Swaptacular SSL/TLS Certificates

Evgeni Pandurksi

2023-08-01

Overview

This document specifies the requirements for the particular kinds of certificates used in Swaptacular for peer authentication.

Every Swaptacular network node (accounting authority, debtors agent, or creditors agent node) runs its own trusted *certificate authority*, and issues a self-signed *root certificate* to itself. Each node's trusted certificate authority issues *server certificates* and *peer certificates*.

The certificates described in this document are Internet X.509 Public Key Infrastructure Certificates (RFC 5280). Therefore, here we deal only with the additional requirements specific to the particular kind of certificates used in Swaptacular to authenticate connections between peer nodes.

Note: The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

Root certificates

Every Swaptacular network node MUST issue **exactly one** public, self-signed root certificate to itself. The expiration date of root certificates SHOULD be set very far in the future (after 500 years for example).

Root certificates MUST include the following standard certificate extensions:

- *Basic Constraints* extension, marked as "critical", with its `cA` boolean field set to `true`, and its `pathLenConstraint` field set to at least `1`, or not set at all. This ensures that the certificate can participate in the chain of trust.
- *Key Usage* extension, marked as "critical" with its `keyCertSign` bit set to `true`. This ensures that the public key can be used for verifying signatures on other certificates.

- *Subject Key Identifier* extension, specifying the SHA-1 hash of the root certificate's public key as key identifier. This provides a means to quickly identify the set of certificates containing a particular public key.

When servers accept client SSL/TLS connections from Swaptacular peers, the node's root certificate SHOULD be configured to be **the only trusted root certificate**. Conversely, when initiating a client SSL/TLS connection to a peer node, the peer node's root certificate SHOULD be configured to be the only trusted root certificate, and the authenticated subject MUST be verified to match the expected subject (the expected peer).

Server certificates

Server certificates are used by Swaptacular nodes' servers to prove their identity before peer nodes' servers. Server certificates MUST be signed with the private key used for signing the node's root certificate. Swaptacular nodes may issue as many server certificates as they want (each one having a different private/public key pair), but the subject of each server certificate MUST be exactly the same as the subject of the node's root certificate. The expiration date of server certificates SHOULD be set in the relatively close future (after 1 year for example).

Server certificates MUST include the following standard certificate extensions:

- *Basic Constraints* extension, marked as "critical", with its `cA` boolean field set to `false`. This ensures that the certificate can only appear at the end of the chain of trust.
- *Key Usage* extension, marked as "critical" with its `digitalSignature` and `keyEncipherment` bits set to `true`. This ensures that the stated key usage is consistent with the purposes listed in the "Extended Key Usage" extension.
- *Extended Key Usage* extension, marked as "non-critical", listing `clientAuth` and `serverAuth` as allowed purposes. This ensures that the certificate can be used for client and server authentication.
- *Subject Key Identifier* extension, marked as "non-critical", specifying the SHA-1 hash of the server public key as key identifier. This provides a means to quickly identify the set of certificates containing a particular public key.
- *Authority Key Identifier* extension, marked as "non-critical", specifying the SHA-1 hash of the root certificate's public key as key identifier. This provides a means of identifying the public key corresponding to the private key used to sign the certificate.

Peer certificates

Peer certificates are issued to peers nodes, so that peers nodes' servers can present them, along with a proper server certificate, to prove their identity (forming a

verifiable chain of trust). Every Swaptacular node should issue and send a peer certificate to each one of its peers. Peer certificates MUST be signed with the private key used for signing the issuing node's root certificate. The expiration date of peer certificates SHOULD be set very far in the future (after 500 years for example).

For every peer certificate, the subject and the subject's public key MUST exactly match those of the corresponding peer's root certificate. Similarly, the issuer and the issuer's public key MUST exactly match those of the issuing node's root certificate.

Peer certificates MUST include the following standard certificate extensions:

- *Basic Constraints* extension, marked as "critical", with its `cA` boolean field set to `true`. This ensures that the certificate can participate in the chain of trust.
- *Key Usage* extension, marked as "critical" with its `keyCertSign` bit set to `true`. This ensures that the public key can be used for verifying signatures on other certificates.
- *Name Constraints* extension, marked as "critical", specifying in its `permittedSubtrees` field a restriction of the `directoryName` form, ensuring that all certificates down the trust chain will not change the subject's `O`, `OU`, and `serialNumber` distinguished name attributes.
- *Subject Key Identifier* extension, marked as "non-critical", specifying the SHA-1 hash of the peer's public key as key identifier. This provides a means to quickly identify the set of certificates containing a particular public key.
- *Authority Key Identifier* extension, marked as "non-critical", specifying the SHA-1 hash of the root certificate's public key as key identifier. This provides a means of identifying the public key corresponding to the private key used to sign the certificate.

Peer certificates issued by *accounting authority* nodes, in addition to the previously listed extensions, MUST include the *Netscape Certificate Comment* extension (OID value: 2.16.840.1.113730.1.13), marked as "non-critical". The content MUST match the following regular expression:

```
Subnet: [a-f0-9]{6,16}
```

The hexadecimal value after the "Subnet: " prefix, specifies the range of creditor/debtor IDs (depending on the peer's node type) which the peer node is responsible for:

- *For creditors agent peers*, the hexadecimal value MUST contain exactly 6 hexadecimal characters. For example, if "Subnet: 123abc" is given to a creditors agent node, the peer node will be responsible for managing all creditor IDs between `0x123abc0000000000` and `0x123abcffffffffff` inclusive.
- *For debtors agent peers*, the hexadecimal value MUST contain at least

8 hexadecimal characters. For example, if "Subnet: 1234abcd" is given to a debtors agent node, the peer node will be responsible for managing all debtor IDs between `0x1234abcd00000000` and `0x1234abcdffffffff` inclusive.

Subject's and issuer's distinguished names

The value of the `subject` and `issuer` fields in the above described certificates MUST be a *distinguished name* (DN), which contains the following *relative distinguished names* (RDNs):

1. An "Organization" attribute (abbreviated as "O") with the value "`Swaptacular Nodes Registry`". (Here and bellow, the quotation marks are not part of the value.)
2. An "Organizational Unit" attribute (abbreviated as "OU") with the value:
   - "`Accounting Authorities`" for accounting authority nodes;
   - "`Creditors Agents`" for creditors agent nodes;
   - "`Debtors Agents`" for debtors agent nodes.
3. A "Serial Number" attribute (abbreviated as "serialNumber") with a value consisting of:
   - **exactly 8 lowercase hexadecimal characters** for accounting authority nodes (see the "Accounting authority nodes' serial numbers" section);
   - **exactly 32 lowercase hexadecimal characters** for creditors agent and debtors agent nodes (see the "Creditors/debtors agent nodes' serial numbers" section).

Apart from the RDNs listed above, subject's and issuer's distinguished names SHOULD NOT contain other RDNs (like a "Country Name" attribute, or a "Common Name" attribute).

Accounting authority nodes' serial numbers

For *accounting authority* nodes, the subject's "Serial Number" attribute MUST specify the range of debtor IDs which the accounting authority node is responsible for.

For example, an accounting authority node with serial number `1234abcd` would be responsible for managing all debtor IDs between `0x1234abcd00000000` and `0x1234abcdffffffff` inclusive.

Note: Every accounting authority node SHOULD list its serial number, along with its public key fingerprint, in a publicly accessible centralized registry. This largely eliminates the possibility malicious nodes to "steal" other nodes' serial numbers (and thus use their range of debtor IDs).

Creditors/debtors agent nodes' serial numbers

For for *creditors agent* and *debtors agent* nodes, the subject's "Serial Number" attribute SHOULD hold the hexadecimal representation of the highest 128-bits, of the SHA-256 hash value, of the DER-encoded public key for the node's root certificate.

Here is an example how the `openssl` and the `hexdump` utilities can be used together, to generate the node's serial number from the file containing the node's public/private key pair:

```
$ openssl rsa -in "root-ca.key" -pubout | \
openssl pkey -pubin -outform DER | \
openssl dgst -sha256 -binary | \
hexdump -n 16 -ve '/1 "%02x"'

...
f857733abf94b4302b9c8889ae0677c7
```

Note: Generating nodes' serial numbers from nodes' public keys, eliminates the possibility malicious nodes to "steal" other nodes' serial numbers.