Manage security certificates

REST API reference

NetApp
September 12, 2025

This PDF was generated from https://docs.netapp.com/us-en/ontap-restapi-9131/security_certificates_endpoint_overview.html on September 12, 2025. Always check docs.netapp.com for the latest.

Table of Contents

Manage security certificates . . . 1
- Security certificates endpoint overview . . . 1
  - Overview . . . 1
  - Installing certificates in ONTAP . . . 1
  - Examples . . . 1
- Retrieve security certificates . . . 11
  - Related ONTAP commands . . . 11
  - Parameters . . . 11
  - Response . . . 14
  - Error . . . 16
  - Definitions . . . 17
- Create or install security certificates . . . 21
  - Required properties . . . 21
  - Recommended optional properties . . . 21
  - Default property values . . . 21
  - Related ONTAP commands . . . 22
  - Parameters . . . 22
  - Request Body . . . 22
  - Response . . . 26
  - Error . . . 28
  - Definitions . . . 29
- Sign security certificates . . . 33
  - Required properties . . . 33
  - Recommended optional properties . . . 33
  - Default property values . . . 33
  - Related ONTAP commands . . . 33
  - Parameters . . . 33
  - Request Body . . . 34
  - Response . . . 35
  - Error . . . 35
  - Definitions . . . 36
- Delete security certificates . . . 37
  - Related ONTAP commands . . . 38
  - Parameters . . . 38
  - Response . . . 38
  - Error . . . 38
  - Definitions . . . 39
- Retrieve security certificates . . . 40
  - Related ONTAP commands . . . 40
  - Parameters . . . 40
  - Response . . . 40
  - Error . . . 44
  - Definitions . . . 45

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
MwMQswCQYDVQQGEwJVUzAeFw0xODA3MTgxOTI5MTRaFw0xOTA3MTgxOTI5\nMTRaMBsxDDAKBg
NVBAMTA3ZzMDELMAkGA1UEBhMCVVMwggEiMA0GCSqGSIb3DQEB\nAQUAA4IBDwAwggEKAoIBAQ
CqFQb27th2ACOmJvWgLh1xRzobSb2ZTQfO561faXQ3\nIbiT+rnRWXetd/s2+iCv91d9LW0NOm
P3MN2f3SFbyze3dl7WrnVbjLmYuI9MfOxs\nfmA+Bh6gpap5Yn2YddqoV6rfNGAuUveNLArNl8
wODk/mpawpEQ93QSa1Zfg1gnoH\nRFrYqiSYT06X5g6RbUuEl4LTGXspz+plU46Za0i6QyxtvZ
4bneibffXN3IigpqI6\nTGUV8R/J3Ps338VxVmSO9ZXBZmvbcJVoysYNICl/oi3fgPZlnBv0tb
swqg4FoZO/\nWT+XHGhLep6cr/Aqg7u6C4RfqbCwzB/XFKDIqnmAQkDBAgMBAAGjgYkwgYYwDA
YD\nVR0TBAUwAwEB/zALBgNVHQ8EBAMCAQYwHQYDVR0OBBYEFN/AnH8qLxocTtumNHIn\nEN4I
FIDBMEoGA1UdIwRDMEGAFN/AnH8qLxocTtumNHInEN4IFIDBoR+kHTAbMQww\nCgYDVQQDEwN2
czAxCzAJBgNVBAYTAlVTgggVQo1Fz4HPVjANBgkqhkiG9w0BAQsF\nAAOCAQEAa0pUEepdeQnd
2Amwg8UFyxayb8eu3E6dlptvtyp+xtjhIC7Dh95CVXhy\nkJS3Tsu60PGR/b2vc3MZtAUpcL4c
eD8XntKPQgBlqoB4bRogCe1TnlGswRXDX5TS\ngMVrRjaWTBF7ikT4UjR05rSxcDGplQRqjnOt
hqi+yPT+29+8a4Uu6J+3Kdrflj4p\n1nSWpuB9EyxtuCILNqXA2ncH7YKtoeNtChKCchhvPcoT
y6Opma6UQn5UMxstkvGT\nVGaN5TlRWv0yiqPXIQblSqXi/uQsuRPcHDu7+KWRFn08USa6QVo2
mDs9P7R9dd0K\n9QAsTjTOF9PlAKgNxGoOJl2y0+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-----
\nMIIFYDCCA0igAwIBAgIQCgFCgAAAAUUjyES1AAAAAjANBgkqhkiG9w0BAQsFADBKMQswCQYD
VQQG\nEwJVUzESMBAGA1UEChMJSWRlblRydXN0MScwJQYDVQQDEx5JZGVuVHJ1c3QgQ29tbWVy
Y2lhbCBS\nb290IENBIDEwHhcNMTQwMTE2MTgxMjIzWhcNMzQwMTE2MTgxMjIzWjBKMQswCQYD
VQQGEwJVUzES\nMBAGA1UEChMJSWRlblRydXN0MScwJQYDVQQDEx5JZGVuVHJ1c3QgQ29tbWVy
Y2lhbCBSb290IENB\nIDEwggIiMA0GCSqGSIb3DQEBAQUAA4ICDwAwggIKAoICAQCnUBneP5k9
1DNG8W9RYYKyqU+PZ4ld\nhNlT3Qwo2dfw/66VQ3KZ+bVdfIrBQuExUHTRgQ18zZshq0PirK1e
hm7zCYofWjK9ouuU+ehcCuz/\nmNKvcbO0U59Oh++SvL3sTzIwiEsXXlfEU8L2ApeN2WIrvyQf
Yo3fw7gpS0l4PJNgiCL8mdo2yMKi\n1CxUAGc1bnO/AljwpN3lsKImesrgNqUZFvX9t++uP0D1
bVoE/c40yiTcdCMbXTMTEl3EASX2MN0C\nXZ/g1Ue9tOsbobtJSdifWwLziuQkkORiT0/Br4sO
dBeo0XKIanoBScy0RnnGF7HamB4HWfp1IYVl\n3ZBWzvurpWCdxJ35UrCLvYf5jysjCiN2O/cz
4ckA82n5S6LgTrx+kzmEB/dEcH7+B1rlsazRGMzy\nNeVJSQjKVsk9+w8YfYs7wRPCTY/JTw43
6R+hDmrfYi7LNQZReSzIJTj0+kuniVyc0uMNOYZKdHzV\nWYfCP04MXFL0PfdSgvHqo6z9STQa
KPNBiDoT7uje/5kdX7rL6B7yuVBgwDHTc+XvvqDtMwt0viAg\nxGds8AgDelWAf0ZOlqf0Hj7h
9tgJ4TNkK2PXMl6f+cB7D3hvl7yTmvmcEpB4eoCHFddydJxVdHix\nuuFucAS6T6C6aMN7/zHw
cz09lCqxC0EOoP5NiGVreTO01wIDAQABo0IwQDAOBgNVHQ8BAf8EBAMC\nAQYwDwYDVR0TAQH/
BAUwAwEB/zAdBgNVHQ4EFgQU7UQZwNPwBovupHu+QucmVMiONnYwDQYJKoZI\nhvcNAQELBQAD
ggIBAA2ukDL2pkt8RHYZYR4nKM1eVO8lvOMIkPkp165oCOGUAFjvLi5+U1KMtlwH\n6oi6mYtQ
lNeCgN9hCQCTrQ0U5s7B8jeUeLBfnLOic7iPBZM4zY0+sLj7wM+x8uwtLRvM7Kqas6pg\nghst
O8OEPVeKlh6cdbjTMM1gCIOQ045U8U1mwF10A0Cj7oV+wh93nAbowacYXVKV7cndJZ5t+qnt\n
ozo00Fl72u1Q8zW/7esUTTHHYPTa8Yec4kjixsU3+wYQ+nVZZjFHKdp2mhzpgq7vmrlR94gjmm
mV\nYjzlVYA211QC//G5Xc7UI2/YRYRKW2XviQzdFKcgyxilJbQN+QHwotL0AMh0jqEqSI5l2x
PE4iUX\nfeu+h1sXIFRRk0pTAwvsXcoz7WL9RccvW9xYoIA55vrX/hMUpu09lEpCdNTDd1lzzY
9GvlU47/ro\nkTLql1gEIt44w8y8bckzOmoKaT+gyOpyj4xjhiO9bTyWnpXgSUyqorkqG5w2gX
jtw+hG4iZZRHUe\n2XWJUc0QhJ1hYMtd+ZciTY6Y5uN/9lu7rs3KSoFrXgvzUeF0K+l+J6fZmU
lO+KWA2yUPHGNiiskz\nZ2s8EIPGrd6ozRaOjfAHN3Gf8qv8QfXBi+wAN10J5U6A7/qxXDgGpR
tK4dw4LTzcqx+QGtVKnO7R\ncGzM7vRX+Bi6hG6H\n-----END CERTIFICATE-----\n\"
}"
```

Installing a certificate on a specific SVM

```
# 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 "{   \"svm\" : {
\"name\" : \"vs0\" }, \"type\": \"server_ca\", \"public_certificate\":
\"-----BEGIN CERTIFICATE-----
\nMIIFYDCCA0igAwIBAgIQCgFCgAAAAUUjyES1AAAAAjANBgkqhkiG9w0BAQsFADBKMQswCQYD
VQQG\nEwJVUzESMBAGA1UEChMJSWRlblRydXN0MScwJQYDVQQDEx5JZGVuVHJ1c3QgQ29tbWVy
Y2lhbCBS\nb290IENBIDEwHhcNMTQwMTE2MTgxMjIzWhcNMzQwMTE2MTgxMjIzWjBKMQswCQYD
VQQGEwJVUzES\nMBAGA1UEChMJSWRlblRydXN0MScwJQYDVQQDEx5JZGVuVHJ1c3QgQ29tbWVy
Y2lhbCBSb290IENB\nIDEwggIiMA0GCSqGSIb3DQEBAQUAA4ICDwAwggIKAoICAQCnUBneP5k9
1DNG8W9RYYKyqU+PZ4ld\nhNlT3Qwo2dfw/66VQ3KZ+bVdfIrBQuExUHTRgQ18zZshq0PirK1e
hm7zCYofWjK9ouuU+ehcCuz/\nmNKvcbO0U59Oh++SvL3sTzIwiEsXXlfEU8L2ApeN2WIrvyQf
Yo3fw7gpS0l4PJNgiCL8mdo2yMKi\n1CxUAGc1bnO/AljwpN3lsKImesrgNqUZFvX9t++uP0D1
bVoE/c40yiTcdCMbXTMTEl3EASX2MN0C\nXZ/g1Ue9tOsbobtJSdifWwLziuQkkORiT0/Br4sO
dBeo0XKIanoBScy0RnnGF7HamB4HWfp1IYVl\n3ZBWzvurpWCdxJ35UrCLvYf5jysjCiN2O/cz
4ckA82n5S6LgTrx+kzmEB/dEcH7+B1rlsazRGMzy\nNeVJSQjKVsk9+w8YfYs7wRPCTY/JTw43
6R+hDmrfYi7LNQZReSzIJTj0+kuniVyc0uMNOYZKdHzV\nWYfCP04MXFL0PfdSgvHqo6z9STQa
KPNBiDoT7uje/5kdX7rL6B7yuVBgwDHTc+XvvqDtMwt0viAg\nxGds8AgDelWAf0ZOlqf0Hj7h
9tgJ4TNkK2PXMl6f+cB7D3hvl7yTmvmcEpB4eoCHFddydJxVdHix\nuuFucAS6T6C6aMN7/zHw
cz09lCqxC0EOoP5NiGVreTO01wIDAQABo0IwQDAOBgNVHQ8BAf8EBAMC\nAQYwDwYDVR0TAQH/
BAUwAwEB/zAdBgNVHQ4EFgQU7UQZwNPwBovupHu+QucmVMiONnYwDQYJKoZI\nhvcNAQELBQAD
ggIBAA2ukDL2pkt8RHYZYR4nKM1eVO8lvOMIkPkp165oCOGUAFjvLi5+U1KMtlwH\n6oi6mYtQ
lNeCgN9hCQCTrQ0U5s7B8jeUeLBfnLOic7iPBZM4zY0+sLj7wM+x8uwtLRvM7Kqas6pg\nghst
O8OEPVeKlh6cdbjTMM1gCIOQ045U8U1mwF10A0Cj7oV+wh93nAbowacYXVKV7cndJZ5t+qnt\n
ozo00Fl72u1Q8zW/7esUTTHHYPTa8Yec4kjixsU3+wYQ+nVZZjFHKdp2mhzpgq7vmrlR94gjmm
mV\nYjzlVYA211QC//G5Xc7UI2/YRYRKW2XviQzdFKcgyxilJbQN+QHwotL0AMh0jqEqSI5l2x
PE4iUX\nfeu+h1sXIFRRk0pTAwvsXcoz7WL9RccvW9xYoIA55vrX/hMUpu09lEpCdNTDd1lzzY
9GvlU47/ro\nkTLql1gEIt44w8y8bckzOmoKaT+gyOpyj4xjhiO9bTyWnpXgSUyqorkqG5w2gX
jtw+hG4iZZRHUe\n2XWJUc0QhJ1hYMtd+ZciTY6Y5uN/9lu7rs3KSoFrXgvzUeF0K+l+J6fZmU
lO+KWA2yUPHGNiiskz\nZ2s8EIPGrd6ozRaOjfAHN3Gf8qv8QfXBi+wAN10J5U6A7/qxXDgGpR
tK4dw4LTzcqx+QGtVKnO7R\ncGzM7vRX+Bi6hG6H\n-----END CERTIFICATE-----\n\"
}"
```

Deleting a certificate using its UUID

```
# 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-----
\nMIICYTCCAUkCAQAwHDENMAsGA1UEAxMEVEVTVDELMAkGA1UEBhMCVVMwggEiMA0G\nCSqGSI
b3DQEBAQUAA4IBDwAwggEKAoIBAQCiBCuVfbYHNdOO7vjRQja4JqL2cHqK\ndrlTj5hz9RVqFK
Z7VPh8DSP9LoTbYWsvrTkbuD0Wi715MVQCsbkq/mHos+Y5lfqs\nNP5K92fc6EhBzBDYFgZGFn
tZYJjEG5MPerIUE7CfVy7o6sjWOlxeY33pjefObyvP\nBcJkBHg6SFJK/TDLvIYJkonLkJEOJo
TI6++a3I/1bCMfUeuRtLU9ThWlna1kMMYK\n4T16/Bxgm4bha2U2jtosc0Wltnld/capc+eqRV
07WVbMmEOTtop3cv0h3N0S6lbn\nFkd96DXzeGWbSHFHckeCZ9bOHhnVbfEa/efkPLx7ziMC8G
tRHHlwbnK7AgMBAAGg\nADANBgkqhkiG9w0BAQsFAAOCAQEAf+rs1i5PHaOSI2HtTM+Hcv/p71
yzgoLL+aeU\ntB0V4iuoXdqY8oQeWoPI92ci0K08JuSpu6D0DwCKlstfwuGkAA2b0Wr7ZDRonT
Uq\nmJ4j3O47MLysW4Db2LbGws/AuDsCIrBJDWHMpHaqsvRbpMx2xQ/V5oagUw5eGGpN\ne4fg
/E2k9mGkpxwkUzT7w1RZirpND4xL+XTzpzeZqgalpXug4yjIXlI5hpRESZ9/\nAkGJSCWxI15I
ZdxxFVXlBcmm6WpJnnboqkcKeXz95GM6Re+oBy9tlgvwvlVd5s8uHX+bycFiZp09Wsm8Ev727M
ziZ+0II9nxwkDKsdPvam+KLI9hLQ==\n-----END CERTIFICATE REQUEST-----\n\",
\"hash_function\": \"sha256\"}"

# The response:
{
"public_certificate": "-----BEGIN CERTIFICATE-----
\nMIIDBzCCAe+gAwIBAgIIFUKQpcqeaUAwDQYJKoZIhvcNAQELBQAwHDENMAsGA1UE\nAxMEUk
FDWDELMAkGA1UEBhMCVVMwHhcNMTgwNzE4MjAzMTA1WhcNMTkwNzE4MjAz\nMTA1WjAcMQ0wCw
YDVQQDEwRURVNUMQswCQYDVQQGEwJVUzCCASIwDQYJKoZIhvcN\nAQEBBQADggEPADCCAQoCgg
EBAKIEK5V9tgc1047u+NFCNrgmovZweop2uVOPmHP1\nFWoUpntU+HwNI/0uhNthay+tORu4PR
aLvXkxVAKxuSr+Yeiz5jmV+qw0/kr3Z9zo\nSEHMENgWBkYWe1lgmMQbkw96shQTsJ9XLujqyN
Y6XF5jfemN585vK88FwmQEeDpI\nUkr9MMu8hgmSicuQkQ4mhMjr75rcj/VsIx9R65G0tT1OFa
WdrWQwxgrhPXr8HGCb\nhuFrZTaO2ixzRaW2eV39xqlz56pFXTtZVsyYQ5O2indy/SHc3RLqVu
cWR33oNfN4\nZZtIcUdyR4Jn1s4eGdVt8Rr95+Q8vHvOIwLwa1EceXBucrsCAwEAAaNNMEswCQ
YD\nVR0TBAIwADAdBgNVHQ4EFgQUJMPxjeW1G76TbbD2tXB8dwSpI3MwHwYDVR0jBBgw\nFoAU
u5aH0mWR4cFoN9i7k96d2op3sPwwDQYJKoZIhvcNAQELBQADggEBAI5ai+Zi\nFQZUXRTqJCgH
sgBThARneVWQYkYpyAXmTR7QeLf1d4ZHL33i4xWCqX3uvW7SFJLe\nZajT2AVmgiDbaWIHtDtv
qz1BY78PSgUwPH/IyARTEOBeikp6KdwMPraehDIBMAcc\nANY58wXiTBbsl8UMD6tGecgnzw6s
xlMmadGvrfJeJmgY4zert6NNvgtTPhcZQdLS\nE0fGzHS6+3ajCCfEEhPNPeR9D0e5Me81i9Es
QGENrnJzTci8rzXPuF4bC3gghrK1\nI1+kmJQ1kLYVUcsntcrIiHmNvtPFJY6stjDgQKS9aDd/
THhPpokPtZoCmE6PDxh6\nR+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
subject_key_identifier	string	query	False	Filter by subject_key_identifier • Introduced in: 9.8
type	string	query	False	Filter by type

Name	Type	In	Required	Description
private_key	string	query	False	Filter by private_key • Introduced in: 9.8
ca	string	query	False	Filter by ca • maxLength: 256 • minLength: 1
uuid	string	query	False	Filter by uuid • Introduced in: 9.8
name	string	query	False	Filter by name • Introduced in: 9.8
intermediate_certificates	string	query	False	Filter by intermediate_certificates • Introduced in: 9.8
common_name	string	query	False	Filter by common_name
key_size	integer	query	False	Filter by key_size
svm.uuid	string	query	False	Filter by svm.uuid
svm.name	string	query	False	Filter by svm.name
scope	string	query	False	Filter by scope
public_certificate	string	query	False	Filter by public_certificate • Introduced in: 9.8

Name	Type	In	Required	Description
serial_number	string	query	False	Filter by serial_number • maxLength: 40 • minLength: 1
expiry_time	string	query	False	Filter by expiry_time
hash_function	string	query	False	Filter by hash_function
authority_key_identifier	string	query	False	Filter by authority_key_identifier • Introduced in: 9.8
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. • Default value: 1 • Max value: 120 • Min value: 0

Name	Type	In	Required	Description
return_records	boolean	query	False	The default is true for GET calls. When set to false, only the number of records is returned. • Default value: 1
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"
    }
  },
  "num_records": 1,
  "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",
      "expiry_time": "string",
      "hash_function": "string",
      "intermediate_certificates": [
        "-----BEGIN CERTIFICATE-----
MIIBuzCCAWWgAwIBAgIIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQAwHDENMAsGA1UE
AxMEVEVTVDELMAkGA1UEBhMCVVMwHhcNMTgwNjA4MTgwOTAxWhcNMTkwNjA4MTgw
OTAxWjAcMQ0wCwYDVQQDEwRURVNUMQswCQYDVQQGEwJVUzBcMA0GCSqGSIb3DQEB
AQUAA0sAMEgCQQDaPvbqUJJFJ6NNTyK3Yb+ytSjJ9aa3yUmYTD9uMiP+6ycjxHWB
e8u9z6yCHsW03ync+dnhE5c5z8wuDAY0fv15AgMBAAGjgYowgYcwDAYDVR0TBAUw
AwEB/zALBgNVHQ8EBAMCAQYwHQYDVR0OBBYEFMJ7Ev/o/3+YNzYh5XNlqqjnw4zm
MEsGA1UdIwREMEKAFMJ7Ev/o/3+YNzYh5XNlqqjnw4zmoSCkHjAcMQ0wCwYDVQQD
EwRURVNUMQswCQYDVQQGEwJVU4IIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQADQQAv
DovYeyGNnknjGI+TVNX6nDbyzf7zUPqnri0KuvObEeybrbPW45sgsnT5dyeE/32U
9Yr6lklnkBtVBDTmLnrC -----END CERTIFICATE-----"
      ],
      "name": "cert1",
      "private_key": "-----BEGIN PRIVATE KEY-----
MIIBVAIBADANBgkqhkiG9w0BAQEFAASCAT4wggE6AgEAAkEAu1/a8f3G47cZ6pel
Hd3aONMNkGJ8vSCH5QjicuDm92VtVwkAACEjIoZSLYlJvPD+odL+lFzVQSmkneW7
VCGqYQIDAQABAkAcfNpg6GCQxoneLOghvlUrRotNZGvqpUOEAvHK3X7AJhz5SU4V
an36qvsAt5ghFMVM2iGvGaXbj0dAd+Jg64pxAiEA32Eh9mPtFSmZhTIUMeGcPmPk
qIYCEuP8a/ZLmI9s4TsCIQDWvLQuvjSVfwPhi0TFAb5wqAET8X5LBFqtGX5QlUep
EwIgFnqM02Gc4wtLoqa2d4qPkYu13+uUW9hLd4XSd6i/OS8CIQDT3elU+Rt+qIwW
```

```
u0cFrVvNYSV3HNzDfS9N/IoxTagfewIgPvXADe5c2EWbhCUkhN+ZCf38AKewK9TW
lQcDy4L+f14= -----END PRIVATE KEY-----",
      "public_certificate": "-----BEGIN CERTIFICATE-----
MIIBuzCCAWWgAwIBAgIIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQAwHDENMAsGA1UE
AxMEVEVTVDELMAkGA1UEBhMCVVMwHhcNMTgwNjA4MTgwOTAxWhcNMTkwNjA4MTgw
OTAxWjAcMQ0wCwYDVQQDEwRURVNUMQswCQYDVQQGEwJVUzBcMA0GCSqGSIb3DQEB
AQUAA0sAMEgCQQDaPvbqUJJFJ6NNTyK3Yb+ytSjJ9aa3yUmYTD9uMiP+6ycjxHWB
e8u9z6yCHsW03ync+dnhE5c5z8wuDAY0fv15AgMBAAGjgYowgYcwDAYDVR0TBAUw
AwEB/zALBgNVHQ8EBAMCAQYwHQYDVR0OBBYEFMJ7Ev/o/3+YNzYh5XNlqqjnw4zm
MEsGA1UdIwREMEKAFMJ7Ev/o/3+YNzYh5XNlqqjnw4zmoSCkHjAcMQ0wCwYDVQQD
EwRURVNUMQswCQYDVQQGEwJVU4IIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQADQQAv
DovYeyGNnknjGI+TVNX6nDbyzf7zUPqnri0KuvObEeybrbPW45sgsnT5dyeE/32U
9Yr6lklnkBtVBDTmLnrC -----END CERTIFICATE-----",
      "scope": "string",
      "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": "string",
      "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	Type of Certificate. The following types are supported: • 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 • x-nullable: true
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
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	The default is false. If set to true, the records are returned. • Default value:

Request Body

Name	Type	Description
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.
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: • 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 • x-nullable: true
uuid	string	Unique ID that identifies a certificate.

Example request


```
{
  "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",
  "expiry_time": "string",
  "hash_function": "string",
  "intermediate_certificates": [
    "-----BEGIN CERTIFICATE-----
MIIBuzCCAWWgAwIBAgIIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQAwHDENMAsGA1UE
AxMEVEVTVDELMAkGA1UEBhMCVVMwHhcNMTgwNjA4MTgwOTAxWhcNMTkwNjA4MTgw
OTAxWjAcMQ0wCwYDVQQDEwRURVNUMQswCQYDVQQGEwJVUzBcMA0GCSqGSIb3DQEB
AQUAA0sAMEgCQQDaPvbqUJJFJ6NNTyK3Yb+ytSjJ9aa3yUmYTD9uMiP+6ycjxHWB
e8u9z6yCHsW03ync+dnhE5c5z8wuDAY0fv15AgMBAAGjgYowgYcwDAYDVR0TBAUw
AwEB/zALBgNVHQ8EBAMCAQYwHQYDVR0OBBYEFMJ7Ev/o/3+YNzYh5XNlqqjnw4zm
MEsGA1UdIwREMEKAFMJ7Ev/o/3+YNzYh5XNlqqjnw4zmoSCkHjAcMQ0wCwYDVQQD
EwRURVNUMQswCQYDVQQGEwJVU4IIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQADQQAv
DovYeyGNnknjGI+TVNX6nDbyzf7zUPqnri0KuvObEeybrbPW45sgsnT5dyeE/32U
9Yr6lklnkBtVBDTmLnrC -----END CERTIFICATE-----"
  ],
  "name": "cert1",
  "private_key": "-----BEGIN PRIVATE KEY-----
MIIBVAIBADANBgkqhkiG9w0BAQEFAASCAT4wggE6AgEAAkEAu1/a8f3G47cZ6pel
Hd3aONMNkGJ8vSCH5QjicuDm92VtVwkAACEjIoZSLYlJvPD+odL+lFzVQSmkneW7
VCGqYQIDAQABAkAcfNpg6GCQxoneLOghvlUrRotNZGvqpUOEAvHK3X7AJhz5SU4V
an36qvsAt5ghFMVM2iGvGaXbj0dAd+Jg64pxAiEA32Eh9mPtFSmZhTIUMeGcPmPk
qIYCEuP8a/ZLmI9s4TsCIQDWvLQuvjSVfwPhi0TFAb5wqAET8X5LBFqtGX5QlUep
EwIgFnqM02Gc4wtLoqa2d4qPkYu13+uUW9hLd4XSd6i/OS8CIQDT3elU+Rt+qIwW
u0cFrVvNYSV3HNzDfS9N/IoxTagfewIgPvXADe5c2EWbhCUkhN+ZCf38AKewK9TW
lQcDy4L+f14= -----END PRIVATE KEY-----",
  "public_certificate": "-----BEGIN CERTIFICATE-----
MIIBuzCCAWWgAwIBAgIIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQAwHDENMAsGA1UE
AxMEVEVTVDELMAkGA1UEBhMCVVMwHhcNMTgwNjA4MTgwOTAxWhcNMTkwNjA4MTgw
OTAxWjAcMQ0wCwYDVQQDEwRURVNUMQswCQYDVQQGEwJVUzBcMA0GCSqGSIb3DQEB
AQUAA0sAMEgCQQDaPvbqUJJFJ6NNTyK3Yb+ytSjJ9aa3yUmYTD9uMiP+6ycjxHWB
e8u9z6yCHsW03ync+dnhE5c5z8wuDAY0fv15AgMBAAGjgYowgYcwDAYDVR0TBAUw
AwEB/zALBgNVHQ8EBAMCAQYwHQYDVR0OBBYEFMJ7Ev/o/3+YNzYh5XNlqqjnw4zm
MEsGA1UdIwREMEKAFMJ7Ev/o/3+YNzYh5XNlqqjnw4zmoSCkHjAcMQ0wCwYDVQQD
EwRURVNUMQswCQYDVQQGEwJVU4IIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQADQQAv
DovYeyGNnknjGI+TVNX6nDbyzf7zUPqnri0KuvObEeybrbPW45sgsnT5dyeE/32U
9Yr6lklnkBtVBDTmLnrC -----END CERTIFICATE-----",
  "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": {
    "name": "svm1",
    "uuid": "02c9e252-41be-11e9-81d5-00a0986138f7"
  },
  "type": "string",
  "uuid": "string"
}
```


Response

```
Status: 201, Created
```

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

Example response


```
{
  "num_records": 1,
  "records": [
    {
      "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",
      "expiry_time": "string",
      "hash_function": "string",
      "intermediate_certificates": [
        "-----BEGIN CERTIFICATE-----
MIIBuzCCAWWgAwIBAgIIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQAwHDENMAsGA1UE
AxMEVEVTVDELMAkGA1UEBhMCVVMwHhcNMTgwNjA4MTgwOTAxWhcNMTkwNjA4MTgw
OTAxWjAcMQ0wCwYDVQQDEwRURVNUMQswCQYDVQQGEwJVUzBcMA0GCSqGSIb3DQEB
AQUAA0sAMEgCQQDaPvbqUJJFJ6NNTyK3Yb+ytSjJ9aa3yUmYTD9uMiP+6ycjxHWB
e8u9z6yCHsW03ync+dnhE5c5z8wuDAY0fv15AgMBAAGjgYowgYcwDAYDVR0TBAUw
AwEB/zALBgNVHQ8EBAMCAQYwHQYDVR0OBBYEFMJ7Ev/o/3+YNzYh5XNlqqjnw4zm
MEsGA1UdIwREMEKAFMJ7Ev/o/3+YNzYh5XNlqqjnw4zmoSCkHjAcMQ0wCwYDVQQD
EwRURVNUMQswCQYDVQQGEwJVU4IIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQADQQAv
DovYeyGNnknjGI+TVNX6nDbyzf7zUPqnri0KuvObEeybrbPW45sgsnT5dyeE/32U
9Yr6lklnkBtVBDTmLnrC -----END CERTIFICATE-----"
      ],
      "name": "cert1",
      "private_key": "-----BEGIN PRIVATE KEY-----
MIIBVAIBADANBgkqhkiG9w0BAQEFAASCAT4wggE6AgEAAkEAu1/a8f3G47cZ6pel
Hd3aONMNkGJ8vSCH5QjicuDm92VtVwkAACEjIoZSLYlJvPD+odL+lFzVQSmkneW7
VCGqYQIDAQABAkAcfNpg6GCQxoneLOghvlUrRotNZGvqpUOEAvHK3X7AJhz5SU4V
an36qvsAt5ghFMVM2iGvGaXbj0dAd+Jg64pxAiEA32Eh9mPtFSmZhTIUMeGcPmPk
qIYCEuP8a/ZLmI9s4TsCIQDWvLQuvjSVfwPhi0TFAb5wqAET8X5LBFqtGX5QlUep
EwIgFnqM02Gc4wtLoqa2d4qPkYu13+uUW9hLd4XSd6i/OS8CIQDT3elU+Rt+qIwW
u0cFrVvNYSV3HNzDfS9N/IoxTagfewIgPvXADe5c2EWbhCUkhN+ZCf38AKewK9TW
lQcDy4L+f14= -----END PRIVATE KEY-----",
      "public_certificate": "-----BEGIN CERTIFICATE-----
MIIBuzCCAWWgAwIBAgIIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQAwHDENMAsGA1UE
AxMEVEVTVDELMAkGA1UEBhMCVVMwHhcNMTgwNjA4MTgwOTAxWhcNMTkwNjA4MTgw
OTAxWjAcMQ0wCwYDVQQDEwRURVNUMQswCQYDVQQGEwJVUzBcMA0GCSqGSIb3DQEB
AQUAA0sAMEgCQQDaPvbqUJJFJ6NNTyK3Yb+ytSjJ9aa3yUmYTD9uMiP+6ycjxHWB
e8u9z6yCHsW03ync+dnhE5c5z8wuDAY0fv15AgMBAAGjgYowgYcwDAYDVR0TBAUw
AwEB/zALBgNVHQ8EBAMCAQYwHQYDVR0OBBYEFMJ7Ev/o/3+YNzYh5XNlqqjnw4zm
MEsGA1UdIwREMEKAFMJ7Ev/o/3+YNzYh5XNlqqjnw4zmoSCkHjAcMQ0wCwYDVQQD
EwRURVNUMQswCQYDVQQGEwJVU4IIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQADQQAv
DovYeyGNnknjGI+TVNX6nDbyzf7zUPqnri0KuvObEeybrbPW45sgsnT5dyeE/32U
9Yr6lklnkBtVBDTmLnrC -----END CERTIFICATE-----",
```

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


Headers

Name	Description	Type
Location	Useful for tracking the resource location	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.

Error Code	Description
52363366	Unsupported key type.
52560123	Failed to read the key due to incorrect formatting.
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.

Definitions

See Definitions

href

Name	Type	Description
href	string	

_links

svm

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

security_certificate

Name	Type	Description
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.
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: • 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 • x-nullable: true
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.

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. • 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


```
{
  "expiry_time": "string",
  "hash_function": "string",
  "signing_request": "'-----BEGIN CERTIFICATE REQUEST-----
MIICYDCCAUgCAQAwGzEMMAoGA1UEAxMDQUJDMQswCQYDVQQGEwJVUzCCASIwDQYJ
KoZIhvcNAQEBBQADggEPADCCAQoCggEBAPF+82SlqT3Vyu3Jx4IAwHcO5EGwLOxy
zQ6KNjz71Fca0n1/A1CbCPyOsSupGVObvdWxX7xLVMJ2SXb7h43GCqYyX6FXJO4F
HOpmLvB+jxdeiW7SDbiZyLUlsvA+oRO/uNlcug773QZdKLjJD64erZZMRUNbUJB8
bARxAUi0FPvgTraSQ0UW5sRLiGKeAyKA4wekYe1VgjHRTBizFbD4dI3njfva/2Bl
jf+kkulgcLJTuJNtkgeimqMKyraYuleYcYk2K+C//0NuNOuPbDfTXCM7O61vik09
Szi8nLN7OXE9KoAA93U/BCpSfpl8XIb4cGnEr8hgVHOOtZSo+KZBFxMCAwEAAaAA
MA0GCSqGSIb3DQEBCwUAA4IBAQC2vFYpvgsFrm5GnPx8tOBD1xsTyYjbWJMD8hAF
lFrvF9Sw9QGCtDyacxkwgJhQx8l8JiIS5GOY6WWLBl9FMkLQNAhDL9xF3WF7vfYq
RKgrz3bd/Vg96fsRZNYIPLGmoEaqLOh3FOCGc2VbdsR9PwOn3fwthxkIRd6ds6/q
jc5cpSmVsCOgu+OKcpRXikYDbkWXfTZ1AhSfn6njBYFdZ9+PNAu/0JRQh5bX60nO
5heniTcAJLwUZP/CQ8nxHY0Wqy+1rAtM33d5cVmhUlBXQSIru/0ZkA/b9fK5Zv8E
ZMADYUoEvIG59Vxhyci8lzYf+Mxl8qBSF+ZdC4yWhzDqZtM9 -----END CERTIFICATE
REQUEST-----'"
}
```


Response

```
Status: 200, Ok
```

Name	Type	Description
public_certificate	string	CA signed public key Certificate

Example response


```
{
  "public_certificate": "string"
}
```


Error

```
Status: Default
```

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.
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.

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 `root_ca` 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 `root_ca` 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.

Name	Type	Description
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.
svm	svm	

Name	Type	Description
type	string	Type of Certificate. The following types are supported: • 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 • x-nullable: true
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",
  "expiry_time": "string",
  "hash_function": "string",
  "intermediate_certificates": [
    "-----BEGIN CERTIFICATE-----
MIIBuzCCAWWgAwIBAgIIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQAwHDENMAsGA1UE
AxMEVEVTVDELMAkGA1UEBhMCVVMwHhcNMTgwNjA4MTgwOTAxWhcNMTkwNjA4MTgw
OTAxWjAcMQ0wCwYDVQQDEwRURVNUMQswCQYDVQQGEwJVUzBcMA0GCSqGSIb3DQEB
AQUAA0sAMEgCQQDaPvbqUJJFJ6NNTyK3Yb+ytSjJ9aa3yUmYTD9uMiP+6ycjxHWB
e8u9z6yCHsW03ync+dnhE5c5z8wuDAY0fv15AgMBAAGjgYowgYcwDAYDVR0TBAUw
AwEB/zALBgNVHQ8EBAMCAQYwHQYDVR0OBBYEFMJ7Ev/o/3+YNzYh5XNlqqjnw4zm
MEsGA1UdIwREMEKAFMJ7Ev/o/3+YNzYh5XNlqqjnw4zmoSCkHjAcMQ0wCwYDVQQD
EwRURVNUMQswCQYDVQQGEwJVU4IIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQADQQAv
DovYeyGNnknjGI+TVNX6nDbyzf7zUPqnri0KuvObEeybrbPW45sgsnT5dyeE/32U
9Yr6lklnkBtVBDTmLnrC -----END CERTIFICATE-----"
  ],
  "name": "cert1",
  "private_key": "-----BEGIN PRIVATE KEY-----
MIIBVAIBADANBgkqhkiG9w0BAQEFAASCAT4wggE6AgEAAkEAu1/a8f3G47cZ6pel
Hd3aONMNkGJ8vSCH5QjicuDm92VtVwkAACEjIoZSLYlJvPD+odL+lFzVQSmkneW7
VCGqYQIDAQABAkAcfNpg6GCQxoneLOghvlUrRotNZGvqpUOEAvHK3X7AJhz5SU4V
an36qvsAt5ghFMVM2iGvGaXbj0dAd+Jg64pxAiEA32Eh9mPtFSmZhTIUMeGcPmPk
qIYCEuP8a/ZLmI9s4TsCIQDWvLQuvjSVfwPhi0TFAb5wqAET8X5LBFqtGX5QlUep
EwIgFnqM02Gc4wtLoqa2d4qPkYu13+uUW9hLd4XSd6i/OS8CIQDT3elU+Rt+qIwW
u0cFrVvNYSV3HNzDfS9N/IoxTagfewIgPvXADe5c2EWbhCUkhN+ZCf38AKewK9TW
lQcDy4L+f14= -----END PRIVATE KEY-----",
  "public_certificate": "-----BEGIN CERTIFICATE-----
MIIBuzCCAWWgAwIBAgIIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQAwHDENMAsGA1UE
AxMEVEVTVDELMAkGA1UEBhMCVVMwHhcNMTgwNjA4MTgwOTAxWhcNMTkwNjA4MTgw
OTAxWjAcMQ0wCwYDVQQDEwRURVNUMQswCQYDVQQGEwJVUzBcMA0GCSqGSIb3DQEB
AQUAA0sAMEgCQQDaPvbqUJJFJ6NNTyK3Yb+ytSjJ9aa3yUmYTD9uMiP+6ycjxHWB
e8u9z6yCHsW03ync+dnhE5c5z8wuDAY0fv15AgMBAAGjgYowgYcwDAYDVR0TBAUw
AwEB/zALBgNVHQ8EBAMCAQYwHQYDVR0OBBYEFMJ7Ev/o/3+YNzYh5XNlqqjnw4zm
MEsGA1UdIwREMEKAFMJ7Ev/o/3+YNzYh5XNlqqjnw4zmoSCkHjAcMQ0wCwYDVQQD
EwRURVNUMQswCQYDVQQGEwJVU4IIFTZBrqZwUUMwDQYJKoZIhvcNAQELBQADQQAv
```

```
DovYeyGNnknjGI+TVNX6nDbyzf7zUPqnri0KuvObEeybrbPW45sgsnT5dyeE/32U
9Yr6lklnkBtVBDTmLnrC -----END CERTIFICATE-----",
  "scope": "string",
  "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": "string",
  "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 © 2025 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.