



Manage security certificates

ONTAP 9.11.1 REST API reference

NetApp
April 02, 2024

Table of Contents

- Manage security certificates 1
 - Security certificates endpoint overview 1
 - Retrieve security certificates 11
 - Create or install security certificates 20
 - Sign security certificates 32
 - Delete security certificates 36
 - Retrieve security certificates 38

Manage security certificates

Security certificates endpoint overview

Overview

This API displays security certificate information and manages the certificates in ONTAP.

Installing certificates in ONTAP

The security certificates GET request retrieves all of the certificates in the cluster.

Examples

Retrieving all certificates installed in the cluster with their common-names

```
# The API:
/api/security/certificates

# The call:
curl -X GET "https://<mgmt-
ip>/api/security/certificates?fields=common_name" -H "accept:
application/hal+json"

# The response:
{
  "records": [
    {
      "svm": {
        "name": "vs0"
      },
      "uuid": "dad2363b-8ac0-11e8-9058-005056b482fc",
      "common_name": "vs0",
      "_links": {
        "self": {
          "href": "/api/security/certificates/dad2363b-8ac0-11e8-9058-
005056b482fc"
        }
      }
    },
    {
      "uuid": "1941e048-8ac1-11e8-9058-005056b482fc",
      "common_name": "ROOT",
      "_links": {
        "self": {
          "href": "/api/security/certificates/1941e048-8ac1-11e8-9058-
```

```
005056b482fc"
  }
}
},
{
  "uuid": "5a3a77a8-892d-11e8-b7da-005056b482fc",
  "common_name": "gshancluster-4",
  "_links": {
    "self": {
      "href": "/api/security/certificates/5a3a77a8-892d-11e8-b7da-
005056b482fc"
    }
  }
}
],
"num_records": 3,
"_links": {
  "self": {
    "href": "/api/security/certificates?fields=common_name"
  }
}
}
```

Retrieving all certificates installed at cluster-scope with their common-names

```
# The API:
/api/security/certificates

# The call:
curl -X GET "https://<mgmt-
ip>/api/security/certificates?scope=cluster&fields=common_name" -H
"accept: application/hal+json"

# The response:
{
  "records": [
    {
      "uuid": "1941e048-8ac1-11e8-9058-005056b482fc",
      "scope": "cluster",
      "common_name": "ROOT",
      "_links": {
        "self": {
          "href": "/api/security/certificates/1941e048-8ac1-11e8-9058-
005056b482fc"
        }
      }
    },
    {
      "uuid": "5a3a77a8-892d-11e8-b7da-005056b482fc",
      "scope": "cluster",
      "common_name": "gshancluster-4",
      "_links": {
        "self": {
          "href": "/api/security/certificates/5a3a77a8-892d-11e8-b7da-
005056b482fc"
        }
      }
    }
  ],
  "num_records": 2,
  "_links": {
    "self": {
      "href": "/api/security/certificates?scope=cluster&fields=common_name"
    }
  }
}
```

Retrieving all certificates installed on a specific SVM with their common-names

```
# The API:
/api/security/certificates

# The call:
curl -X GET "https://<mgmt-
ip>/api/security/certificates?svm.name=vs0&fields=common_name" -H "accept:
application/hal+json"

# The response:
{
  "records": [
    {
      "svm": {
        "name": "vs0"
      },
      "uuid": "dad2363b-8ac0-11e8-9058-005056b482fc",
      "common_name": "vs0",
      "_links": {
        "self": {
          "href": "/api/security/certificates/dad2363b-8ac0-11e8-9058-
005056b482fc"
        }
      }
    },
    ],
  "num_records": 1,
  "_links": {
    "self": {
      "href": "/api/security/certificates?svm.name=vs0&fields=common_name"
    }
  }
}
```

Retrieving a certificate using its UUID for all fields

```
# The API:
/api/security/certificates/{uuid}
```

```

# The call:
curl -X GET "https://<mgmt-ip>/api/security/certificates/dad2363b-8ac0-11e8-9058-005056b482fc?fields=*" -H "accept: application/hal+json"

# The response:
{
  "svm": {
    "uuid": "d817293c-8ac0-11e8-9058-005056b482fc",
    "name": "vs0"
  },
  "uuid": "dad2363b-8ac0-11e8-9058-005056b482fc",
  "scope": "svm",
  "type": "server",
  "common_name": "vs0",
  "serial_number": "15428D45CF81CF56",
  "ca": "vs0",
  "hash_function": "sha256",
  "key_size": 2048,
  "expiry_time": "2019-07-18T15:29:14-04:00",
  "public_certificate": "-----BEGIN CERTIFICATE-----
\nMIIDQjCCAIqgAwIBAgIIFUKNRC+Bz1YwDQYJKoZIhvcNAQELBQAwGzEMMAoGA1UE\nAxMDdn
MwMQswCQYDVQQGEwJVUzAeFw0xODA3MTgxOTI1MTRaFw0xOTA3MTgxOTI1\nMTRaMBsxDDAKBg
NVBAMTA3ZmDELMakGA1UEBhMCMVVMwggEiMA0GCSqGSIb3DQEB\nAQUA4IBDwAwggEKAoIBAQCqFQb27th2ACOMJvWgLh1xRzobSb2ZTQfO561faXQ3\n\nIbit+rnRWXetd/s2+iCv91d9LW0NOM
P3MN2f3SFbyze3dl7WrnVbjLmYuI9MfOxs\nfmA+Bh6gpap5Yn2YddqoV6rfNGAuUveNLArN18
wODk/mpawpEQ93QSa1Zfg1gnoH\nrFrYqiSYT06X5g6RbUuEl4LTGXspz+plU46Za0i6QyxtvZ
4bneibffXN3IigpqI6\nnTGUV8R/J3Ps338VxVmSO9ZXBZmvbcJVoySYNIC1/oi3fgPZlnBv0tb
swqg4FoZO/\nWT+XHGHlep6cr/Aqg7u6C4RfqbCwzB/XFKDIqnmAQkDBAgMBAAGjgYkkgYYwDA
YD\nVR0TBAUwAwEB/zALBgNVHQ8EBAMCAQYwHQYDVR0OBBYEFN/AnH8qLxocTtumNHIn\nnEN4I
FIDBMEoGA1UdIwRDMEGAFN/AnH8qLxocTtumNHInEN4IFIDBoR+kHTAbMQww\nnCgYDVQQDEwN2
czAxCzAJBgNVBAYTA1VTgggVQo1Fz4HPVjANBgkqhkiG9w0BAQsF\nnAAOCAQEAA0pUEepdeQnd
2Amwg8UFyxayb8eu3E6dlptvtyp+xtjhIC7Dh95CVXhy\nnkJS3Tsu60PGR/b2vc3MZtAUpcL4c
eD8XntKPQgBlqoB4bRogCe1TnlGswRXDX5TS\nngMvrRjaWTBF7ikt4UjR05rSxcDGplQRqjnOt
hqi+yPT+29+8a4Uu6J+3Kdrflj4p\nn1nSWpuB9EyxtuCILNqXA2ncH7YKtoeNtChKCchhvPcoT
y6Opma6UQn5UMxstkvGT\nnVGaN5TlRWv0yiqPXIQblSqXi/uQsuRPcHDu7+KWRFn08USa6QVo2
mDs9P7R9dd0K\nn9QAsTjTOF9PlAKgNxGoOJl2y0+48AA==\n-----END CERTIFICATE-----
\n",
  "_links": {
    "self": {
      "href": "/api/security/certificates/dad2363b-8ac0-11e8-9058-005056b482fc"
    }
  }
}

```

Creating a certificate in a cluster

These certificates can be used to help administrators enable certificate-based authentication and to enable SSL-based communication to the cluster.

```
# The API:
/api/security/certificates

# The call:
curl -X POST "https://<mgmt-ip>/api/security/certificates" -H "accept:
application/hal+json" -H "Content-Type: application/json" -d "{
  \"common_name\": \"TEST-SERVER\", \"type\": \"server\" }"
```

Installing a certificate in a cluster

These certificates can be used to help administrators enable certificate-based authentication and to enable-SSL based communication to the cluster.


```

# The API:
/api/security/certificates

# The call:
curl -X POST "https://<mgmt-ip>/api/security/certificates" -H "accept:
application/json" -H "Content-Type: application/json" -d "{ \"type\":
\"server_ca\", \"public_certificate\": \"-----BEGIN CERTIFICATE-----
\nMIIFYDCCA0igAwIBAgIQCgFCgAAAAUjyES1AAAAAjANBgkqhkiG9w0BAQsFADBKMQswCQYD
VQQG\nEwJVUzESMBAGA1UEChMJSWRlbnRydXN0MScwJQYDVQQDEx5JZGVuVHJ1c3QgQ29tbWVy
Y2lhbCBS\nb290IENBIDEwHhcNMTQwMTE2MTg5MjIzWhcNMTQwMTE2MTg5MjIzWjBKMQswCQYD
VQQGEwJVUzES\nMBAGA1UEChMJSWRlbnRydXN0MScwJQYDVQQDEx5JZGVuVHJ1c3QgQ29tbWVy
Y2lhbCBSb290IENB\nnIDEwggIiMA0GCSqGSIb3DQEBAQUAA4ICDwAwggIKAoICAQCnUBneP5k9
1DNG8W9RYYKyqU+PZ4ld\nhNlT3Qwo2dfw/66VQ3KZ+bVdfIrBQuExUHRgQl8zZshq0PirK1e
hm7zCYofWjK9ouuU+ehcCuz/\nmNKvcb00U59Oh++SvL3sTzIwiEsXXlFEU8L2ApeN2WIrvyQf
Yo3fw7gpS0l4PJNgiCL8mdo2yMKi\nlCxUAGclbnO/AljwpN3lsKImesrgNqUZFvX9t++uP0D1
bVoE/c40yiTcdCMbXTMTEl3EASX2MN0C\nXZ/g1Ue9tOsobotJSdifWwLziuQkkORiT0/Br4s0
dBeo0XKIanoBScy0RnnGF7HamB4HWfp1IYVl\n3ZBWzvurpWCdxJ35UrClvYf5jysjCiN2O/cz
4ckA82n5S6LgTrx+kzmEB/dEcH7+B1rlsazRGMzy\nNeVJSQjKVsk9+w8YfYs7wRPCTY/JTw43
6R+hDmrfYi7LNQZReSzIJTj0+kuniVyc0uMNOYZkDhZV\nWYfCP04MXFL0PfdSgvHqo6z9STQa
KPNBiDoT7uje/5kdX7rL6B7yuVBgdHTc+XvvqDtMwt0viAg\nxGds8AgDelWAF0Z0lqf0Hj7h
9tgJ4TNkK2PXm16f+cb7D3hvl7yTmvmcEpB4eoCHFddyJxVdHix\nnuuFucAS6T6C6aMN7/zHw
cz09lCqxCOEOoP5NiGVreTO01wIDAQABo0IwQDAOBgNVHQ8BAf8EBAMC\nAQYwDwYDVR0TAQH/
BAUwAwEB/zAdBgNVHQ4EFgQU7UQZwNPwBovupHu+QucmVMiONnYwDQYJKoZI\nnhvcNAQELBQAD
ggIBAA2ukDL2pkt8RHYZYR4nKM1eVO8lvOMIkPkp165oCOGUAFjvLi5+U1KMtlwH\n6oi6mYtQ
lNeCgN9hCQCTrQ0U5s7B8jeUeLBfnLOic7iPBZM4zy0+sLj7wM+x8uwtLRvM7Kqas6pg\nnghst
O8OEPVeKlh6cdbhTMM1gC1OQ045U8U1mwF10A0Cj7oV+wh93nAbowacYXVKV7cndJZ5t+qnt\n
ozo00F172u1Q8zW/7esUTTHHYPTa8Yec4kjixsU3+wYQ+nVZZjFHKdp2mhZpgq7vmr1R94gJmm
mV\nYjz1VYA211QC//G5Xc7UI2/YRYRKW2XviQzdFKcgyxilJbQN+QHwotL0AMh0jqEqSI5l2x
PE4iUX\nnfeu+h1sXIFRRk0pTAvsXcoz7WL9RccvW9xYoIA55vrX/hMUpu091EpCdNTDd1lzzY
9GvlU47/ro\nnkTLq11gEIt44w8y8bckzOmoKaT+gyOpyj4xjhi09bTyWnpXgSUyqorkqG5w2gX
jtw+hG4izZRHUe\n2XWJUc0QhJ1hYmtD+ZciTY6Y5uN/9lu7rs3KSoFrXgvzUeF0K+l+J6fZmU
lO+KWA2yUPHGNiiskz\nZ2s8EIPGrd6ozRaOjfAHN3Gf8qv8QfXBi+wAN10J5U6A7/qxXDgGpR
tK4dw4LTzcx+QGtVKno7R\nncGzM7vRX+Bi6hG6H\n-----END CERTIFICATE-----\n\"
}"

```

Installing a certificate on a specific SVM


```
# The API:  
/api/security/certificates/{uuid}  
  
# The call:  
curl -X DELETE "https://<mgmt-ip>/api/security/certificates/dad2363b-8ac0-  
11e8-9058-005056b482fc?fields=*" -H "accept: application/hal+json"
```

Signing a new certificate signing request using an existing CA certificate UUID

Once you have created a certificate of type "root_ca", you can use that certificate to act as a local Certificate Authority to sign new certificate signing requests. The following example signs a new certificate signing request using an existing CA certificate UUID. If successful, the API returns a signed certificate.

```

# The API:
/api/security/certificates/{ca.uuid}/sign

# The call:
curl -X POST "https://<mgmt-ip>/api/security/certificates/253add53-8ac9-11e8-9058-005056b482fc/sign" -H "accept: application/json" -H "Content-Type: application/json" -d "{ \"signing_request\": \"-----BEGIN CERTIFICATE REQUEST-----
\nMIICYTCCAUAkCAQAwhDENMAsGA1UEAxMEVEVTVDELMAkGA1UEBhMCVVMwggEiMA0G\nnCSqGSI
b3DQEBAQUAA4IBDwAwggEKAoIBAQCIBCuVfbYHNdOO7vjRQja4JqL2cHqK\ndr1Tj5hz9RVqFK
Z7VPPh8DSP9LoTbYWsvrTkbuD0Wi715MVQCsbkq/mHos+Y51fqs\nNP5K92fc6EhBzBDYFgZGFn
tZYJjEG5MPerIUE7CfVy7o6sjWolxeY33pjefObyvP\nBcJkBHg6SFJK/TDLvIYJkonLkJEOJo
TI6++a3I/1bCMfUeuRtLU9ThWlna1kMMYK\n4T16/Bxgm4bha2U2jtosc0Wltnld/capc+eqRV
07WVbMmEOTtop3cv0h3N0S61bn\nFkd96DXzeGWbSHFHckeCZ9bOHhnVbfEa/efkPLx7ziMC8G
tRHHlwbNk7AgMBAAGg\nADANBgkqhkiG9w0BAQsFAAOCAQEaf+rs1i5PHaOSI2HtTM+Hcv/p71
yzgoLL+aeU\ntB0V4iuoXdqY8oQeWoPI92ci0K08JuSpu6D0DwCK1stfwuGkAA2b0Wr7ZDRonT
Uq\nmJ4j3O47MLysW4Db2LbGws/AuDsCIrBJDWHMPhaqsvRbpMx2xQ/V5oagUw5eGGpN\ne4fg
/E2k9mGkpxwUzT7w1RZirpND4xL+XTzpzzeZqgalpXug4yjIXlI5hpRESZ9/\nAkGJSCWxi15I
ZdxxFVXlBcmm6WpJnnboqkcKeXz95GM6Re+oBy9tlgvwv1Vd5s8uHX+bycFiZp09Wsm8Ev727M
ziZ+0II9nxwkDKsdPvam+KLI9hLQ==\n-----END CERTIFICATE REQUEST-----\n\",
\"hash_function\": \"sha256\"}"

# The response:
{
  "public_certificate": "-----BEGIN CERTIFICATE-----
\nMIIDBzCCAe+gAwIBAgIIFUKQpcqeaUAWDQYJKoZIhvcNAQELBQAwhDENMAsGA1UE\nnAxMEUk
FDWDELMAkGA1UEBhMCVVMwHhcNMTgwNze4MjAzMTA1WhcNMTkwNze4MjAz\nnMTA1WjAcMQ0wCw
YDVQQDEWRURVNUMQswCQYDVQQGEWJVUzCCASIwDQYJKoZIhvcN\nnAQEBBQADggEPADCCAQoCgg
EBAKIEK5V9tgc1047u+NFCNrgmovZweop2uVOPmHP1\nnFWoUpntU+HwNI/0uhNthay+tORu4PR
aLvXkxVAKXuSr+Yeiz5jmV+qw0/kr3Z9zo\nnSEHMENgWBkYWellgmMQbkw96shQTsJ9XLujqyN
Y6XF5jfemN585vK88FwmQEeDpI\nnUkr9MMu8hgmSicuQkQ4mhMjr75rcj/VsIx9R65G0tT10Fa
WdrWQwxgrhPXR8HGCb\nnhuFrZTa02ixzRaW2eV39xqlz56pFXtTzVsyYQ502indy/SHc3RLqVu
cWR33oNfn4\nnZZtIcUdyR4Jn1s4eGdVt8Rr95+Q8vHvOIwLwa1EceXBucrsCAwEAAaNNMEswCQ
YD\nnVR0TBAlwADAdBgNVHQ4EFgQUJMPxjeW1G76TbbD2tXB8dwSpI3MwHwYDVR0jBBgw\nnFoAU
u5aH0mWR4cFoN9i7k96d2op3sPwwDQYJKoZIhvcNAQELBQADggEBAl5ai+Zi\nnFQZUXRTqJCgH
sgBThARneVWQYkYpyAXmTR7QeLfld4ZHL33i4xWCqX3uvW7SFJLe\nnZajT2AVmgiDbaWIHtDtv
qz1BY78PSgUwPH/IyARTEOBeikp6KdwMPraehDIBMAcc\nnANY58wXiTBbsl8UMD6tGecgnzw6s
xlMmadGvrfJeJmgY4zert6NNvgtPPhcZQdLS\nnE0fGzHS6+3ajCCfEEhPNPer9D0e5Me8li9Es
QGENrnJzTci8rzXPuF4bC3gghrK1\nnI1+kmJQ1kLYVUcsntcrIiHmNvtPFJY6stjDgQKS9aDd/
THhPpokPtZoCmE6PDxh6\nnR+dO6C0hcDKHFzA=\n-----END CERTIFICATE-----\n"
}

```

Generate a new Certificate Signing Request (CSR)

```
# The API:
/api/security/certificate-signing-request

# The call:
curl -X POST "https://<mgmt-ip>/api/security/certificate-signing-request"
-H "accept: application/json" -H "Content-Type: application/json" -d "{
  \"algorithm\": \"rsa\", \"extended_key_usage\": [\"serverauth\"],
  \"hash_function\": \"sha256\", \"key_usage\": [\"digitalsignature\"],
  \"security_strength\": \"112\", \"subject_alternatives\": { \"dns\": [
  \"*.example.com\", \"*.example1.com\" ], \"email\": [\"abc@example.com\",
  \"abc@example1.com\"], \"ip\": [\"10.225.34.223\", \"10.225.34.224\"],
  \"uri\": [\"http://example.com\", \"http://example1.com\"] },
  \"subject_name\": \"C=US,O=NTAP,CN=test.domain.com\"}"
{
  \"csr\": \"-----BEGIN CERTIFICATE REQUEST-----\n-----END CERTIFICATE
REQUEST-----\n\",
  \"generated_private_key\": \"-----BEGIN PRIVATE KEY-----\n-----END PRIVATE
KEY-----\n\"
}
```

Retrieve security certificates

GET /security/certificates

Introduced In: 9.6

Retrieves security certificates.

Related ONTAP commands

- security certificate show

Parameters

| Name | Type | In | Required | Description |
|-------------|--------|-------|----------|--|
| private_key | string | query | False | Filter by private_key <ul style="list-style-type: none"> • Introduced in: 9.8 |
| uuid | string | query | False | Filter by uuid <ul style="list-style-type: none"> • Introduced in: 9.8 |

| Name | Type | In | Required | Description |
|---------------------------|---------|-------|----------|--|
| type | string | query | False | Filter by type |
| subject_key_identifier | string | query | False | Filter by subject_key_identifier • Introduced in: 9.8 |
| public_certificate | string | query | False | Filter by public_certificate • Introduced in: 9.8 |
| ca | string | query | False | Filter by ca • maxLength: 256 • minLength: 1 |
| common_name | string | query | False | Filter by common_name |
| authority_key_identifier | string | query | False | Filter by authority_key_identifier • Introduced in: 9.8 |
| serial_number | string | query | False | Filter by serial_number • maxLength: 40 • minLength: 1 |
| key_size | integer | query | False | Filter by key_size |
| intermediate_certificates | string | query | False | Filter by intermediate_certificates • Introduced in: 9.8 |
| hash_function | string | query | False | Filter by hash_function |

| Name | Type | In | Required | Description |
|----------------|---------------|-------|----------|---|
| name | string | query | False | Filter by name <ul style="list-style-type: none"> • Introduced in: 9.8 |
| expiry_time | string | query | False | Filter by expiry_time |
| scope | string | query | False | Filter by scope |
| svm.uuid | string | query | False | Filter by svm.uuid |
| svm.name | string | query | False | Filter by svm.name |
| fields | array[string] | query | False | Specify the fields to return. |
| max_records | integer | query | False | Limit the number of records returned. |
| return_timeout | integer | query | False | The number of seconds to allow the call to execute before returning. When iterating over a collection, the default is 15 seconds. ONTAP returns earlier if either max records or the end of the collection is reached. <ul style="list-style-type: none"> • Default value: 1 • Max value: 120 • Min value: 0 |
| return_records | boolean | query | False | The default is true for GET calls. When set to false, only the number of records is returned. <ul style="list-style-type: none"> • Default value: 1 |

| Name | Type | In | Required | Description |
|----------|---------------|-------|----------|---|
| order_by | array[string] | query | False | Order results by specified fields and optional [asc |

Response

Status: 200, Ok

| Name | Type | Description |
|-------------|---|-------------------|
| _links | _links | |
| num_records | integer | Number of records |
| records | array[security_certificate] | |

Example response

```
{
  "_links": {
    "next": {
      "href": "/api/resourcelink"
    },
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "records": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "authority_key_identifier":
"26:1F:C5:53:5B:D7:9E:E2:37:74:F4:F4:06:09:03:3D:EB:41:75:D7",
    "ca": "string",
    "common_name": "test.domain.com",
    "hash_function": "sha1",
    "intermediate_certificates": {
    },
    "name": "cert1",
    "private_key": "-----BEGIN PRIVATE KEY-----
MIIBVAIBADANBgkqhkiG9w0BAQEFAASCAT4wggE6AgEAAkEAu1/a8f3G47cZ6pel
Hd3aONMNkGJ8vSCH5QjicuDm92VtVwkaACEjIoZSLYlJvPD+odL+lFzVQSmkneW7
VCgqYQIDAQABAkAcfNpg6GCQxoneLOghvlUrRotNZGvqpUOEAvHK3X7AJhz5SU4V
an36qvsAt5ghFMVM2iGvGaXbj0dAd+Jg64pxAiEA32Eh9mPtFSmZhTIUMeGcPmPk
qIYCEuP8a/ZLmI9s4TsciQDWvLQuvjSVfwPhi0TFAb5wqAET8X5LBFqtGX5QlUep
EwIgfngM02Gc4wtLoqa2d4qPkYu13+uUW9hLd4Xsd6i/OS8CIQDT3elU+Rt+qIwW
u0cFrVvNYSV3HNzDfS9N/IoxTagfewIgpVxADe5c2EWbhCUkhN+ZCf38AKewK9TW
lQcDy4L+f14= -----END PRIVATE KEY-----",
    "public_certificate": "-----BEGIN CERTIFICATE-----
MIIBuzCCAWWgAwIBAgIIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQAwhDENMAsGA1UE
AxMEVEVTVDELMAkGA1UEBhMCVVMwHhcNMtgnjA4MTgwOTAxWhcNMtgnjA4MTgw
OTAxWjAcMQ0wCwYDVQQDEwRURVNUMQswCQYDVQQGEwJVUzBcMA0GCSqGSIb3DQEB
AQUAA0sAMEgCQQDaPvbqUJJFJ6NNTyK3Yb+ytSjJ9aa3yUmYTD9uMiP+6ycjxHWB
e8u9z6yCHsW03ync+dnhE5c5z8wuDAY0fv15AgMBAAGjgYowgYcwDAYDVR0TBAUw
AwEB/zALBgNVHQ8EBAMCAQYwHQYDVR0OBBYEFMJ7Ev/o/3+YNzYh5XNlqqjnw4zm
MEsGA1UdIwREMEKAFMJ7Ev/o/3+YNzYh5XNlqqjnw4zmoSCKHjAcMQ0wCwYDVQQD
EwRURVNUMQswCQYDVQQGEwJVU4IIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQADQQA
vDovYeyGNknjGI+TVNX6nDbyzf7zUPqnri0KuvObEeybrbPW45sgsnT5dyeE/32U
9Yr6lklknbTtVBDTmLnrc -----END CERTIFICATE-----",
    "scope": "svm",
  }
}
```

```

    "serial_number": "string",
    "subject_key_identifier":
    "26:1F:C5:53:5B:D7:9E:E2:37:74:F4:F4:06:09:03:3D:EB:41:75:D8",
    "svm": {
      "_links": {
        "self": {
          "href": "/api/resourcelink"
        }
      },
      "name": "svm1",
      "uuid": "02c9e252-41be-11e9-81d5-00a0986138f7"
    },
    "type": "client",
    "uuid": "string"
  }
}

```

Error

Status: Default, Error

| Name | Type | Description |
|-------|-------|-------------|
| error | error | |

Example error

```

{
  "error": {
    "arguments": {
      "code": "string",
      "message": "string"
    },
    "code": "4",
    "message": "entry doesn't exist",
    "target": "uuid"
  }
}

```

Definitions

See Definitions

href

| Name | Type | Description |
|------|--------|-------------|
| href | string | |

_links

| Name | Type | Description |
|------|----------------------|-------------|
| next | href | |
| self | href | |

_links

| Name | Type | Description |
|------|----------------------|-------------|
| self | href | |

svm

| Name | Type | Description |
|--------|------------------------|-----------------------------------|
| _links | _links | |
| name | string | The name of the SVM. |
| uuid | string | The unique identifier of the SVM. |

security_certificate

| Name | Type | Description |
|--------------------------|------------------------|--|
| _links | _links | |
| authority_key_identifier | string | Provides the key identifier of the issuing CA certificate that signed the SSL certificate. |
| ca | string | Certificate authority |
| common_name | string | FQDN or custom common name. Provide on POST when creating a self-signed certificate. |

| Name | Type | Description |
|---------------------------|---------------|---|
| expiry_time | string | Certificate expiration time. Can be provided on POST if creating self-signed certificate. The expiration time range is between 1 day to 10 years. |
| hash_function | string | Hashing function. Can be provided on POST when creating a self-signed certificate. Hash functions md5 and sha1 are not allowed on POST. |
| intermediate_certificates | array[string] | Chain of intermediate Certificates in PEM format. Only valid in POST when installing a certificate. |
| key_size | integer | Key size of requested Certificate in bits. One of 512, 1024, 1536, 2048, 3072. Can be provided on POST if creating self-signed certificate. Key size of 512 is not allowed on POST. |
| name | string | Certificate name. If not provided in POST, a unique name specific to the SVM is automatically generated. |
| private_key | string | Private key Certificate in PEM format. Only valid for create when installing a CA-signed certificate. This is not audited. |
| public_certificate | string | Public key Certificate in PEM format. If this is not provided in POST, a self-signed certificate is created. |
| scope | string | Set to "svm" for interfaces owned by an SVM. Otherwise, set to "cluster". |
| serial_number | string | Serial number of certificate. |
| subject_key_identifier | string | Provides the key identifier used to identify the public key in the SSL certificate. |

| Name | Type | Description |
|------|---------------------|--|
| svm | svm | |
| type | string | <p>Type of Certificate. The following types are supported:</p> <ul style="list-style-type: none"> • client - a certificate and its private key used by an SSL client in ONTAP. • server - a certificate and its private key used by an SSL server in ONTAP. • client_ca - a Certificate Authority certificate used by an SSL server in ONTAP to verify an SSL client certificate. • server_ca - a Certificate Authority certificate used by an SSL client in ONTAP to verify an SSL server certificate. • root_ca - a self-signed certificate used by ONTAP to sign other certificates by acting as a Certificate Authority. • enum: ["client", "server", "client_ca", "server_ca", "root_ca"] • Introduced in: 9.6 |
| uuid | string | Unique ID that identifies a certificate. |

error_arguments

| Name | Type | Description |
|---------|--------|------------------|
| code | string | Argument code |
| message | string | Message argument |

error

| Name | Type | Description |
|-----------|--|-------------------|
| arguments | array[error_arguments] | Message arguments |

| Name | Type | Description |
|---------|--------|---|
| code | string | Error code |
| message | string | Error message |
| target | string | The target parameter that caused the error. |

Create or install security certificates

POST /security/certificates

Introduced In: 9.6

Creates or installs a certificate.

Required properties

- `svm.uuid` or `svm.name` - Existing SVM in which to create or install the certificate.
- `common_name` - Common name of the certificate. Required when creating a certificate.
- `type` - Type of certificate.
- `public_certificate` - Public key certificate in PEM format. Required when installing a certificate.
- `private_key` - Private key certificate in PEM format. Required when installing a CA-signed certificate.

Recommended optional properties

- `expiry_time` - Certificate expiration time. Specifying an expiration time is recommended when creating a certificate.
- `key_size` - Key size of the certificate in bits. Specifying a strong key size is recommended when creating a certificate.
- `name` - Unique certificate name per SVM. If one is not provided, it is automatically generated.

Default property values

If not specified in POST, the following default property values are assigned:

- `key_size` - *2048*
- `expiry_time` - *P365DT*
- `hash_function` - *sha256*

Related ONTAP commands

- `security certificate create`

- security certificate install

Parameters

| Name | Type | In | Required | Description |
|----------------|---------|-------|----------|---|
| return_records | boolean | query | False | <p>The default is false. If set to true, the records are returned.</p> <ul style="list-style-type: none"> • Default value: |

Request Body

| Name | Type | Description |
|---------------------------|------------------------|---|
| _links | _links | |
| authority_key_identifier | string | Provides the key identifier of the issuing CA certificate that signed the SSL certificate. |
| ca | string | Certificate authority |
| common_name | string | FQDN or custom common name. Provide on POST when creating a self-signed certificate. |
| expiry_time | string | Certificate expiration time. Can be provided on POST if creating self-signed certificate. The expiration time range is between 1 day to 10 years. |
| hash_function | string | Hashing function. Can be provided on POST when creating a self-signed certificate. Hash functions md5 and sha1 are not allowed on POST. |
| intermediate_certificates | array[string] | Chain of intermediate Certificates in PEM format. Only valid in POST when installing a certificate. |
| key_size | integer | Key size of requested Certificate in bits. One of 512, 1024, 1536, 2048, 3072. Can be provided on POST if creating self-signed certificate. Key size of 512 is not allowed on POST. |

| Name | Type | Description |
|------------------------|--------|--|
| name | string | Certificate name. If not provided in POST, a unique name specific to the SVM is automatically generated. |
| private_key | string | Private key Certificate in PEM format. Only valid for create when installing a CA-signed certificate. This is not audited. |
| public_certificate | string | Public key Certificate in PEM format. If this is not provided in POST, a self-signed certificate is created. |
| scope | string | Set to "svm" for interfaces owned by an SVM. Otherwise, set to "cluster". |
| serial_number | string | Serial number of certificate. |
| subject_key_identifier | string | Provides the key identifier used to identify the public key in the SSL certificate. |
| svm | svm | |

| Name | Type | Description |
|------|--------|--|
| type | string | <p>Type of Certificate. The following types are supported:</p> <ul style="list-style-type: none"> • client - a certificate and its private key used by an SSL client in ONTAP. • server - a certificate and its private key used by an SSL server in ONTAP. • client_ca - a Certificate Authority certificate used by an SSL server in ONTAP to verify an SSL client certificate. • server_ca - a Certificate Authority certificate used by an SSL client in ONTAP to verify an SSL server certificate. • root_ca - a self-signed certificate used by ONTAP to sign other certificates by acting as a Certificate Authority. • enum: ["client", "server", "client_ca", "server_ca", "root_ca"] • Introduced in: 9.6 |
| uuid | string | Unique ID that identifies a certificate. |

Example request

```
{
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "authority_key_identifier":
  "26:1F:C5:53:5B:D7:9E:E2:37:74:F4:F4:06:09:03:3D:EB:41:75:D7",
  "ca": "string",
  "common_name": "test.domain.com",
  "hash_function": "sha1",
  "intermediate_certificates": {
  },
  "name": "cert1",
  "private_key": "-----BEGIN PRIVATE KEY-----
MIIBVAIBADANBqkqhkiG9w0BAQEFAASCAT4wggE6AgEAAkEAu1/a8f3G47cZ6pel
Hd3aONMNkGJ8vSCH5QjicuDm92VtVwkAAACEjIoZSLYlJvPD+odL+lFzVQSmkneW7
VCGqYQIDAQABAKAcfNpg6GCQxoneLOghv1UrRotNZGvqpUOEAvHK3X7AJhz5SU4V
an36qvsAt5ghFMVM2iGvGaXbj0dAd+Jg64pxAiEA32Eh9mPtFSmZhTIUMeGcPmPk
qIYCEuP8a/ZLmI9s4TsCIQDWvLQuvjSVfwPhi0TFAb5wqAET8X5LBFqtGX5QlUep
EwIgfFnqM02Gc4wtLoqa2d4qPkYu13+uUW9hLd4XSd6i/OS8CIQDT3elU+Rt+qIwW
u0cFrVvNYSV3HNzDfS9N/IoxTagfewIgpvXADe5c2EWbhCUkhN+ZCf38AKewK9TW
lQcDy4L+f14= -----END PRIVATE KEY-----",
  "public_certificate": "-----BEGIN CERTIFICATE-----
MIIBuzCCAwwGawIBAgIIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQAwHdENMAsgA1UE
AxMEVEVTVDELMAkGA1UEBhMCVVMwHhcNMTgwNjA4MTgwOTAxWhcNMTkwNjA4MTgw
OTAxWjAcMQ0wCwYDVQQDEwRURVNUMQswCQYDVQQGEwJVUzBcMA0GCsGSIb3DQEBA
QUAA0sAMEgCQQDaPvbqUJjFJ6NNTyK3Yb+ytSjJ9aa3yUmYTD9uMiP+6ycjxHWB
e8u9z6yCHsW03ync+dnhE5c5z8wuDAY0fv15AgMBAAGjgYowgYcwDAYDVR0TBAAUw
AwEB/zALBgNVHQ8EBAMCAQYwHQYDVR0OBByEFMJ7Ev/o/3+YNzYh5XNlqqjnw4zm
MEsGA1UdIwREMEKAFMJ7Ev/o/3+YNzYh5XNlqqjnw4zmoSCkhjAcMQ0wCwYDVQQD
EwRURVNUMQswCQYDVQQGEwJVU4IIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQADQQA
vDovYeyGNknjGI+TVNX6nDbyzf7zUPqnri0KuvObEeybrbPW45sgsnT5dyeE/32U
9Yr6lklknbTvbDTmLnrC -----END CERTIFICATE-----",
  "scope": "svm",
  "serial_number": "string",
  "subject_key_identifier":
  "26:1F:C5:53:5B:D7:9E:E2:37:74:F4:F4:06:09:03:3D:EB:41:75:D8",
  "svm": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    }
  },
}
```

```
    "name": "svm1",
    "uuid": "02c9e252-41be-11e9-81d5-00a0986138f7"
  },
  "type": "client",
  "uuid": "string"
}
```

Response

Status: 201, Created

| Name | Type | Description |
|------------------------|---|-------------------|
| _links | _links | |
| num_records | integer | Number of records |
| records | array[security_certificate] | |

Example response

```
{
  "_links": {
    "next": {
      "href": "/api/resourcelink"
    },
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "records": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    },
    "authority_key_identifier":
"26:1F:C5:53:5B:D7:9E:E2:37:74:F4:F4:06:09:03:3D:EB:41:75:D7",
    "ca": "string",
    "common_name": "test.domain.com",
    "hash_function": "sha1",
    "intermediate_certificates": {
    },
    "name": "cert1",
    "private_key": "-----BEGIN PRIVATE KEY-----
MIIBVAIBADANBgkqhkiG9w0BAQEFAASCAT4wggE6AgEAAkEAu1/a8f3G47cZ6pel
Hd3aONMNkGJ8vSCH5QjicuDm92VtVwkAAACEjIoZSLYlJvPD+odL+lFzVQSmkneW7
VCgqYQIDAQABAkAcfNpg6GCQxoneLOghvlUrRotNZGvqpUOEAvHK3X7AJhz5SU4V
an36qvsAt5ghFMVM2iGvGaXbj0dAd+Jg64pxAiEA32Eh9mPtFSmZhTIUMeGcPmPk
qIYCEuP8a/ZLmI9s4TsciQDWvLQuvjSVfwPhi0TFab5wqAET8X5LBFqtGX5QlUep
EwIgfngM02Gc4wtLoqa2d4qPkYu13+uUW9hLd4Xsd6i/OS8CIQDT3elU+Rt+qIwW
u0cFrVvNYSV3HNzDfS9N/IoxTagfewIgpVxADe5c2EWbhCUkhN+ZCf38AKewK9TW
lQcDy4L+f14= -----END PRIVATE KEY-----",
    "public_certificate": "-----BEGIN CERTIFICATE-----
MIIBuzCCAwwGAWIBAgIIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQAwHdENMAsGA1UE
AxMEVEVTVDELMAkGA1UEBhMCVVMwHhcNMjgwNjA4MTgwOTAxWhcNMjgwNjA4MTgw
OTAxWjAcMQ0wCwYDVQQDEwRURVNUMQswCQYDVQQGEwJVUzBcMA0GCSqGSIb3DQEB
AQUAA0sAMEgCQQDaPvbqUJjFJ6NNTyK3Yb+ytSjJ9aa3yUmYTD9uMiP+6ycjxHWB
e8u9z6yCHsW03ync+dnhE5c5z8wuDAY0fv15AgMBAAGjgYowgYcwDAYDVR0TBAUw
AwEB/zALBgNVHQ8EBAMCAQYwHQYDVR0OBBYEFMJ7Ev/o/3+YNzYh5XNlqqjnw4zm
MEsGA1UdIwREMEKAFMJ7Ev/o/3+YNzYh5XNlqqjnw4zmoSCKHjAcMQ0wCwYDVQQD
EwRURVNUMQswCQYDVQQGEwJVU4IIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQADQQA
vDovYeyGNknjGI+TVNX6nDbyzf7zUPqnri0KuvObEeybrbPW45sgsnT5dyeE/32U
9Yr6lklknbTtVBDTmLnrc -----END CERTIFICATE-----",
    "scope": "svm",
  }
}
```

```

"serial_number": "string",
"subject_key_identifier":
"26:1F:C5:53:5B:D7:9E:E2:37:74:F4:F4:06:09:03:3D:EB:41:75:D8",
"svm": {
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "name": "svm1",
  "uuid": "02c9e252-41be-11e9-81d5-00a0986138f7"
},
"type": "client",
"uuid": "string"
}
}

```

Error

Status: Default

ONTAP Error Response Codes

| Error Code | Description |
|------------|---|
| 3735645 | Cannot specify a value for serial. It is generated automatically. |
| 3735622 | The certificate type is not supported. |
| 3735664 | The specified key size is not supported in FIPS mode. |
| 3735665 | The specified hash function is not supported in FIPS mode. |
| 3735553 | Failed to create self-signed Certificate. |
| 3735646 | Failed to store the certificates. |
| 3735693 | The certificate installation failed as private key was empty. |
| 3735618 | Cannot accept private key for server_ca or client_ca. |
| 52363365 | Failed to allocate memory. |
| 52559975 | Failed to read the certificate due to incorrect formatting. |
| 52363366 | Unsupported key type. |
| 52560123 | Failed to read the key due to incorrect formatting. |

| Error Code | Description |
|------------|---|
| 52559972 | The certificates start date is later than the current date. |
| 52559976 | The certificate and private key do not match. |
| 52559973 | The certificate has expired. |
| 52363366 | Logic error: use of a dead object. |
| 3735696 | Intermediate certificates are not supported with client_ca and server_ca type certificates. |
| 52559974 | The certificate is not supported in FIPS mode. |
| 3735676 | Cannot continue the installation without a value for the common name. Since the subject field in the certificate is empty, the field "common_name" must have a value to continue with the installation. |
| 3735558 | Failed to extract information about Common Name from the certificate. |
| 3735588 | The common name (CN) extracted from the certificate is not valid. |
| 3735632 | Failed to extract Certificate Authority Information from the certificate. |

| Name | Type | Description |
|-------|-----------------------|-------------|
| error | error | |

Example error

```

{
  "error": {
    "arguments": {
      "code": "string",
      "message": "string"
    },
    "code": "4",
    "message": "entry doesn't exist",
    "target": "uuid"
  }
}

```

Definitions

See Definitions

href

| Name | Type | Description |
|------|--------|-------------|
| href | string | |

_links

| Name | Type | Description |
|------|----------------------|-------------|
| self | href | |

svm

| Name | Type | Description |
|--------|------------------------|-----------------------------------|
| _links | _links | |
| name | string | The name of the SVM. |
| uuid | string | The unique identifier of the SVM. |

security_certificate

| Name | Type | Description |
|--------------------------|------------------------|---|
| _links | _links | |
| authority_key_identifier | string | Provides the key identifier of the issuing CA certificate that signed the SSL certificate. |
| ca | string | Certificate authority |
| common_name | string | FQDN or custom common name. Provide on POST when creating a self-signed certificate. |
| expiry_time | string | Certificate expiration time. Can be provided on POST if creating self-signed certificate. The expiration time range is between 1 day to 10 years. |
| hash_function | string | Hashing function. Can be provided on POST when creating a self-signed certificate. Hash functions md5 and sha1 are not allowed on POST. |

| Name | Type | Description |
|---------------------------|---------------|---|
| intermediate_certificates | array[string] | Chain of intermediate Certificates in PEM format. Only valid in POST when installing a certificate. |
| key_size | integer | Key size of requested Certificate in bits. One of 512, 1024, 1536, 2048, 3072. Can be provided on POST if creating self-signed certificate. Key size of 512 is not allowed on POST. |
| name | string | Certificate name. If not provided in POST, a unique name specific to the SVM is automatically generated. |
| private_key | string | Private key Certificate in PEM format. Only valid for create when installing a CA-signed certificate. This is not audited. |
| public_certificate | string | Public key Certificate in PEM format. If this is not provided in POST, a self-signed certificate is created. |
| scope | string | Set to "svm" for interfaces owned by an SVM. Otherwise, set to "cluster". |
| serial_number | string | Serial number of certificate. |
| subject_key_identifier | string | Provides the key identifier used to identify the public key in the SSL certificate. |
| svm | svm | |

| Name | Type | Description |
|------|--------|---|
| type | string | Type of Certificate. The following types are supported: <ul style="list-style-type: none"> • client - a certificate and its private key used by an SSL client in ONTAP. • server - a certificate and its private key used by an SSL server in ONTAP. • client_ca - a Certificate Authority certificate used by an SSL server in ONTAP to verify an SSL client certificate. • server_ca - a Certificate Authority certificate used by an SSL client in ONTAP to verify an SSL server certificate. • root_ca - a self-signed certificate used by ONTAP to sign other certificates by acting as a Certificate Authority. • enum: ["client", "server", "client_ca", "server_ca", "root_ca"] • Introduced in: 9.6 |
| uuid | string | Unique ID that identifies a certificate. |

_links

| Name | Type | Description |
|------|----------------------|-------------|
| next | href | |
| self | href | |

error_arguments

| Name | Type | Description |
|---------|--------|------------------|
| code | string | Argument code |
| message | string | Message argument |

error

| Name | Type | Description |
|-----------|--|---|
| arguments | array[error_arguments] | Message arguments |
| code | string | Error code |
| message | string | Error message |
| target | string | The target parameter that caused the error. |

Sign security certificates

POST /security/certificates/{ca.uuid}/sign

Introduced In: 9.6

Signs a certificate.

Required properties

- `signing_request` - Certificate signing request to be signed by the given certificate authority.

Recommended optional properties

- `expiry_time` - Certificate expiration time. Specifying an expiration time for a signed certificate is recommended.
- `hash_function` - Hashing function. Specifying a strong hashing function is recommended when signing a certificate.

Default property values

If not specified in POST, the following default property values are assigned:

- `expiry_time` - *P365DT*
- `hash_function` - *sha256*

Related ONTAP commands

- `security certificate sign` This API is used to sign a certificate request using a pre-existing self-signed root certificate. The self-signed root certificate acts as a certificate authority within its scope and maintains the records of its signed certificates.

The root certificate can be created for a given SVM or for the cluster using [POST `security/certificates`].

Parameters

| Name | Type | In | Required | Description |
|----------------|---------|-------|----------|---|
| ca.uuid | string | path | True | UUID of the existing certificate authority certificate |
| return_records | boolean | query | False | The default is false. If set to true, the records are returned. <ul style="list-style-type: none">• Default value: |

Request Body

| Name | Type | Description |
|-----------------|--------|--|
| expiry_time | string | Certificate expiration time. The allowed expiration time range is between 1 day to 10 years. |
| hash_function | string | Hashing function |
| signing_request | string | Certificate signing request to be signed by the given certificate authority. Request should be in X509 PEM format. |

Example request

```
{
  "hash_function": "sha256",
  "signing_request": "'-----BEGIN CERTIFICATE REQUEST-----
MIICYDCCAUGCAQAwGzEMMAoGA1UEAxMDQUJDMQswCQYDVQQGEwJVUzCCASIwDQYJ
KoZIhvcNAQEBBQADggEPADCCAQoCggEBAPF+82SlqT3Vyu3Jx4IAwHcO5EGwLOxy
zQ6KNjz71Fca0n1/A1CbCPyOsSupGVObvdWxX7xLVMJ2SXB7h43GCqYyX6FXJO4F
HOpmLvB+jxdeiW7SDbiZyLUlsvA+oRO/uNlcug773QZdKLjJD64erZZMRUNbUJB8
bARxAUi0FPvgTraSQ0UW5sRLiGKeAyKA4wekYe1VgjHRTBizFbD4dI3njfva/2Bl
jf+kkulgcLJTUJNtkgeimqMKYraYuleYcYk2K+C//0NuNOuPbDfTXCM7O61vik09
Szi8nLN70XE9KoAA93U/BCpSfpl8XIb4cGnEr8hgVHOotZSo+KZBFxMCAwEAAaAA
MA0GCSqGSIB3DQEBChwUAA4IBAQC2vFYpvgsFrm5GnPx8tOBD1xsTyYjbWJMD8hAF
lFrvF9Sw9QGctDyacxkwgJhQx8l8JiIS5GOY6WWLB19FMkLQNAhDL9xF3WF7vfYq
RKgrz3bd/Vg96fsRZNYIPLGmoEaqLOh3FOCGc2VbdsR9PwOn3fwthxkIRd6ds6/q
jc5cpSmVsCOgu+OKcpRXikYDbkWXfTZ1AhSfn6njBYFdZ9+PNAu/0JRQh5bX60nO
5heniTcAJLWUZP/CQ8nxHY0Wqy+lrAtM33d5cVmHULBXQSIru/0ZkA/b9fK5Zv8E
ZMADYUoEvIG59VxhyCi8lzYf+Mxl8qBSF+ZdC4yWhzDqZtM9 -----END CERTIFICATE
REQUEST-----'"
}
```

Response

Status: 200, Ok

| Name | Type | Description |
|--------------------|--------|----------------------------------|
| public_certificate | string | CA signed public key Certificate |

Error

Status: Default

ONTAP Error Response Codes

| Error Code | Description |
|------------|--|
| 3735628 | Failed to use CA certificate for signing. |
| 3735665 | The specified hash function is not supported in FIPS mode. |
| 52559974 | The certificate is not supported in FIPS mode. |

| Error Code | Description |
|------------|---|
| 3735626 | Failed to generate signed Certificate. |
| 3735558 | Failed to extract information about Common Name from the certificate. |
| 3735588 | The common name (CN) extracted from the certificate is not valid. |
| 3735632 | Failed to extract Certificate Authority Information from the certificate. |
| 3735629 | Failed to sign the certificate because Common Name of signing certificate and Common Name of CA certificate are same. |
| 3735630 | Failed to sign the certificate because expiry date of signing certificate exceeds the expiry date of CA certificate. |

| Name | Type | Description |
|-------|-------|-------------|
| error | error | |

Example error

```
{
  "error": {
    "arguments": {
      "code": "string",
      "message": "string"
    },
    "code": "4",
    "message": "entry doesn't exist",
    "target": "uuid"
  }
}
```

Definitions

See Definitions

security_certificate_sign

| Name | Type | Description |
|-----------------|--------|--|
| expiry_time | string | Certificate expiration time. The allowed expiration time range is between 1 day to 10 years. |
| hash_function | string | Hashing function |
| signing_request | string | Certificate signing request to be signed by the given certificate authority. Request should be in X509 PEM format. |

error_arguments

| Name | Type | Description |
|---------|--------|------------------|
| code | string | Argument code |
| message | string | Message argument |

error

| Name | Type | Description |
|-----------|--|---|
| arguments | array[error_arguments] | Message arguments |
| code | string | Error code |
| message | string | Error message |
| target | string | The target parameter that caused the error. |

Delete security certificates

DELETE /security/certificates/{uuid}

Introduced In: 9.6

Deletes a security certificate.

Related ONTAP commands

- `security certificate delete`

Parameters

| Name | Type | In | Required | Description |
|------|--------|------|----------|------------------|
| uuid | string | path | True | Certificate UUID |

Response

Status: 200, Ok

Error

Status: Default

ONTAP Error Response Codes

| Error Code | Description |
|------------|--|
| 3735644 | Cannot delete server-chain certificate. Reason: There is a corresponding server certificate for it. |
| 3735679 | Cannot delete pre-installed server_ca certificates through REST. Use CLI or ZAPI. |
| 3735650 | Deleting this client_ca certificate directly is not supported. Delete the corresponding root-ca certificate using type <code>root_ca</code> to delete the root, client, and server certificates. |
| 3735627 | Deleting this server_ca certificate directly is not supported. Delete the corresponding root-ca certificate using type <code>root_ca</code> to delete the root, client, and server certificates. |
| 3735589 | Cannot delete certificate. |
| 3735590 | Cannot delete certificate. Failed to remove SSL configuration for the certificate. |
| 3735683 | Cannot remove this certificate while external key manager is configured. |

| Name | Type | Description |
|-------|-----------------------|-------------|
| error | error | |

Example error

```
{
  "error": {
    "arguments": {
      "code": "string",
      "message": "string"
    },
    "code": "4",
    "message": "entry doesn't exist",
    "target": "uuid"
  }
}
```

Definitions

See Definitions

error_arguments

| Name | Type | Description |
|---------|--------|------------------|
| code | string | Argument code |
| message | string | Message argument |

error

| Name | Type | Description |
|-----------|--|---|
| arguments | array[error_arguments] | Message arguments |
| code | string | Error code |
| message | string | Error message |
| target | string | The target parameter that caused the error. |

Retrieve security certificates

GET /security/certificates/{uuid}

Introduced In: 9.6

Retrieves security certificates.

Related ONTAP commands

- `security certificate show`

Parameters

| Name | Type | In | Required | Description |
|--------|---------------|-------|----------|-------------------------------|
| uuid | string | path | True | Certificate UUID |
| fields | array[string] | query | False | Specify the fields to return. |

Response

Status: 200, Ok

| Name | Type | Description |
|---------------------------|------------------------|---|
| _links | _links | |
| authority_key_identifier | string | Provides the key identifier of the issuing CA certificate that signed the SSL certificate. |
| ca | string | Certificate authority |
| common_name | string | FQDN or custom common name. Provide on POST when creating a self-signed certificate. |
| expiry_time | string | Certificate expiration time. Can be provided on POST if creating self-signed certificate. The expiration time range is between 1 day to 10 years. |
| hash_function | string | Hashing function. Can be provided on POST when creating a self-signed certificate. Hash functions md5 and sha1 are not allowed on POST. |
| intermediate_certificates | array[string] | Chain of intermediate Certificates in PEM format. Only valid in POST when installing a certificate. |

| Name | Type | Description |
|------------------------|---------------------|---|
| key_size | integer | Key size of requested Certificate in bits. One of 512, 1024, 1536, 2048, 3072. Can be provided on POST if creating self-signed certificate. Key size of 512 is not allowed on POST. |
| name | string | Certificate name. If not provided in POST, a unique name specific to the SVM is automatically generated. |
| private_key | string | Private key Certificate in PEM format. Only valid for create when installing a CA-signed certificate. This is not audited. |
| public_certificate | string | Public key Certificate in PEM format. If this is not provided in POST, a self-signed certificate is created. |
| scope | string | Set to "svm" for interfaces owned by an SVM. Otherwise, set to "cluster". |
| serial_number | string | Serial number of certificate. |
| subject_key_identifier | string | Provides the key identifier used to identify the public key in the SSL certificate. |
| svm | svm | |

| Name | Type | Description |
|------|--------|--|
| type | string | <p>Type of Certificate. The following types are supported:</p> <ul style="list-style-type: none"> • client - a certificate and its private key used by an SSL client in ONTAP. • server - a certificate and its private key used by an SSL server in ONTAP. • client_ca - a Certificate Authority certificate used by an SSL server in ONTAP to verify an SSL client certificate. • server_ca - a Certificate Authority certificate used by an SSL client in ONTAP to verify an SSL server certificate. • root_ca - a self-signed certificate used by ONTAP to sign other certificates by acting as a Certificate Authority. • enum: ["client", "server", "client_ca", "server_ca", "root_ca"] • Introduced in: 9.6 |
| uuid | string | Unique ID that identifies a certificate. |

Example response

```
{
  "_links": {
    "self": {
      "href": "/api/resourcelink"
    }
  },
  "authority_key_identifier":
  "26:1F:C5:53:5B:D7:9E:E2:37:74:F4:F4:06:09:03:3D:EB:41:75:D7",
  "ca": "string",
  "common_name": "test.domain.com",
  "hash_function": "sha1",
  "intermediate_certificates": {
  },
  "name": "cert1",
  "private_key": "-----BEGIN PRIVATE KEY-----
MIIBVAIBADANBqkqhkiG9w0BAQEFAASCAT4wggE6AgEAAkEAu1/a8f3G47cZ6pel
Hd3aONMNkGJ8vSCH5QjicuDm92VtVwkAACEjIoZSLYlJvPD+odL+lFzVQSmkneW7
VCGqYQIDAQABAKAcfNpg6GCQxoneLOghv1UrRotNZGvqpUOEAvHK3X7AJhz5SU4V
an36qvsAt5ghFMVM2iGvGaXbj0dAd+Jg64pxAiEA32Eh9mPtFSmZhTIUMeGcPmPk
qIYCEuP8a/ZLmI9s4TsCIQDWvLQuvjSVfwPhi0TFAb5wqAET8X5LBFqtGX5QlUep
EwIgfFnqM02Gc4wtLoqa2d4qPkYu13+uUW9hLd4XSd6i/OS8CIQDT3elU+Rt+qIwW
u0cFrVvNYSV3HNzDfS9N/IoxTagfewIgpvXADe5c2EWbhCUkhN+ZCf38AKewK9TW
lQcDy4L+f14= -----END PRIVATE KEY-----",
  "public_certificate": "-----BEGIN CERTIFICATE-----
MIIBuzCCAwwGawIBAgIIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQAwHdENMAsGA1UE
AxMEVEVETVDELMakGA1UEBhMCMVVMwHhcNMTgwNjA4MTgwOTAxWhcNMTkwNjA4MTgw
OTAxWjAcMQ0wCwYDVQQDEwRURVNUMQswCQYDVQQGEwJVUzBcMA0GCSqGSIb3DQEB
AQUAA0sAMEgCQQDaPvbqUJFJ6NNTyK3Yb+ytSjJ9aa3yUmYTD9uMiP+6ycjxHWB
e8u9z6yCHsW03ync+dnhE5c5z8wuDAY0fv15AgMBAAGjgYowgYcwDAYDVR0TBAAUw
AwEB/zALBgNVHQ8EBAMCAQYwHQYDVR0OBByEFMJ7Ev/o/3+YNzYh5XNlqqjnw4zm
MEsGA1UdIwREMEKAFMJ7Ev/o/3+YNzYh5XNlqqjnw4zmoSCkhjAcMQ0wCwYDVQQD
EwRURVNUMQswCQYDVQQGEwJVU4IIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQADQQAy
DovYeyGNknjGI+TVNX6nDbyzf7zUPqnri0KuvObEeybrbPW45sgsnT5dyeE/32U
9Yr6lklknbTvbDTmLnrC -----END CERTIFICATE-----",
  "scope": "svm",
  "serial_number": "string",
  "subject_key_identifier":
  "26:1F:C5:53:5B:D7:9E:E2:37:74:F4:F4:06:09:03:3D:EB:41:75:D8",
  "svm": {
    "_links": {
      "self": {
        "href": "/api/resourcelink"
      }
    }
  },
}
```

```
    "name": "svm1",
    "uuid": "02c9e252-41be-11e9-81d5-00a0986138f7"
  },
  "type": "client",
  "uuid": "string"
}
```

Error

Status: Default, Error

| Name | Type | Description |
|-------|-------|-------------|
| error | error | |

Example error

```
{
  "error": {
    "arguments": {
      "code": "string",
      "message": "string"
    },
    "code": "4",
    "message": "entry doesn't exist",
    "target": "uuid"
  }
}
```

Definitions

See Definitions

href

| Name | Type | Description |
|------|--------|-------------|
| href | string | |

_links

| Name | Type | Description |
|------|----------------------|-------------|
| self | href | |

svm

| Name | Type | Description |
|--------|------------------------|-----------------------------------|
| _links | _links | |
| name | string | The name of the SVM. |
| uuid | string | The unique identifier of the SVM. |

error_arguments

| Name | Type | Description |
|---------|--------|------------------|
| code | string | Argument code |
| message | string | Message argument |

error

| Name | Type | Description |
|-----------|--|---|
| arguments | array[error_arguments] | Message arguments |
| code | string | Error code |
| message | string | Error message |
| target | string | The target parameter that caused the error. |

Copyright information

Copyright © 2024 NetApp, Inc. All Rights Reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

LIMITED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (b)(3) of the Rights in Technical Data -Noncommercial Items at DFARS 252.227-7013 (FEB 2014) and FAR 52.227-19 (DEC 2007).

Data contained herein pertains to a commercial product and/or commercial service (as defined in FAR 2.101) and is proprietary to NetApp, Inc. All NetApp technical data and computer software provided under this Agreement is commercial in nature and developed solely at private expense. The U.S. Government has a non-exclusive, non-transferrable, nonsublicensable, worldwide, limited irrevocable license to use the Data only in connection with and in support of the U.S. Government contract under which the Data was delivered. Except as provided herein, the Data may not be used, disclosed, reproduced, modified, performed, or displayed without the prior written approval of NetApp, Inc. United States Government license rights for the Department of Defense are limited to those rights identified in DFARS clause 252.227-7015(b) (FEB 2014).

Trademark information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.