# Qualys Cloud Platform (VM, PC) v8.x

## API Release Notes

Version 8.18.1

March 19, 2019 (revised May 31, 2019)

This new version of the Qualys Cloud Platform (VM, PC) includes improvements to the Qualys API. You'll find all the details in our user guides, available at the time of release. Just log in to your Qualys account and go to Help > Resources.

### What's New

New Support for HashiCorp Vault

Option Profile API - DTD/XSD Change

### URL to the Qualys API Server

Qualys maintains multiple Qualys platforms. The Qualys API server URL that you should use for API requests depends on the platform where your account is located.

| Account Location | API Server URL |
|---|---|
| Qualys US Platform 1 | https://qualysapi.qualys.com |
| Qualys US Platform 2 | https://qualysapi.qg2.apps.qualys.com |
| Qualys US Platform 3 | https://qualysapi.qg3.apps.qualys.com |
| Qualys US Platform 4 | https://qualysapi.qg4.apps.qualys.com |
| Qualys EU Platform 1 | https://qualysapi.qualys.eu |
| Qualys EU Platform 2 | https://qualysapi.qg2.apps.qualys.eu |
| Qualys India Platform 1 | https://qualysapi.qg1.apps.qualys.in |
| Qualys Private Cloud Platform | https://qualysapi.<customer_base_url> |

The Qualys API documentation and sample code use the API server URL for the Qualys US Platform 1. If your account is located on another platform, please replace this URL with the appropriate server URL for your account.

# New Support for HashiCorp Vault

| | |
|---|---|
| APIs affected | /api/2.0/fo/vault |
| New or Updated API | Updated |
| DTD or XSD changes | Yes |
| APIs affected | /api/2.0/fo/auth/windows/ |
| New or Updated API | Updated |
| DTD or XSD changes | Yes |
| APIs affected | /api/2.0/fo/auth/unix/ |
| New or Updated API | Updated |
| DTD or XSD changes | Yes |

Note - The earlier version of the Cloud Suite 8.18.1 API Release Notes failed to mention that we updated the Vault View DTD to include new elements.

This new vault type can be used to retrieve authentication credentials from a HashiCorp vault. We updated the authentication vault API (create, update, list, view) and the authentication record API (create, update, list) to support the new vault type. We updated the DTDs for listing Windows and Unix records, and the DTD for Vault View.

## Authentication Vault API

You can now create, update, list and view HashiCorp vaults.

### Create/Update HashiCorp Authentication Vault

Use the parameter “action=create” or “action=update” to create/update a new vault in your account.

| Parameter | Description |
|---|---|
| action=create | (Required) |
| title={value} | (Required) The vault title. |
| type={value} | (Required) Specify type=HashiCorp |
| comments={value} | (Optional) User defined comments. |
| url={value} | (Required) The HTTP or HTTPS URL to access the HashiCorp Vault HTTP API. |
| api_version{value} | (Optional) The HashiCorp Vault HTTP API version. This is v1 by default, which is the only supported version. |

| Parameter | Description |
|---|---|
| ssl_verify={0\|**1**} | (Optional) When set to 1 (the default), our service will verify the SSL certificate of the web server to make sure the certificate is valid and trusted. When set to 0, our service will not verify the certificate of the web server. |
| auth_type={value} | (Required to create vault, optional to update vault) HashiCorp Vault API supports three authentication types. First choose any one of the authentication method you want to use (Username/Password, Cert or App Role) and then provide login credentials for authenticating to the vault server via the HashiCorp Vault HTTP API.<br><br>Valid authentication values for API are: userpass, cert and approle. |

### Auth Type: Username/password

Choose the Username/Password authentication method to authenticate to the vault server with a username and password combination.

| Parameter | Description |
|---|---|
| path={value} | (Optional) The path for the Username/Password authentication method. The default path is auth/userpass but you can specify a custom path like auth/my-path. |
| username={value} | (Required to create and update vault) The user account that can access the vault server. |
| password={value} | (Required to create and update vault) The password for the user account. |

### Auth Type: Cert

Choose the Cert authentication method to authenticate to the vault server using SSL/TLS client certificates which are either signed by a CA (Certificate Authority) or self-signed. CA certificates are associated with a role name.

| Parameter | Description |
|---|---|
| path={value} | (Optional) The path for the Cert authentication method. The default path is auth/cert but you can specify a custom path like auth/my-path. |
| role_name={value} | (Required to create and update vault) The role associated with the CA certificate. |

| | |
|---|---|
| cert={value} | (Required to create and update vault) The client certificate for authentication. Enter the certificate block after the key block and be sure to include the first and last line (-----BEGIN CERTIFICATE----- and -----END CERTIFICATE-----).<br>For a create/update request, if the cert parameter is specified, then the private_key parameter must also be specified. |
| private_key={value} | (Required to create and update vault) The private key for authentication. Copy the contents of private key file (id_rsa) and be sure to include the first and last line (-----BEGIN PRIVATE KEY----- and -----END PRIVATE KEY-----). |
| passphrase{value} | (Optional) The private key passphrase, if the private key is encrypted. |

Auth Type: App Role

Choose the App Role authentication method to authenticate to the vault server with a vault-defined role.

| Parameter | Description |
|---|---|
| path={value} | (Optional) The path for the App Role authentication method. The default path is auth/approle but you can specify a custom path like auth/my-path. |
| role_id={value} | (Required to create and update vault) The role ID of the App Role you want to use for authentication. |
| secret_id={value} | (Optional) The secret ID of the App Role you want to use for authentication. |

### Sample - Create HashiCorp Vault with username and password

API request:

```
curl -u "USERNAME:PASSWORD" -H "X-Requested-With: Curl" -d
"action=create&type=HashiCorp&title=My HashiCorp
Vault&url=https://hashicorp.example.com/api&ssl_verify=0&auth_type=userpa
ss&username=user&password=abc123&comments=creating HashiCorp VAULT from
api"
"https://qualysapi.qualys.com/api/2.0/fo/vault/"
```

XML output:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE SIMPLE_RETURN SYSTEM
"https://qualysapi.qualys.com/api/2.0/simple_return.dtd">
<SIMPLE_RETURN>
  <RESPONSE>
```

```
    <DATETIME>2019-03-13T11:49:48Z</DATETIME>
    <TEXT>Success</TEXT>
    <ITEM_LIST>
      <ITEM>
        <KEY>ID</KEY>
        <VALUE>910196</VALUE>
      </ITEM>
    </ITEM_LIST>
  </RESPONSE>
</SIMPLE_RETURN>
```

### Sample - Create HashiCorp Vault with Cert

API request:

```
curl -u "USERNAME:PASSWORD" -H "X-Requested-With: Curl" -d
"action=create&type=HashiCorp&title=My HashiCorp
Vault&url=https://hashiCorp.example.com/api&ssl_verify=0&auth_type=cert&r
ole_name=testroot&cert=-----BEGIN CERTIFICATE-----
MIIEyjCCArKgAwIBAgIBAjANBgkqhkiG9w0BAQUFADCBljEbMBkGA1UEAwwSU2Nh-----END
CERTIFICATE-----&private_key=key&comments=creating HashiCorp VAULT from
api"
"https://qualysapi.qualys.com/api/2.0/fo/vault/"
```

XML output:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE SIMPLE_RETURN SYSTEM
"https://qualysapi.qualys.com/api/2.0/simple_return.dtd">
<SIMPLE_RETURN>
    <RESPONSE>
        <DATETIME>2019-03-06T00:36:12Z</DATETIME>
        <TEXT>Success</TEXT>
        <ITEM_LIST>
            <ITEM>
                <KEY>ID</KEY>
                <VALUE>27014</VALUE>
            </ITEM>
        </ITEM_LIST>
    </RESPONSE>
</SIMPLE_RETURN>
```

### Sample - Create HashiCorp Vault with App Role

API request:

```
curl -u "USERNAME:PASSWORD" -H "X-Requested-With: Curl" -d
"action=create&type=HashiCorp&title=My HashiCorp
Vault&url=https://hashicorp.example.com/api&ssl_verify=0&auth_type=approl
e&role_id=admin01&comments=creating HashiCorp VAULT from api"
"https://qualysapi.qualys.com/api/2.0/fo/vault/"
```

XML output:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE SIMPLE_RETURN SYSTEM
"https://qualysapi.qualys.com/api/2.0/simple_return.dtd">
<SIMPLE_RETURN>
    <RESPONSE>
        <DATETIME>2019-03-06T00:36:12Z</DATETIME>
        <TEXT>Success</TEXT>
        <ITEM_LIST>
            <ITEM>
                <KEY>ID</KEY>
                <VALUE>57016</VALUE>
            </ITEM>
        </ITEM_LIST>
    </RESPONSE>
</SIMPLE_RETURN>
```

### Sample - Update HashiCorp Vault

API request:

```
curl -u "USERNAME:PASSWORD" -H "X-Requested-With: Curl" -d
"action=update&id=910196&type=HashiCorp&title=My HashiCorp
Vault&api_version=v1&url=https://hashicorp.example.com/api&ssl_verify=0&a
uth_type=userpass&path=auth/my-
path&username=user&password=abc123&comments=updating HashiCorp VAULT from
api"
"https://qualysapi.qualys.com/api/2.0/fo/vault/"
```

XML output:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE SIMPLE_RETURN SYSTEM
"https://qualysapi.qualys.com/api/2.0/simple_return.dtd">
<SIMPLE_RETURN>
  <RESPONSE>
    <DATETIME>2019-03-13T12:04:19Z</DATETIME>
    <TEXT>Success</TEXT>
    <ITEM_LIST>
      <ITEM>
        <KEY>ID</KEY>
        <VALUE>910196</VALUE>
      </ITEM>
    </ITEM_LIST>
  </RESPONSE>
</SIMPLE_RETURN>
```

### Sample - List Authentication Vaults

API request:

```
curl -u "USERNAME:PASSWORD" -H "X-Requested-With: Curl" -d
"action=list"
"https://qualysapi.qualys.com/api/2.0/fo/vault/"
```

XML output:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE AUTH_VAULT_LIST_OUTPUT SYSTEM
"https://qualysapi.qualys.com/api/2.0/fo/vault/vault_output.dtd">
<AUTH_VAULT_LIST_OUTPUT>
  <RESPONSE>
    <DATETIME>2019-03-14T07:10:36Z</DATETIME>
    <STATUS>Success</STATUS>
    <COUNT>1</COUNT>
    <AUTH_VAULTS>
      <AUTH_VAULT>
        <TITLE><![CDATA[HASHCORP_API_VAULT_UPDATED]]></TITLE>
        <VAULT_TYPE><![CDATA[HashiCorp]]></VAULT_TYPE>
        <LAST_MODIFIED>
          <DATETIME>2019-03-13T12:04:29Z</DATETIME>
          <BY>amp_at</BY>
        </LAST_MODIFIED>
        <ID>910196</ID>
      </AUTH_VAULT>
    </AUTH_VAULTS>
  </RESPONSE>
</AUTH_VAULT_LIST_OUTPUT>
```

### Sample - View HashiCorp Vault

API request:

```
curl -u "USERNAME:PASSWORD" -H "X-Requested-With: Curl" -d
"action=view&id=929202"
https://qualysapi.qualys.com/api/2.0/fo/vault/
```

XML output:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE VAULT_OUTPUT SYSTEM
"https://qualysapi.qualys.com/api/2.0/fo/vault/vault_view.dtd">
<VAULT_OUTPUT>
  <RESPONSE>
    <DATETIME>2019-05-31T09:20:42Z</DATETIME>
    <VAULT_QUEST>
      <TITLE><![CDATA[HashiApprole]]></TITLE>
      <COMMENTS><![CDATA[]]></COMMENTS>
```

```
      <VAULT_TYPE><![CDATA[HashiCorp]]></VAULT_TYPE>
      <CREATED_ON>2019-03-27T12:44:19Z</CREATED_ON>
      <OWNER>ewr_at</OWNER>
      <LAST_MODIFIED>
        <DATETIME>2019-03-27T13:16:44Z</DATETIME>
        <BY>ewr_at</BY>
      </LAST_MODIFIED>
      <URL><![CDATA[https://hashicorp.example.com/api]]></URL>
      <SSL_VERIFY><![CDATA[1]]></SSL_VERIFY>
      <API_VERSION><![CDATA[v1]]></API_VERSION>
      <AUTH_TYPE><![CDATA[approle]]></AUTH_TYPE>
      <PATH><![CDATA[auth/hashi/new/12345/path]]></PATH>
      <ROLE_ID><![CDATA[11111111-1111-1111-1111-111111111111]]></ROLE_ID>
      <SECRET_ID><![CDATA[22222222-2222-2222-2222-
222222222222]]></SECRET_ID>
      <ID>929202</ID>
    </VAULT_QUEST>
  </RESPONSE>
</VAULT_OUTPUT>
```

## Updated DTD

<base_url>/api/2.0/fo/vault/vault_view.dtd

We added several elements (in bold) to the DTD.

```
<!-- QUALYS VAULT_VIEW DTD -->
<!-- $Revision: 00000 $ -->
<!ELEMENT VAULT_OUTPUT (REQUEST?,RESPONSE)>

<!ELEMENT REQUEST (DATETIME, USER_LOGIN, RESOURCE, PARAM_LIST?,
POST_DATA?)>
<!ELEMENT DATETIME (#PCDATA)>
<!ELEMENT USER_LOGIN (#PCDATA)>
<!ELEMENT RESOURCE (#PCDATA)>
<!ELEMENT PARAM_LIST (PARAM+)>
<!ELEMENT PARAM (KEY, VALUE)>
<!ELEMENT KEY (#PCDATA)>
<!ELEMENT VALUE (#PCDATA)>
<!-- if returned, POST_DATA will be urlencoded -->
<!ELEMENT POST_DATA (#PCDATA)>

<!ELEMENT RESPONSE (DATETIME, VAULT_QUEST)>

<!ELEMENT VAULT_QUEST (TITLE, COMMENTS, VAULT_TYPE, CREATED_ON?,
```

```
OWNER?, LAST_MODIFIED?, APPID?, APPKEY?, USERNAME?, URL?,
SSL_VERIFY?, DOMAIN?, API_USERNAME?, WEB_USERNAME?,
SERVER_ADDRESS?, PORT?, SAFE?, API_VERSION?, AUTH_TYPE?, PATH?,
ROLE_NAME?, ROLE_ID?, SECRET_ID?, APP_ID?, (UUID|ID))>
...
<!ELEMENT SAFE (#PCDATA)>
<!ELEMENT API_VERSION (#PCDATA)>
<!ELEMENT AUTH_TYPE (#PCDATA)>
<!ELEMENT PATH (#PCDATA)>
<!ELEMENT ROLE_NAME (#PCDATA)>
<!ELEMENT ROLE_ID (#PCDATA)>
<!ELEMENT SECRET_ID (#PCDATA)>
<!ELEMENT APP_ID (#PCDATA)>
<!ELEMENT LAST_MODIFIED (DATETIME, BY?)>
<!ELEMENT BY (#PCDATA)>
<!-- EOF -->
```

## Authentication Record API

Create, update, list authentication records with HashiCorp vaults. You can use HashiCorp vaults with Windows, Unix and Cisco records.

### Create Authentication Record

Choose the HashiCorp vault in your authentication record and provide details about the KV (Key-Value) secrets engine where your login credentials (secrets) are stored.

| Parameter | Description |
|---|---|
| action=create\|update | (Required) |
| login_type={value} | (Required to create/update vault information)<br>Specify login_type=vault to add vault information. By default, the parameter is set to basic. |
| vault_id={value} | (Required when action=create and login_type=vault) A vault ID. |
| vault_type={value} | (Required when action=create and login_type=vault)<br>Specify vault_type=HashiCorp |
| ips={value} | (Required to create record) The IP address(es) the server will log into using the record's credentials. Multiple entries are comma separated.<br>(Optional to update record) IPs specified will overwrite existing IPs in the record, and existing IPs will be removed. |
| secret_kv_path={value} | (Optional) The path of the secret engine. The default is "secret". |

| | |
|---|---|
| secret_kv_name={value} | (Required) The secret name which stores key-value pairs. |
| secret_kv_key={value} | (Required) The key name for identifying a specific key-value pair. |

### Sample - Create Unix record with HashiCorp Vault

API request:

```
curl -u "USERNAME:PASSWORD" -H "X-Requested-With: Curl" -d
"action=create&title=unix-api-pk-
test1&username=root&login_type=vault&vault_id=26011&vault_type=HashiCorp&
ips=10.10.10.3&secret_kv_path=path/secret/unixapi/test1&secret_kv_name=se
cret-name1&secret_kv_key=secret-key1"
"https://qualysapi.qualys.com/api/2.0/fo/auth/unix/"
```

add_params.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<UNIX_AUTH_PARAMS>
 <PRIVATE_KEY_CERTIFICATES>
 <PRIVATE_KEY_CERTIFICATE>
 <PRIVATE_KEY_INFO type="vault">
 <DIGITAL_VAULT>
 <VAULT_TYPE>HashiCorp</VAULT_TYPE>
 <VAULT_ID>26004</VAULT_ID>
<SECRET_KV_NAME><![CDATA[secret_name]]></SECRET_KV_NAME>
<SECRET_KV_KEY><![CDATA[secret_key]]></SECRET_KV_KEY>
 </DIGITAL_VAULT>
 </PRIVATE_KEY_INFO>
 <PASSPHRASE_INFO type="vault">
 <DIGITAL_VAULT>
 <VAULT_USERNAME>root</VAULT_USERNAME>
 <VAULT_TYPE>HashiCorp</VAULT_TYPE>
 <VAULT_ID>26004</VAULT_ID>
<SECRET_KV_PATH><![CDATA[auth/approle/test]]></SECRET_KV_PATH>
<SECRET_KV_NAME><![CDATA[secret_name]]></SECRET_KV_NAME>
<SECRET_KV_KEY><![CDATA[secret_key]]></SECRET_KV_KEY>
 </DIGITAL_VAULT>
 </PASSPHRASE_INFO>
 </PRIVATE_KEY_CERTIFICATE>
 </PRIVATE_KEY_CERTIFICATES>
</UNIX_AUTH_PARAMS>
```

XML output:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE BATCH_RETURN SYSTEM
"https://qualysapi.qualys.com/api/2.0/batch_return.dtd">
<BATCH_RETURN>
```

```
    <RESPONSE>
        <DATETIME>2019-03-05T00:09:43Z</DATETIME>
        <BATCH_LIST>
            <BATCH>
                <TEXT>Successfully Created</TEXT>
                <ID_SET>
                    <ID>76013</ID>
                </ID_SET>
            </BATCH>
        </BATCH_LIST>
    </RESPONSE>
</BATCH_RETURN>
```

### Sample - List Unix authentication records

API request:

```
curl -u "USERNAME:PASSWORD" -H "X-Requested-With: Curl" -d
"action=list&ids=76013"
"https://qualysapi.qualys.com/api/2.0/fo/auth/unix/">
```

XML output:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE AUTH_UNIX_LIST_OUTPUT SYSTEM
"https://qualysapi.qualys.com/api/2.0/fo/auth/unix/auth_unix_list_output.
dtd">
<AUTH_UNIX_LIST_OUTPUT>
    <RESPONSE>
        <DATETIME>2019-03-05T00:18:28Z</DATETIME>
        <AUTH_UNIX_LIST>
            <AUTH_UNIX>
                ...
                <CLEARTEXT_PASSWORD>0</CLEARTEXT_PASSWORD>
                <PRIVATE_KEY_CERTIFICATE_LIST>
                    <PRIVATE_KEY_CERTIFICATE>
                        <ID>12008</ID>
                        <PRIVATE_KEY_INFO type="vault">
                            <DIGITAL_VAULT>
                                <DIGITAL_VAULT_ID>
                                    <![CDATA[26004]]>
                                </DIGITAL_VAULT_ID>
                                <DIGITAL_VAULT_TYPE>
                                    <![CDATA[HashiCorp]]>
                                </DIGITAL_VAULT_TYPE>
                                <DIGITAL_VAULT_TITLE>
                                    <![CDATA[hashi1]]>
                                </DIGITAL_VAULT_TITLE>
                                <VAULT_SECRET_KV_PATH>
                                    <![CDATA[secret]]>
```

```
                        </VAULT_SECRET_KV_PATH>
                        <VAULT_SECRET_KV_NAME>
                            <![CDATA[secret_name]]>
                        </VAULT_SECRET_KV_NAME>
                        <VAULT_SECRET_KV_KEY>
                            <![CDATA[secret_key]]>
                        </VAULT_SECRET_KV_KEY>
                    </DIGITAL_VAULT>
                </PRIVATE_KEY_INFO>
                <PASSPHRASE_INFO type="vault">
                    <DIGITAL_VAULT>
                        <DIGITAL_VAULT_ID>
                            <![CDATA[26004]]>
                        </DIGITAL_VAULT_ID>
                        <DIGITAL_VAULT_TYPE>
                            <![CDATA[HashiCorp]]>
                        </DIGITAL_VAULT_TYPE>
                        <DIGITAL_VAULT_TITLE>
                            <![CDATA[hashi1]]>
                        </DIGITAL_VAULT_TITLE>
                        <VAULT_USERNAME>
                            <![CDATA[root]]>
                        </VAULT_USERNAME>
                        <VAULT_SECRET_KV_PATH>
                            <![CDATA[auth/approle/test]]>
                        </VAULT_SECRET_KV_PATH>
                        <VAULT_SECRET_KV_NAME>
                            <![CDATA[secret_name]]>
                        </VAULT_SECRET_KV_NAME>
                        <VAULT_SECRET_KV_KEY>
                            <![CDATA[secret_key]]>
                        </VAULT_SECRET_KV_KEY>
                    </DIGITAL_VAULT>
                </PASSPHRASE_INFO>
            </PRIVATE_KEY_CERTIFICATE>
        </PRIVATE_KEY_CERTIFICATE_LIST>
        ...
```

## DTD Updates

We added VAULT_SECRET_KV_PATH?, VAULT_SECRET_KV_NAME?, VAULT_SECRET_KV_KEY to the Windows and Unix Authentication List Output DTDs. The Cisco authentication record uses Unix Authentication List Output DTD.

### Windows Authentication Records List Output DTD

<baseURL>/api/2.0/fo/auth/windows/auth_windows_list_output.dtd

```
<!-- QUALYS AUTH_WINDOWS_LIST_OUTPUT DTD -->
<!ELEMENT AUTH_WINDOWS_LIST_OUTPUT (REQUEST?, RESPONSE)>
...
<!ELEMENT LOGIN_TYPE (#PCDATA)>
<!ELEMENT DIGITAL_VAULT (DIGITAL_VAULT_ID, DIGITAL_VAULT_TYPE,
DIGITAL_VAULT_TITLE, VAULT_FOLDER?, VAULT_FILE?,
VAULT_SECRET_NAME?, VAULT_SYSTEM_NAME?, VAULT_EP_NAME?,
VAULT_EP_TYPE?, VAULT_EP_CONT?, VAULT_NS_TYPE?, VAULT_NS_NAME?,
VAULT_ACCOUNT_NAME?, VAULT_AUTHORIZATION_NAME?,
VAULT_TARGET_NAME?, VAULT_SECRET_KV_PATH?, VAULT_SECRET_KV_NAME?,
VAULT_SECRET_KV_KEY?)>
<!ELEMENT DIGITAL_VAULT_ID (#PCDATA)>
<!ELEMENT DIGITAL_VAULT_TYPE (#PCDATA)>
<!ELEMENT DIGITAL_VAULT_TITLE (#PCDATA)>
<!ELEMENT VAULT_FOLDER (#PCDATA)>
<!ELEMENT VAULT_FILE (#PCDATA)>
<!ELEMENT VAULT_SECRET_NAME (#PCDATA)>
<!ELEMENT VAULT_SYSTEM_NAME (#PCDATA)>
<!ELEMENT VAULT_EP_NAME (#PCDATA)>
<!ELEMENT VAULT_EP_TYPE (#PCDATA)>
<!ELEMENT VAULT_EP_CONT (#PCDATA)>
<!ELEMENT VAULT_NS_TYPE (#PCDATA)>
<!ELEMENT VAULT_NS_NAME (#PCDATA)>
<!ELEMENT VAULT_ACCOUNT_NAME (#PCDATA)>
<!ELEMENT VAULT_AUTHORIZATION_NAME (#PCDATA)>
<!ELEMENT VAULT_TARGET_NAME (#PCDATA)>
<!ELEMENT VAULT_SECRET_KV_PATH (#PCDATA)>
<!ELEMENT VAULT_SECRET_KV_NAME (#PCDATA)>
<!ELEMENT VAULT_SECRET_KV_KEY (#PCDATA)>
....
<!-- EOF -->
```

### Unix Authentication Records List Output DTD

<baseURL>/api/2.0/fo/auth/unix/auth_unix_list_output.dtd

```
<!-- QUALYS AUTH_UNIX_LIST_OUTPUT DTD -->
<!ELEMENT AUTH_UNIX_LIST_OUTPUT (REQUEST?, RESPONSE)>
....
<!ELEMENT CERTIFICATE EMPTY>
<!ATTLIST CERTIFICATE type (x.509|openssh) #REQUIRED>
<!ELEMENT DIGITAL_VAULT (DIGITAL_VAULT_ID, DIGITAL_VAULT_TYPE,
DIGITAL_VAULT_TITLE, VAULT_USERNAME?, VAULT_FOLDER?, VAULT_FILE?,
VAULT_SECRET_NAME?, VAULT_SYSTEM_NAME?, VAULT_EP_NAME?,
VAULT_EP_TYPE?, VAULT_EP_CONT?, VAULT_NS_TYPE?, VAULT_NS_NAME?,
VAULT_ACCOUNT_NAME?, VAULT_AUTHORIZATION_NAME?,
VAULT_TARGET_NAME?, VAULT_SECRET_KV_PATH?, VAULT_SECRET_KV_NAME?,
VAULT_SECRET_KV_KEY?)>
<!ELEMENT DIGITAL_VAULT_ID (#PCDATA)>
<!ELEMENT DIGITAL_VAULT_TYPE (#PCDATA)>
<!ELEMENT DIGITAL_VAULT_TITLE (#PCDATA)>
<!ELEMENT VAULT_USERNAME (#PCDATA)>
<!ELEMENT VAULT_FOLDER (#PCDATA)>
<!ELEMENT VAULT_FILE (#PCDATA)>
<!ELEMENT VAULT_SECRET_NAME (#PCDATA)>
<!ELEMENT VAULT_SYSTEM_NAME (#PCDATA)>
<!ELEMENT VAULT_EP_NAME (#PCDATA)>
<!ELEMENT VAULT_EP_TYPE (#PCDATA)>
<!ELEMENT VAULT_EP_CONT (#PCDATA)>
<!ELEMENT VAULT_NS_TYPE (#PCDATA)>
<!ELEMENT VAULT_NS_NAME (#PCDATA)>
<!ELEMENT VAULT_ACCOUNT_NAME (#PCDATA)>
<!ELEMENT VAULT_AUTHORIZATION_NAME (#PCDATA)>
<!ELEMENT VAULT_TARGET_NAME (#PCDATA)>
<!ELEMENT VAULT_SECRET_KV_PATH (#PCDATA)>
<!ELEMENT VAULT_SECRET_KV_NAME (#PCDATA)>
<!ELEMENT VAULT_SECRET_KV_KEY (#PCDATA)>
...
<!-- EOF -->
```

# Option Profile API - DTD/XSD Change

| | |
|---|---|
| APIs affected | /api/2.0/fo/subscription/option_profile/ |
| New or Updated API | Neither (DTD/XSD change only) |
| DTD or XSD changes | Yes |

We added a new element ETHERNET_IP_PROBING to the Option Profile Info DTD and Option Profile XSD. This flag is for Qualys Internal Use Only.

### DTD update (option_profile_info.dtd)

Updated DTD: <baseURL>/api/2.0/fo/subscription/option_profile/option_profile_info.dtd

```
<!ELEMENT OPTION_PROFILES (OPTION_PROFILE)*>
...
<!ELEMENT SCAN (PORTS?, SCAN_DEAD_HOSTS?, CLOSE_VULNERABILITIES?,
PURGE_OLD_HOST_OS_CHANGED?, PERFORMANCE?,
LOAD_BALANCER_DETECTION?, PASSWORD_BRUTE_FORCING?,
VULNERABILITY_DETECTION?, AUTHENTICATION?,
ADDL_CERT_DETECTION?, DISSOLVABLE_AGENT?, LITE_OS_SCAN?,
ETHERNET_IP_PROBING?, CUSTOM_HTTP_HEADER?, HOST_ALIVE_TESTING?,
SCAN_RESTRICTION?, SYSTEM_AUTH_RECORD?,
FILE_INTEGRITY_MONITORING?, CONTROL_TYPES?, DO_NOT_OVERWRITE_OS?,
TEST_AUTHENTICATION?)>
...
<!ELEMENT LITE_OS_SCAN (#PCDATA)>
<!ELEMENT ETHERNET_IP_PROBING (#PCDATA)>
<!ELEMENT CUSTOM_HTTP_HEADER (VALUE?, DEFINITION_KEY?,
DEFINITION_VALUE?)>
<!ELEMENT VALUE (#PCDATA)>
<!ELEMENT DEFINITION_KEY (#PCDATA)>
<!ELEMENT DEFINITION_VALUE (#PCDATA)>
...
```

### XSD update (option_profiles.xsd)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema attributeFormDefault="unqualified"
elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="OPTION_PROFILES"
type="OPTION_PROFILESType"/>
    <xs:complexType name="CONTROL_TYPESType">
```

```
        <xs:sequence>
            <xs:element name="FIM_CONTROLS_ENABLED">
                <xs:simpleType>
                    <xs:restriction base="xs:integer">
                        <xs:enumeration value="1"/>
                        <xs:enumeration value="0"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="CUSTOM_WMI_QUERY_CHECKS">
                <xs:simpleType>
                    <xs:restriction base="xs:integer">
                        <xs:enumeration value="1"/>
                        <xs:enumeration value="0"/>
                    </xs:restriction>
                </xs:simpleType>
            ...
            <xs:element name="ETHERNET_IP_PROBING" minOccurs="0">
               <xs:simpleType>
                   <xs:restriction base="xs:integer">
                       <xs:enumeration value="1"/>
                       <xs:enumeration value="0"/>
                   </xs:restriction>
               </xs:simpleType>
            </xs:element>
           ...
</xs:schema>
```