Image signature verification

Cloud Volumes ONTAP

NetApp
September 22, 2023

Table of Contents

Image signature verification

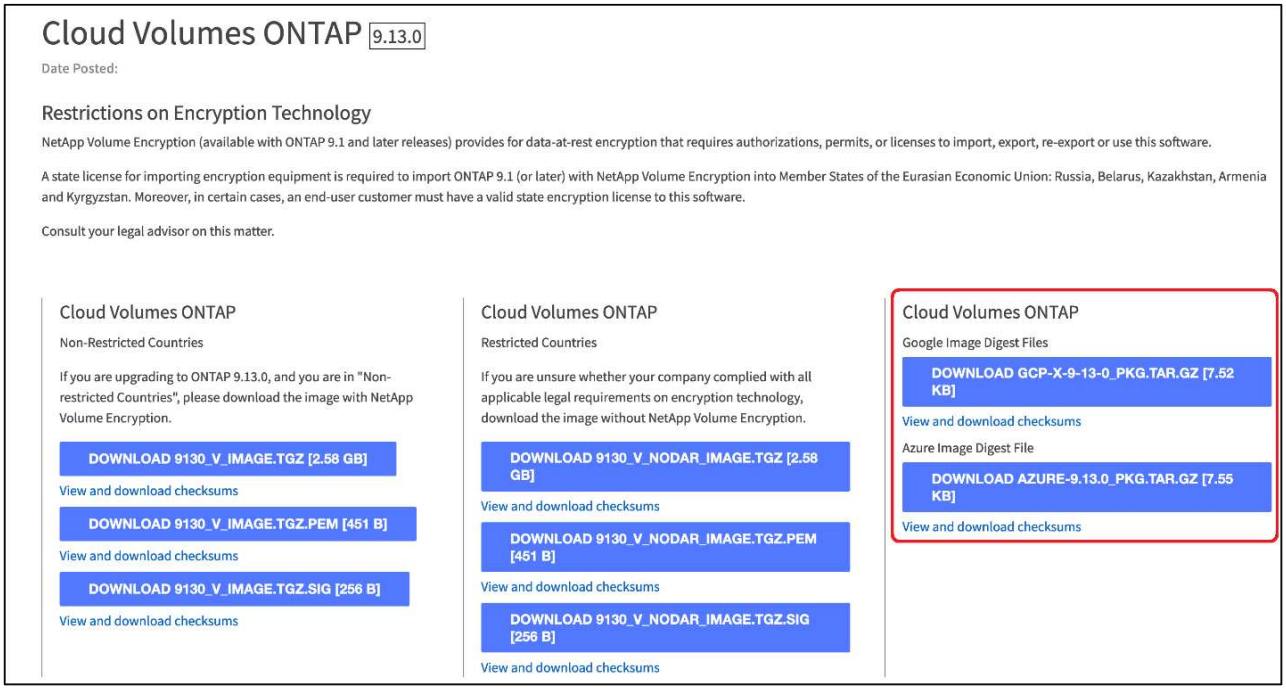
Verify Google Cloud signed images

To verify the exported Google Cloud signed image, you must download the image digest file from the NSS to validate the disk.raw file and digest file contents.

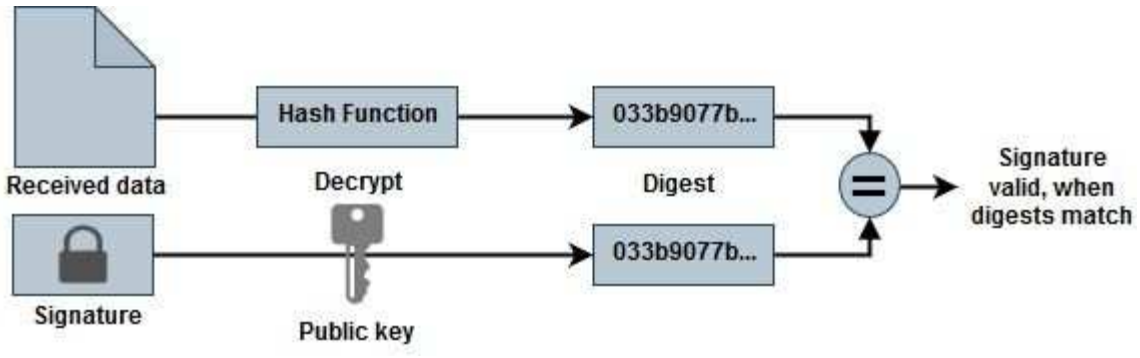
Signed image verification workflow summary

The following is an overview of the Google Cloud signed image verification workflow process.

- From the NSS, download the Google Cloud archive containing the following files:
  - Signed digest (.sig)
  - Certificate containing the public key (.pem)
  - Certificate chain (.pem)



- Download the converted disk.raw file
- Validate the certificate using the certificate chain
- Validate the signed digest using the certificate contain the public key
  - Decrypt the signed digest using the public key to extract the digest of the image file
  - Create a digest of the downloaded disk.raw file
  - Compare the two digest file for validation



Verification of disk.raw file and digest file contents using OpenSSL

You can verify the Google Cloud downloaded disk.raw file against the digest file contents available through the NSS using OpenSSL.

> The OpenSSL commands to validate the image are compatible with Linux, Mac OS, and Windows machines.

Steps

1. Verify the certificate using OpenSSL.

Click to display the script

```
# Step 1 - Optional, but recommended: Verify the certificate using
OpenSSL

# Step 1.1 - Copy the Certificate and certificate chain to a
directory
$ openssl version
LibreSSL 3.3.6
$ ls -l
total 48
-rw-r--r--@ 1 example-user  engr  8537 Jan 19 15:42 Certificate-
Chain-GCP-CVO-20230119-0XXXXX.pem
-rw-r--r--@ 1 example-user  engr  2365 Jan 19 15:42 Certificate-GCP-
CVO-20230119-0XXXXX.pem

# Step 1.2 - Get the OSCP URL
$ oscp_url=$(openssl x509 -noout -ocsp_uri -in <Certificate-
Chain.pem>)
$ oscp_url=$(openssl x509 -noout -ocsp_uri -in Certificate-Chain-
GCP-CVO-20230119-0XXXXX.pem)
$ echo $oscp_url
http://ocsp.entrust.net

# Step 1.3 - Generate an OCSP request for the certificate
$ openssl ocsp -issuer <Certificate-Chain.pem> -CAfile <Certificate-
Chain.pem> -cert <Certificate.pem> -reqout <request.der>
$ openssl ocsp -issuer Certificate-Chain-GCP-CVO-20230119-0XXXXX.pem
-CAfile Certificate-Chain-GCP-CVO-20230119-0XXXXX.pem -cert
Certificate-GCP-CVO-20230119-0XXXXX.pem -reqout req.der

# Step 1.4 - Optional: Check the new file "req.der" has been
generated
$ ls -l
total 56
-rw-r--r--@ 1 example-user  engr  8537 Jan 19 15:42 Certificate-
Chain-GCP-CVO-20230119-0XXXXX.pem
-rw-r--r--@ 1 example-user  engr  2365 Jan 19 15:42 Certificate-GCP-
CVO-20230119-0XXXXX.pem
-rw-r--r--  1 example-user  engr   120 Jan 19 16:50 req.der

# Step 1.5 - Connect to the OCSP Manager using openssl to send the
OCSP request
$ openssl ocsp -issuer <Certificate-Chain.pem> -CAfile <Certificate-
Chain.pem> -cert <Certificate.pem> -url ${ocsp_url} -resp_text
-respout <response.der>
```

```
$ openssl ocsp -issuer Certificate-Chain-GCP-CVO-20230119-0XXXXX.pem
-CAfile Certificate-Chain-GCP-CVO-20230119-0XXXXX.pem -cert
Certificate-GCP-CVO-20230119-0XXXXX.pem  -url ${ocsp_url} -resp_text
-respout resp.der
OCSP Response Data:
    OCSP Response Status: successful (0x0)
    Response Type: Basic OCSP Response
    Version: 1 (0x0)
    Responder Id: C = US, O = "Entrust, Inc.", CN = Entrust Extended
Validation Code Signing CA - EVCS2
    Produced At: Jan 19 15:14:00 2023 GMT
    Responses:
    Certificate ID:
      Hash Algorithm: sha1
      Issuer Name Hash: 69FA640329AB84E27220FE0927647B8194B91F2A
      Issuer Key Hash: CE894F8251AA15A28462CA312361D261FBF8FE78
      Serial Number: 5994B3D01D26D594BD1D0FA7098C6FF5
    Cert Status: good
    This Update: Jan 19 15:00:00 2023 GMT
    Next Update: Jan 26 14:59:59 2023 GMT

    Signature Algorithm: sha512WithRSAEncryption
         0b:b6:61:e4:03:5f:98:6f:10:1c:9a:f7:5f:6f:c7:e3:f4:72:
         f2:30:f4:86:88:9a:b9:ba:1e:d6:f6:47:af:dc:ea:e4:cd:31:
         af:e3:7a:20:35:9e:60:db:28:9c:7f:2e:17:7b:a5:11:40:4f:
         1e:72:f7:f8:ef:e3:23:43:1b:bb:28:1a:6f:c6:9c:c5:0c:14:
         d3:5d:bd:9b:6b:28:fb:94:5e:8a:ef:40:20:72:a4:41:df:55:
         cf:f3:db:1b:39:e0:30:63:c9:c7:1f:38:7e:7f:ec:f4:25:7b:
         1e:95:4c:70:6c:83:17:c3:db:b2:47:e1:38:53:ee:0a:55:c0:
         15:6a:82:20:b2:ea:59:eb:9c:ea:7e:97:aa:50:d7:bc:28:60:
         8c:d4:21:92:1c:13:19:b4:e0:66:cb:59:ed:2e:f8:dc:7b:49:
         e3:40:f2:b6:dc:d7:2d:2e:dd:21:82:07:bb:3a:55:99:f7:59:
         5d:4a:4d:ca:e7:8f:1c:d3:9a:3f:17:7b:7a:c4:57:b2:57:a8:
         b4:c0:a5:02:bd:59:9c:50:32:ff:16:b1:65:3a:9c:8c:70:3b:
         9e:be:bc:4f:f9:86:97:b1:62:3c:b2:a9:46:08:be:6b:1b:3c:
         24:14:59:28:c6:ae:e8:d5:64:b2:f8:cc:28:24:5c:b2:c8:d8:
         5a:af:9d:55:48:96:f6:3e:c6:bf:a6:0c:a4:c0:ab:d6:57:03:
         2b:72:43:b0:6a:9f:52:ef:43:bb:14:6a:ce:66:cc:6c:4e:66:
         17:20:a3:64:e0:c6:d1:82:0a:d7:41:8a:cc:17:fd:21:b5:c6:
         d2:3a:af:55:2e:2a:b8:c7:21:41:69:e1:44:ab:a1:dd:df:6d:
         15:99:90:cc:a0:74:1e:e5:2e:07:3f:50:e6:72:a6:b9:ae:fc:
         44:15:eb:81:3d:1a:f8:17:b6:0b:ff:05:76:9d:30:06:40:72:
         cf:d5:c4:6f:8b:c9:14:76:09:6b:3d:6a:70:2c:5a:c4:51:92:
         e5:cd:84:b6:f9:d9:d5:bc:8d:72:b7:7c:13:9c:41:89:a8:97:
         6f:4a:11:5f:8f:b6:c9:b5:df:00:7e:97:20:e7:29:2e:2b:12:
         77:dc:e2:63:48:87:42:49:1d:fc:d0:94:a8:8d:18:f9:07:85:
```

```
         e4:d0:3e:9a:4a:d7:d5:d0:02:51:c3:51:1c:73:12:96:2d:75:
         22:83:a6:70:5a:4a:2b:f2:98:d9:ae:1b:57:53:3d:3b:58:82:
         38:fc:fa:cb:57:43:3f:3e:7e:e0:6d:5b:d6:fc:67:7e:07:7e:
         fb:a3:76:43:26:8f:d1:42:d6:a6:33:4e:9e:e0:a0:51:b4:c4:
         bc:e3:10:0d:bf:23:6c:4b
WARNING: no nonce in response
Response Verify OK
Certificate-GCP-CVO-20230119-0XXXXX.pem: good
    This Update: Jan 19 15:00:00 2023 GMT
    Next Update: Jan 26 14:59:59 2023 GMT

# Step 1.5 - Optional: Check the response file "response.der" has
been generated. Verify its contents.
$ ls -l
total 64
-rw-r--r--@ 1 example-user  engr  8537 Jan 19 15:42 Certificate-
Chain-GCP-CVO-20230119-0XXXXX.pem
-rw-r--r--@ 1 example-user  engr  2365 Jan 19 15:42 Certificate-GCP-
CVO-20230119-0XXXXX.pem
-rw-r--r--  1 example-user  engr   120 Jan 19 16:50 req.der
-rw-r--r--  1 example-user  engr   806 Jan 19 16:51 resp.der

# Step 1.6 - Verify the chain of trust and expiration dates against
the local host
$ openssl version -d
OPENSSLDIR: "/private/etc/ssl"
$ OPENSSLDIR=$(openssl version -d | cut -d '"' -f2)
$ echo $OPENSSLDIR
/private/etc/ssl

$ openssl verify -untrusted <Certificate-Chain.pem> -CApath <OpenSSL
dir> <Certificate.pem>
$ openssl verify -untrusted Certificate-Chain-GCP-CVO-20230119-
0XXXXX.pem -CApath ${OPENSSLDIR} Certificate-GCP-CVO-20230119-
0XXXXX.pem
Certificate-GCP-CVO-20230119-0XXXXX.pem: OK
```

2. Place the downloaded disk.raw file, the signature, and certificates in a directory.
3. Extract the public key from the certificate using OpenSSL.
4. Decrypt the signature using the extracted public key and verify the contents of the downloaded disk.raw file.

Click to display the script

```
# Step 1 - Place the downloaded disk.raw, the signature and the
certificates in a directory
$ ls -l
-rw-r--r--@ 1 example-user  staff  Jan 19 15:42 Certificate-Chain-
GCP-CVO-20230119-0XXXXX.pem
-rw-r--r--@ 1 example-user  staff  Jan 19 15:42 Certificate-GCP-CVO-
20230119-0XXXXX.pem
-rw-r--r--@ 1 example-user  staff  Jan 19 15:42 GCP_CVO_20230119-
XXXXXX_digest.sig
-rw-r--r--@ 1 example-user  staff  Jan 19 16:39 disk.raw

# Step 2 - Extract the public key from the certificate
$ openssl x509 -pubkey -noout -in (certificate.pem) >
(public_key.pem)
$ openssl x509 -pubkey -noout -in Certificate-GCP-CVO-20230119-
0XXXXX.pem > CVO-GCP-pubkey.pem

$ ls -l
-rw-r--r--@ 1 example-user  staff  Jan 19 15:42 Certificate-Chain-
GCP-CVO-20230119-0XXXXX.pem
-rw-r--r--@ 1 example-user  staff  Jan 19 15:42 Certificate-GCP-CVO-
20230119-0XXXXX.pem
-rw-r--r--@ 1 example-user  staff  Jan 19 17:02 CVO-GCP-pubkey.pem
-rw-r--r--@ 1 example-user  staff  Jan 19 15:42 GCP_CVO_20230119-
XXXXXX_digest.sig
-rw-r--r--@ 1 example-user  staff  Jan 19 16:39 disk.raw

# Step 3 - Decrypt the signature using the extracted public key and
verify the contents of the downloaded disk.raw
$ openssl dgst -verify (public_key) -keyform PEM -sha256 -signature
(signed digest) -binary (downloaded or obtained disk.raw)
$ openssl dgst -verify CVO-GCP-pubkey.pem -keyform PEM -sha256
-signature GCP_CVO_20230119-XXXXXX_digest.sig -binary disk.raw
Verified OK

# A failed response would look like this
$ openssl dgst -verify CVO-GCP-pubkey.pem -keyform PEM -sha256
-signature GCP_CVO_20230119-XXXXXX_digest.sig -binary
../sample_file.txt
Verification Failure
```

Copyright information

Trademark information