SUSE

Securing Communication with TLS Certificates

WHAT?
: TLS certificates are key elements when establishing secure network communication.

WHY?
: You want to learn how to generate and sign TLS certificates to establish secured network communication.

EFFORT
: One hour is enough to learn how to manage TLS certificates and create a certificate signed by a private CA for a trusted network environment.

GOAL
: You can generate, sign and manage TLS certificates, and include them in the system-wide certificate store.

REQUIREMENTS

- To perform selected tasks, you need `root` or **`sudo`** privileges.

Publication Date: 06 Mar 2025

Contents

1 Introduction 3

2 Issuing and installing TLS certificates 5

3 Creating a private CA 7

4 Creating a server private key 8

5 Creating a CSR 9

6 Signing a CSR 12

7 System-wide CA certificate store 13

8 Troubleshooting 15

9 Legal Notice 18

Glossary 19

A GNU Free Documentation License 21

1 Introduction

Two systems, such as client and server hosts, communicate by exchanging data over the network. To avoid exposing such communication and possibly sensitive data, you can configure the systems to encrypt the mutual communication using *TLS certificate*.

This topic introduces basic theory behind establishing encrypted communication.

1.1 How is secure communication established?

The following steps illustrate the process of establishing secure communication between the client and the server. The process is often referred to as a *TLS handshake*.

1. **Client “hello”.** The TLS handshake is started by the client sending a `ClientHello` message to the server. Such message includes information about the *TLS* protocol versions and cryptographic algorithms supported by the client, as well as a random value used for generating session keys.

2. **Server “hello”.** When the server receives the `ClientHello` message, it responds with a `ServerHello` message. Such a message includes the TLS protocol version selected by the server, the selected cryptographic algorithm, and a random value. The server also sends its digital certificate that includes the *public key*.

3. **Certificate verification.** After the client receives the server's certificate, the client verifies that it is signed by a trusted CA and that the server's domain name matches the one in the certificate.

4. **Key exchange.** After the server's certificate is successfully verified, the client generates a secret and encrypts it with the server's public key from the certificate. This encrypted secret is sent to the server in a `ClientKeyExchange` message.

5. **Session key generation.** Both the client and the server use the exchanged data—client and server random values and the generated secret—to generate a shared session key. This key will be used to encrypt and decrypt data transmitted during the TLS session. Both parties confirm that subsequent messages will be encrypted by sending a `ChangeCipherSpec` message.

6. **Finished.** Finally, both the client and the server send a `Finished` message to confirm that the TLS handshake is complete. These messages contain a hash of all preceding TLS handshake messages, encrypted with the shared session key. If the hashes match on both sides, the TLS handshake is successful and the secure TLS connection is established.

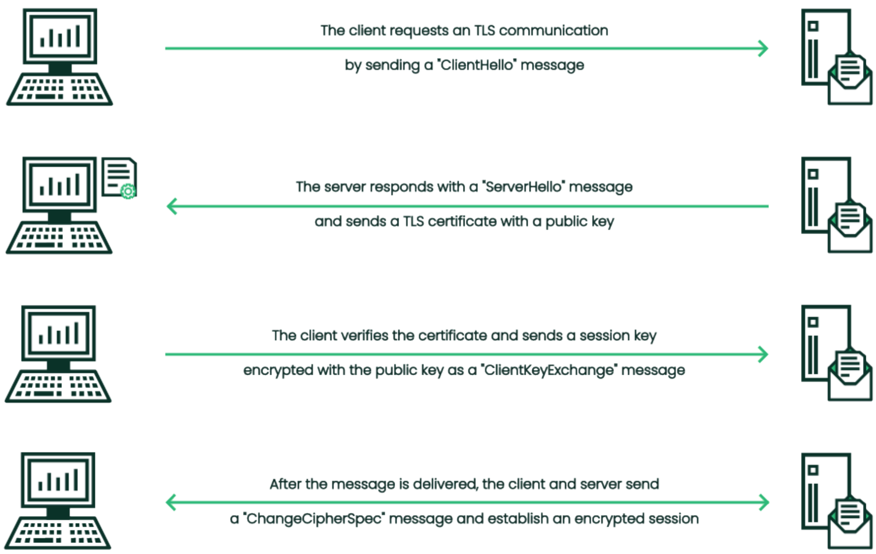

FIGURE 1: TLS HANDSHAKE PROCEDURE

1.2 Benefits of using secure communication

Several key benefits of secure communication are:

- **Confidentiality:** Secure communication ensures that sensitive information remains confidential and cannot be accessed by unauthorized parties. This is particularly important for protecting personal data and financial information.
- **Integrity:** Secure communication guarantees the integrity of transmitted data and ensures that it remains unchanged during transmission. This helps prevent unauthorized modification of data and maintain its accuracy and reliability.
- **Authentication:** Secure communication allows parties to authenticate each other's identities, verifying that they are who they claim to be. This helps prevent impersonation and ensures that communication is conducted securely between trusted parties.
- **Compliance:** Secure communication helps organizations follow regulatory requirements and industry standards related to data protection and privacy. Compliance with regulations such as GDPR often requires the implementation of secure communication protocols and practices.

2 Issuing and installing TLS certificates

The following procedures outline the TLS certificate issuance and installation process using both trusted and private CAs.

2.1 Using a trusted CA

1. **Generate a private key.** A *private key* is required on the server—such as a Web server—that will use the certificate. The key is kept secure and used to sign messages and decrypt data.
2. **Create a certificate signing request (CSR).** Generate a *CSR* using the private key. The CSR contains information about the entity requesting the certificate and the public key that will be included in the certificate.

3. **Submit the CSR to a CA signing.** The *CA* verifies the information in the CSR and issues a signed certificate if the request is approved. This step may involve validation of the domain ownership or organization identity, depending on the type of certificate requested.
4. **Receive the signed certificate.** After the CA approves the CSR, it issues a signed certificate. The certificate includes the digital signature of the CA, the public key of the server, and additional information about the server, such as the domain name or organization.
5. **Install the certificate on the server.** Install the signed certificate on the server that will use it. This typically involves configuring the server software—such as the Apache Web server—to use the certificate for securing communication.
6. **Configure the server and client for secure communication.** After the certificate is installed on the server, configure the server and client software to use secure communication protocols, for example, HTTPS that uses the certificate for encryption and authentication.
7. **Regular maintenance and renewal.** Regularly monitor the validity of the certificate and renew it before it expires. Keep the private key secure and ensure that certificates are properly configured and updated as needed.

2.2 Using a private CA

1. **Create a private certificate authority (CA).** A private CA is only needed to sign the server certificate yourself and *not* have it signed by a trusted CA.
2. **Generate a private key.** A private key is required on the server—such as a Web server—that will use the certificate. The key is kept secure and used to sign messages and decrypt data.
3. **Create a certificate signing request (CSR).** Generate a CSR using the private key. The CSR contains information about the entity requesting the certificate and the public key that will be included in the certificate.
4. **Sign the CSR with your private CA.** If you intend to use the TLS certificate for testing or internal purposes, you can sign the CSR yourself by your private CA instead of a publicly trusted CA.
5. **Install the certificate on the server.** Install the signed certificate on the server that will use it. This typically involves configuring the server software—such as the Apache Web server—to use the certificate for securing communication.

6. **Configure the server and client for secure communication.** After the certificate is installed on the server, configure the server and client software to use secure communication protocols, for example, HTTPS that uses the certificate for encryption and authentication.
7. **Regular maintenance and renewal.** Regularly monitor the validity of the certificate and renew it before it expires. Keep the private key secure and ensure that certificates are properly configured and updated as needed.

3 Creating a private CA

If you do not plan to get the server certificate signed by a trusted CA, create your private CA. Such an approach is fine for testing and internal scenarios, but not for production network applications.

REQUIREMENTS

- To perform selected tasks, you need `root` or **`sudo`** privileges.

1. Generate a private key and a certificate for your private CA. The following command creates a new 2048 bit-long RSA key using the SHA-256 hash algorithm. During its creation, you will be asked for an encryption passphrase. Remember the passphrase because you will need it when signing the server CSR with your private CA in *Section 6, “Signing a CSR”*.

```
> openssl req -x509 -sha256 -days 3650 -newkey rsa:2048 \
-subj "/CN=myRootCA/C=CZ/L=Prague" -keyout myRootCA.key -out myRootCA.crt
```

   Replace the `-subj` value with your suitable string.

2. Secure the private key by restricting its permissions and ownership so that only the `root` can read it.

```
> sudo chown root:root myRootCA.key
> sudo chmod 600 myRootCA.key
```

3. *(Optional)* Verify the validity of the CA private key.

```
> openssl pkey -check -in myRootCA.key
Enter pass phrase for myRootCA.key:
Key is valid
-----BEGIN PRIVATE KEY-----
MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEAAoIBAQCq/oAhZ0VGSsGb
```

```
[...]
8q3vKA+KRtxhMgW1f50U2qo=
-----END PRIVATE KEY-----
```

4. *(Optional)* Verify your private CA by displaying its basic data. It must include the `CA:TRUE` flag.

```
> openssl x509 -in myRootCA.crt -text -noout
Certificate:
[...]
    Signature Algorithm: sha256WithRSAEncryption
    Issuer: CN = myRootCA, C = CZ, L = Prague
    Validity
        Not Before: Apr 23 14:41:53 2024 GMT
        Not After : Apr 17 14:41:53 2029 GMT
    Subject: CN = myRootCA, C = CZ, L = Prague
    X509v3 extensions:
[...]
        X509v3 Basic Constraints: critical
          CA:TRUE
[...]
```

NEXT STEPS

- To trust the server certificate, client machines need to know the private CA certificate with which the server's private certificate was signed. Refer to *Section 7.3, "Adding new CA certificates"* for details about storing CA certificates in a system-wide certificate store on the client.
- Create a server private TLS key as described in *Section 4, "Creating a server private key"*.

4 Creating a server private key

A server needs a private TLS key to encrypt data required by a client. The following procedure describes how to generate it.

REQUIREMENTS

- To perform selected tasks, you need `root` or **`sudo`** privileges.

1. Generate a private TLS key. The following example generates an unencrypted 256-bit *ECDSA* key.

> **Tip**
>
> You can verify if ECDSA keys are supported by listing its available elliptic curves with the **`openssl ecparam -list_curves`** command.

```
> openssl ecparam -name prime256v1 -genkey -out server.key
```

The speed of generating the key depends on the hardware, the selected encryption algorithm, and the length of the key.

2. Move the generated private key to a safe location on the server and secure it by restricting its permissions and ownership so that only the `root` can read it.

```
> sudo chown root:root server.key
> sudo chmod 600 server.key
```

3. *(Optional)* Verify the consistency of the private key.

```
> sudo openssl pkey -check -in server.key
Key is valid
-----BEGIN PRIVATE KEY-----
MIGHAgEAMBMGByqGSM49AgEGCCqGSM49AwEHBG0wawIBAQQgUnW551Qru4+pbWqu
JRXQVFi45N1j+qmx7dEnr+8eox+hRANCAAR+rRLHBGjq2H2j0q09efVt99JB/R7h
QkDLjVMa9jemVH1g3YiVIEAHyCVjms2rC06lkU1S+z8WsRjh6A/ev8//
-----END PRIVATE KEY-----
```

NEXT STEPS

- Create a certificate signing request (CSR) so that your preferred CA can sign and validate it. Find more details in *Section 5, "Creating a CSR"*.

5 Creating a CSR

Establishing TLS encrypted communication requires a valid TLS certificate signed by a CA. To obtain such a certificate, you need to create a certificate signing request (CSR).

REQUIREMENTS

- You have previously created a private TLS key as described in *Section 4, "Creating a server private key"*.

1. *(Optional)* Prepare a configuration template that simplifies the process of creating your CSR, for example:

```
> cat example_csr.cnf
[req]
prompt = no
distinguished_name = dn
req_extensions     = re

[dn]
countryName             = CZ
localityName            = Prague
organizationName        = Example organization
commonName              = www.example.org
emailAddress            = wilber@example.org

[re]
extendedKeyUsage = serverAuth,clientAuth

[alt_names]
DNS.1 = example.org
DNS.2 = www.example.org
IP.1 = 10.0.0.5
IP.2 = 10.0.0.6
```

Important: Exact request matches

The `[alt_names]` section can list domain names and IP addresses that must match the URL that the client will request. In the above example, while the following requests will be accepted:

```
https://example.org/
https://www.example.org/
https://10.0.0.5/
https://10.0.0.6/
```

the request for `https://www2.example.org/` will be refused.

2. a. If you previously prepared a CSR template, create the CSR using the template and the private key.

```
> openssl req -new -key server.key \
 -config example_csr.cnf \
 -out server.csr
```

b. If you did not prepare the CSR template, omit the `-config` option from the command above.

```
> openssl req -new -key server.key -out server.csr
```

You will be prompted for additional information about the certificate. When asked for a `challenge password` and an `optional company name`, leave it blank.

Important: Common Name

The `Common Name (CN)` value must be the fully qualified host name of the server host.

```
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:CZ
State or Province Name (full name) [Some-State]: Czech Republic
Locality Name (eg, city) []:Prague
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Example
 organization
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:www.example.org
Email Address []:wilber@example.org

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

NEXT STEPS

- Submit the CSR to a CA of your choice. If you want to use the certificate inside a trusted network, you can sign the certificate with your private CA as described in *Section 6, "Signing a CSR"*.

6 Signing a CSR

Establishing TLS encrypted communication requires that a certificate authority (CA) signs the certificate signing request (CSR). For testing or internal purposes, you can sign the CSR yourself by a private CA.

REQUIREMENTS

- You created a private TLS key as described in *Section 4, "Creating a server private key"*.
- You created a CSR file as described in *Section 5, "Creating a CSR"*.

1. *(Optional)* Prepare an optional configuration file that specifies additional server and client extensions, for example:

```
> cat example_ssl_ext.cnf
[ server-cert ]
keyUsage = digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth
basicConstraints=CA:FALSE

[ client-cert ]
keyUsage = digitalSignature
extendedKeyUsage = clientAuth
basicConstraints=CA:FALSE
```

2. Sign the CSR with your private CA, optionally using the extension configuration file. You will be asked for your private CA passphrase that you entered per *Section 3, "Creating a private CA"*.

```
openssl x509 -req -days 730 -in server.csr -out server.crt \
-CA myRootCA.crt -CAkey myRootCA.key -CAcreateserial \
-extfile example_ssl_ext.cnf -extensions server-cert
```

The certificate remains valid for two years. Adjust the value to your needs. After this period, you must create a new CSR and have a CA sign the certificate.
The `server.crt` file will be created that includes the signed certificate.

3. *(Optional)* Verify that the data included in the resulting certificate matches your requirements.

```
> openssl x509 -text -noout -in server.crt
[...]
```

```
Issuer: C = CZ, ST = Some-State, L = Prague, O = Example organization, CN =
 www.example.org, emailAddress = wilber@example.org
Validity
    Not Before: Apr 15 09:18:11 2024 GMT
    Not After : Apr 15 09:18:11 2026 GMT
Subject: C = CZ, ST = Some-State, L = Prague, O = Example organization, CN =
 www.example.org, emailAddress = wilber@example.org
Subject Public Key Info:
    Public Key Algorithm: id-ecPublicKey[...]
X509v3 extensions:
        X509v3 Key Usage:
            Digital Signature, Key Encipherment
        X509v3 Extended Key Usage:
            TLS Web Server Authentication
[...]
```

NEXT STEPS

- Install the certificate on the server that will use it for providing encrypted content.

7 System-wide CA certificate store

A shared system-wide CA store is a centralized repository for storing trusted root certificates and user-specific certificates on a system. This store is used by software applications and components within the operating system to establish secure connections, validate the authenticity of TLS certificates presented by servers, and verify the identity of individuals or entities. By default, the store contains the Mozilla CA certificate list included in the `ca-certificates-mozilla` package. You can either update this list or select another certificate list.

Tip: OpenSSL vs NSS certificate store

By default, there are two different CA certificate stores in SLES: the OpenSSL-based store and the NSS (Network Security System)-based store. Many GUI-based Web browsers—such as Firefox—use the NSS certificate store. To avoid installing CA certificates in both certificate stores, install the plug-in package `p11-kit-nss-trust` that makes the NSS store look up certificates in the OpenSSL store automatically.

7.1 Where is the CA certificate store on the file system?

In SLES, the shared system-wide certificate store is located in the following directories:

/usr/share/pki/trust/anchors
: CA certificates trust anchors provided by the system.

/usr/share/pki/trust/blacklist
: Distrusted CA certificates provided by the system.

/etc/pki/trust/anchors
: CA certificates trust anchors provided by the system administrators.

/etc/pki/trust/blacklist
: Distrusted CA certificates provided by the system administrators.

7.2 Benefits of using a system-wide CA certificate store

Some of the key benefits of using a shared certificate store are:

- **Security:** Centralizing trusted certificates helps ensure that all applications and services use a consistent set of trusted certificates to verify the authenticity of TLS connections.
- **Simplified management:** Instead of each application or service maintaining its own list of trusted certificates, they can rely on the system-wide certificate store.
- **Ease of update:** System administrators can update the trusted certificates in the system-wide store as needed, either manually or through automated mechanisms such as operating system updates. This ensures that systems remain up to date with the latest trusted certificates and security standards.

7.3 Adding new CA certificates

To let applications know about a new private CA certificate on the system, add it to a system-wide certificate store on the client and use the **`update-ca-certificates`** command to update the store.

> REQUIREMENTS
>
> - The `ca-certificate` package is installed on the system.

- You created a CA certificate as described in *Section 3, "Creating a private CA"*.
- To perform selected tasks, you need `root` or **`sudo`** privileges.

1. Copy the certificate file to the `/etc/pki/trust/anchors` directory.

```
> sudo cp myRootCA.crt /etc/pki/trust/anchors/
```

2. Update the system-wide certificate store.

```
> sudo update-ca-certificates
```

FOR MORE INFORMATION

- The **`man 8 update-ca-certificates`** manual page.

8 Troubleshooting

This topic describes how to examine TLS certificates.

8.1 How do I examine a TLS certificate?

To review a server certificate that is available locally as a file, use the following command:

```
> openssl x509 -in server.crt -noout -text
Certificate:
  Data:
      Version: 3 (0x2)
      Serial Number:
          6a:43:f5:55:ef:03:ee:0e:23:c8:99:82:71:39:f2:22:1a:da:53:db
      Signature Algorithm: ecdsa-with-SHA256
      Issuer: 1 C = CZ, ST = Some-State, L = Prague, O = Example organization, CN =
 www.example.org, emailAddress = wilber@example.org
      Validity 2
          Not Before: Apr 15 09:30:31 2024 GMT
          Not After : May 15 09:30:31 2024 GMT
      Subject: 3 C = CZ, ST = Some-State, L = Prague, O = Example organization, CN =
 www.example.org, emailAddress = wilber@example.org
      Subject Public Key Info:
          Public Key Algorithm: id-ecPublicKey
              Public-Key: (256 bit)
              pub:
                  04:7e:ad:12:c7:04:68:ea:d8:7d:a3:3a:ad:3d:79:
[...]
```

```
            ASN1 OID: prime256v1
            NIST CURVE: P-256
      X509v3 extensions:
          X509v3 Subject Alternative Name: 4
            DNS:*.example.org, DNS:example.org
          X509v3 Key Usage: 5
              Digital Signature, Key Encipherment
          X509v3 Extended Key Usage:
              TLS Web Server Authentication
  Signature Algorithm: ecdsa-with-SHA256
       30:46:02:21:00:a8:4f:7d:16:f2:93:01:b9:3d:31:e3:c6:6c:
[...]
```

1. Describes who issued the certificate. It shows only the parent-level issuer, not the top-level issuer from the certificate chain.
2. Displays certificate validity time.
3. Describes the server that uses the certificate. The most important attribute is `CN` (Common Name), as it contains the fully qualified domain name of the server that can use this certificate. `CN` can contain wildcards, for example, `*.example.org`. In this case, the certificate can be used by multiple servers on the same domain `example.org`.
4. When used, it overrides the subject's `CN` specification. It can contain `DNS:` entries for domain names and `IP:` entries for IP addresses.
5. Specifies purposes for which the certificate can be used.

8.2 How do I download server TLS certificate files?

The following command downloads just the server certificate from the example server *suse.com*. Replace it with the domain name of the server you want to examine.

```
> echo | openssl s_client -showcerts -connect suse.com:443 2>/dev/null \
| openssl x509 -out 1.pem
```

Most public servers use several subordinate CAs in the certificate chain. To download the whole certificate chain into individual files, use the following command:

```
> echo | openssl s_client -showcerts -connect suse.com:443 2>/dev/null \
| awk -v RS="-----BEGIN CERTIFICATE-----" \
'NR>1{sub(/-----END CERTIFICATE-----.*/,"-----END CERTIFICATE-----"); \
print RS$0>NR-1".pem"}'
```

The command writes all certificate files named increasingly `1.pem`, `2.pem` and so on, up to the last certificate in the chain.

8.3 How do I verify the whole TLS certificate chain?

The client host normally has a collection of root CAs and verifies the server certificates against them. But the certificate that the server provides can be a subordinate CA. To trust the server's certificate, the client needs to verify its complete *TLS certificate chain*. The server sends all certificates as part of its `Server Hello` message and you can download them as described in *Section 8.2, "How do I download server TLS certificate files?"*.

The **`openssl`** command uses the `-subject_hash` and `-issuer_hash` options to retrieve the subject and issuer hashes from a certificate, for example:

```
> openssl x509 -in 1.pem -noout -issuer_hash
2401d14f
```

The client compares the issuer hash `2401d14f` with the issuer hashes of its collection of root CAs. In this case, no root CA has an issuer hash of `2401d14f`. As the client cannot trust the server yet, it has to verify the other certificates from the chain as well. Notice that the next certificate in the chain has the `-subject_hash` identical to the `-issuer_hash` from the server certificate.

```
> openssl x509 -in 2.pem -noout -subject_hash
2401d14f
```

Display the issuer hash of this certificate.

```
> openssl x509 -in 2.pem -noout -issuer_hash
ce5e74ef
```

Again, this issuer hash cannot be found among the client's root CAs. Continue with checking the issuer and subject hashes of the next certificates in the chain until you reach the last one. The results are summarized in the following table:

TLS certificate	Subject hash	Issuer hash
1.pem (suse.com)	4309e692	2401d14f
2.pem (first subCA)	2401d14f	ce5e74ef
3.pem (second subCA)	ce5e74ef	09789157
4.pem (third subCA)	09789157	f387163d
????????	f387163d	????????

FIGURE 2: SUBJECT AND ISSUER HASHES OF A CERTIFICATE CHAIN

The last issuer hash `f387163d` finally has a matching CA:

```
> ls -l /etc/ssl/certs/f387163d*
lrwxrwxrwx 1 root root 24 Oct 4 09:55 /etc/ssl/certs/f387163d.0 ->
 Starfield_Class_2_CA.pem
```

If the discovered issuer certificate is valid, you can trust the server as much as you trust the root CAs in your client certificate store.

```
> openssl x509 -in /etc/ssl/certs/Starfield_Class_2_CA.pem -noout -dates
notBefore=Jun 29 17:39:16 2004 GMT
notAfter=Jun 29 17:39:16 2034 GMT
```

To confirm that the certificate is a root CA, check its subject and issuer hashes. Because root CA certificates are self-signed, they have both hashes identical.

```
> openssl x509 -in /etc/ssl/certs/Starfield_Class_2_CA.pem -noout -issuer_hash -
subject_hash
f387163d
f387163d
```

The complete certificate chain then looks as follows:

TLS certificate	Subject hash	Issuer hash
1.pem (suse.com)	4309e692	2401d14f
2.pem (first subCA)	2401d14f	ce5e74ef
3.pem (second subCA)	ce5e74ef	09789157
4.pem (third subCA)	09789157	f387163d
Starfield_Class_2_CZ (rootCA)	f387163d	f387163d

FIGURE 3: SUBJECT AND ISSUER HASHES OF A CERTIFICATE CHAIN

9 Legal Notice

Glossary

CA

A *certificate authority* (CA) responsible for issuing digital certificates that authenticate the identity of individuals, organizations or servers on the Internet. It can be either a publicly trusted CA, such as Let's Encrypt (https://letsencrypt.org/), or a private CA in which case you sign the certificate yourself.

CSR

To have a certificate signed by a CA, you need to generate a public key and send it to the CA for signing. This process is called a *certificate signing request* (CSR).

DH key exchange

Diffie–Hellman key exchange (DH) is a mathematical method of securely exchanging cryptographic keys over a public channel and was one of the first public-key protocols. It was named after cryptologists Whitfield Diffie and Martin Hellman. DH is one of the earliest practical examples of public key exchange based on a private key and a corresponding public key.

ECDSA

Elliptic curve digital signature algorithm (ECDSA) is a cryptographic algorithm used to generate digital signatures based on the mathematics of elliptic curves. It is widely used for digital signature generation and verification in cryptographic protocols and applications, including TLS for secure communication over the Internet.

end-entity certificate

TLS certificates issued to individual entities such as Web sites, servers or clients. These certificates are signed by subordinate CAs and are used to authenticate and encrypt communication between clients and servers.

Openssl

An open-source software library that provides cryptographic functions and utilities to secure communication. It is used in SLES for implementing secure protocols, such as TLS.

PEM

A privacy enhanced mail (PEM) file is a commonly used file format for storing cryptographic objects such as certificates, private keys and certificate signing requests.

PFS

Perfect forward secrecy (PFS) is a feature of specific key-agreement protocols assuring that session keys are not compromised even if long-term secrets—such as the private key of the server—used in the session key exchange are compromised.

PKI

The public key infrastructure (PKI) is a set of policies, processes, software and hardware used to create, manage, distribute, use, store and revoke digital certificates. PKI enables secure communication and authentication over insecure networks, such as the Internet.

private key

A counterpart to the public key in asymmetric cryptography. It is kept secret and known only to the owner of the key pair. When a client connects to a server over TLS, the server sends its digital certificate with a public key. The client uses it to encrypt data, ensuring that only the server's private key can decrypt it.

public key

A part of a TLS certificate which acts as a digital identity for the server. When a client connects to the server over TLS, it requests the server's certificate, which contains the public key. The public key is freely distributable and is used for data encryption.

root CA

A trusted entity at the top of the certificate chain. It signs subordinate CAs and end-entity CAs. Root certificates are preinstalled in SLES to establish trust.

SSL

Secure socket layer (SSL) was the predecessor of the TLS protocol. SSL version 3.0 was replaced by TLS 1.0 in 1999 to address existing vulnerabilities.

subordinate CA

CAs that live between the root and end-entity certificates and are used to sign other certificates, including end-entity certificates. Their main purpose is to define and authorize the types of certificates that can be requested from the root CA. For example, there can be different subordinate CAs for different locations, or encryption key types.

TLS

Transport layer security (TLS) is a protocol that provides secure communication between client-server applications. TLS uses asymmetric cryptography with a pair of private and public keys. TLS is the successor to the SSL (Secure Sockets Layer) protocol.

TLS certificate

A digital X.509 certificate that helps to secure communication within a client-server system. Using TLS certificate, the client can authenticate the identity of the server and encrypt their mutual communication. The certificate usually includes information about its issuer (CA), identity of its holder, the associated public key, digital signature, and its validation period.

TLS certificate chain

A series of certificates used to establish the authenticity and trustworthiness of a particular certificate. A certificate chain can consist of the following certificate types: *root CA*, *subordinate CA* and *end-entity certificate*.

TLS handshake

A series of messages exchanged between the client and the server to establish a secure connection. The process involves negotiating cryptographic parameters, authenticating identities, and agreeing on encryption keys before secure communication can begin.

Trust anchor

Trust anchor is a root certificate that is inherently trusted by a system or application. This root certificate is used to verify the signatures of other certificates in the certificate chain.

X.509

A standard that defines the format of public key certificates.

A GNU Free Documentation License

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See https://www.gnu.org/copyleft/ ↗.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

```
Copyright (c) YEAR YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.2
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled “GNU
Free Documentation License”.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with...Texts.” line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.