress":
"bmalinowski@vmware.com",
"EnrollmentUserName": "zenny",
"EventTime": "2020-02-24T16:52:25.9231303Z",
"EnrollmentStatus": "Enrolled",
"CompromisedStatus": "",
"CompromisedTimeStamp": "2020-02-24T16:52:28.9075371Z", "ComplianceStatus": "Compliant",
"PhoneNumber": "",
"Udid": "902dca357c02dbc1306ff99ffbf4b9c80218f3a3",
"SerialNumber": "CCQQ44BMGGK6",
"MACAddress": "000000000000",
"DeviceIMEI": "",
"EnrollmentUserId": 12,
"AssetNumber": "902dca357c02dbc1306ff99ffbf4b9c80218f3a3",
"Platform": "Apple",
"OperatingSystem": "12.4.5",
"Ownership": "CorporateDedicated",
"SIMMCC": "",
```

```
"CurrentMCC": "",
"OrganizationGroupName": "CEG",
"DeviceUUID": "f90f1abd-b914-4895-9b59-82e8aecf0e66", "EnrollmentUserUUID":
"fe13e814-7415-4fce-9f47-de5a55bfe57a"}
```

Certificate Provider Design

The following are the steps that need to be executed by the customer requesting Certificates to be provisioned through Escrow Gateway by configuring an EG credential profile.

1 Create a new Profile and configure a credential payload with Credential Source as Escrow Gateway and add Signing Certificate or Encryption Certificate or both. You can either publish the profile now or do so after uploading the certificates to EG using the steps below.



2 After receiving the Event Notification, the Certificate Provider should first request or provision the necessary certificate(s) that is to be delivered to the device.

   a Output: Client Certificate for a given User and/or Device combination.

3 The Certificate Provider should then upload the certificate(s) to the Escrow Gateway. This process can be asynchronous because the entire Escrow Gateway workflow is asynchronous. Escrow Gateway also requires mutual TLS authentication to service requests. Here we will construct a request using cURL.

   a Input variables for constructing the cURL request:

      1 Escrow Gateway Server URL - EG server that is setup in the customer's environment

      2 certificate.pfx - A client certificate that is in the approved client certificate list configured at EG used for mTLS authentication with Escrow Gateway

      3 password - Password for the certificate.pfx file used for mTLS

4 certificate-payload.json - a required json file containing the SMIME certificates to be uploaded to EG for a given User and Device combination

   a device_uuid - DeviceUUID value from the Event Notification payload

   b user_uuid - EnrollmentUserUUID value from the Event Notification payload

   c smime_certificates - json containing the following optional fields based on how the Profile is configured in Console in Step 1.

      - signing - An array of base64 encoded Signing certificates as required, generated in step 2.
      - encryption - An array of base64 encoded Encryption certificates as required, generated in step 2
      - archived - An array of expired signing/encryption certificates of a given user and/or device combination

```
Sample cURL to upload certificates to EG

curl -ik -X POST '<Escrow Gateway Server URL>/v2/certificates' \
--header 'Content-Type: application/json' \
--header 'Accept: application/json' \
--cert <certificate.pfx>:<password> \
--cert-type p12 \
--data <Sample certificate-payload.json>
```

b. Renew/replace SMIME certs.

The expiration of these certs are tracked by the customers. So they are responsible for uploading new certificates for the expiring certs.