



如何快速生成自签名 SSL 证书

2021/10/24 · 7 分钟阅读时长

本机或团队开发环境、公司内部系统等经常用到自签名 SSL/TLS 证书。如何方便快捷地生成自签名证书？本文介绍几个好用的工具及相关操作的快速参考等。本篇是“自签名证书与安全”系列之一，记录个人最佳实践。

SSL/TLS 协议应用于传输安全¹，需要结合数字证书。条件允许的情况下，应该选择使用**可信证书**，即由权威数字证书认证中心（CA）颁发的有效证书。对于一般应用，免费的证书服务也很方便申请与使用，如 [Let's Encrypt](#) 和 [ZeroSSL](#) 等。

与可信证书相对应的，是**自签名证书**，即由自己（非权威）作为 CA 然后颁发的证书。由于根证书没有内置到各操作系统及应用环境中，自签名证书默认是不被客户端信任的。**如果管理得当，使用自签名证书也能达到与可信证书相近的安全性，并且不会增加太多工作量。**

什么时候用自签名的证书？

通常内网服务、内网或本地开发与测试环境等某些场景，有时只能使用自签名证书。可能的原因有：

- **本机**开发或测试基于 https 协议的 Web 服务时，需要为 `localhost 127.0.0.1` 生成自签名 SSL 证书。权威 CA 不支持颁发此类证书。
- 团队内部开发测试环境，条件受限不能或不方便使用可信证书。例如：
 - 因为安全权限或内网限制，所以无法申请免费证书（测试服务不支持外网访问；开发人员没有测试域名的 DNS 解析权限所以不能随意修改 TXT 记录；提运维工单太麻烦流程太长等等）。
 - 临时测试，没法添加新 DNS 解析甚至干脆没有域名，想用 `example.com` 之类的域名做 hosts 解析临时用来测试。
- 中小公司的内网办公与生产系统，需要全面启用 https 以增强安全性、防止窃听与中间人攻击。同时因内外网隔离、成本、灵活性等各种原因不便申请使用可信证书。

生成什么样的自签名证书？

首先，需要创建一个自签名根 CA。同一组织环境下应该只创建一个，后面具体服务的 SSL 证书或者中级 CA 证书都只从该根 CA 颁发。这一方面是考虑安全性，可以集中控制颁发和撤销证书的权限；另一方面考虑易于管理，所有客户端只用安装信任一次自签名的 CA 证书，后续颁发的证书都自动信任。

其次，根据前文所列的场景，示例操作中支持的证书“通用名称（Common Name, CN）”或“DNS 名称”会包含以下三种：

1. 本机主机名和 IP：localhost 127.0.0.1
2. 测试域名和 IP：example.com test.example.com 192.168.8.8
3. 通配符测试域名：*.example.com，通配符证书是比较省事的做法，例如用一个反向代理对多个服务同时启用 https。

最后，有关加密或哈希算法的选择，参考了 Mozilla 的 [Security/Server Side TLS](#) 的推荐配置。CA 使用 prime256v1，普通证书使用 RSA:2048

关于文件格式和扩展名的简单说明：

- .pem ——Base64 编码的 DER 格式，多数时候都是这种格式。
- .cer/.crt ——通常是 Base64 编码的 DER 格式，极少数情况下可能是二进制的 DER 格式。
- .der ——DER格式，也可能是 Base64 编码过的（pem）
- .key ——实际是 pem，只是为了区别该文件是私钥所以这样命名
- .pfx/.p12 ——公私钥打包成单个文件，通常是 PKCS#12 格式，该格式标准的前身是 PFX，所以沿用了这个扩展名

选择合适的工具

对比与选择

以下是个人常用的几种与证书和签名有关的工具，以及各自的优劣：

- [openssl](#) 命令行，不建议
 - 功能全面且多数环境中已经内置，所以不需要额外下载安装。
 - 仅对生成 SSL 证书而言，操作参数较复杂，尤其是需要生成 CA 和中间证书的时候。
 - Google 或百度结果中的操作指引，可能有坑。例如 [macOS 下 openssl 默认配置不包含 v3 ca](#)，导致生成的根证书在一些环境下会报错等。
- [mkcert](#)，推荐个人使用
 - 可以生成 localhost 与 127.0.0.1 之类本地地址的证书。
 - 命令行参数支持自动安装生成的自签名根证书到操作系统等常用环境，不会弹出证书警告。
 - 不建议与其他人共享生成的根证书，因为谁都能用它来签发新证书，存在安全隐患。所以最好是个人使用。
- [step-ca](#)，推荐团队使用

- 既支持命令行使用，也支持作为私有 CA 服务器运行
- 作为 CA 服务时支持**开启 ACME 协议**，可以把它当作 Let's Encrypt 的内网替代，尤其是研发没有域名控制权时，用 HOSTS 解析也能进行开发测试。
- 支持吊销证书等功能
- 更多功能如 REST API 等可以从官方网站了解

openssl 快速参考

搜索“生成自签名证书”，结果前列通常都是使用 openssl。openssl 功能强大但比较复杂。下面是经过测试的生成自签名根证书，然后用自签名根证书颁发其它证书的操作示例。示例已经尽可能的减少操作步骤。



本机个人测试最简操作

只是个人在本机测试的话，可以如下直接生成一个证书，既是根证书又是服务实体证书。**不建议**在需要生成多个证书时使用，因为每一个证书都要安装到根证书存储中，混乱且不安全。

```
openssl req -x509 -newkey rsa:2048 -sha256 -days 3650 -nodes -  
keyout localhost.key -out localhost.crt -subj  
"/C=CN/CN=localhost" -addext  
"subjectAltName=DNS:localhost,IP:127.0.0.1,IP:::1"
```

注意 addext 参数需要 openssl 1.1.1 版本才支持

分步操作

以下操作分别在 CentOS 7 (openssl 1.0.2) 和 CentOS 8 (openssl 1.1.1) 版本下测试，并且某些命令必须使用 bash，其它版本和运行环境下可能出错。



openssl 命令的执行不仅与版本有关，也与配置文件紧密相关。所以同样的命令行，在不同系统环境下运行，生成的证书实际上可能有区别。默认配置文件常见路径包括 /etc/ssl/openssl.cnf 和 /etc/pki/tls/openssl.cnf

1. 创建根证书以及私钥²:

```
# 使用默认配置文件配置，创建的证书默认支持作为 CA  
openssl req -x509 -sha256 -days 3650 \  
-newkey ec -pkeyopt ec_paramgen_curve:prime256v1 \  
-keyout root_ca.key -out root_ca.crt \  

```

```
-subj "/C=CN/O=Example Ltd./OU=Root CA/CN=Example Root CA"
```

```
# 查看证书信息，确认包含 CA:TRUE
```

```
openssl x509 -text -noout -in root_ca.crt
```

- 说明:

- `-days 3650` 中的 3650 为证书有效天数，可修改
- `-newkey ec -pkeyout ec_paramgen_curve:prime256v1` 可改为使用 RSA 算法: `-newkey rsa:4096`
- `-keyout root_ca.key -out root_ca.crt` 分别指定了私钥和公钥的保存文件名，可修改
- `-subj` 后跟的是 `/C` 是“国家/地区”，`/O` 是“组织”，`/OU` 是“组织单位”，`/CN` 是“通用名称”。至少提供 `/C` 和 `/CN`（例如 `"/C=CN/CN=Localhost Root CA"` 作为个人本机测试使用）
- 生成私钥时会提示设置保护密码，在用该 CA 证书颁发其他证书时，需要输入这里设置的密码。请记住设置的密码。出于安全原因不建议忽略密码保护，如果坚持不使用密码，请额外添加 `-nodes` 参数（openssl 3.0 使用 `-noenc`）。

2. 从自签名根证书颁发其他证书

```
# 先创建 CSR，下例中使用通配符域名
```

```
openssl req -newkey rsa:2048 -nodes \
-keyout example.com.key -out example.com.csr \
-subj "/C=CN/CN=*.example.com"
```

```
# 生成证书并用 CA 给证书签名（需要在 bash 下运行，sh 不支持）
```

```
# 示例额外添加了 DNS 和 IP 地址，可根据需要取舍
```

```
openssl x509 -req -days 365 -sha256 \
-extfile <(printf "subjectAltName=DNS:example.com,DNS:test.example.com,IP:192.1
-CA root_ca.crt -CAkey root_ca.key -CAcreateserial \
-in example.com.csr -out example.com.crt
```

- 说明:

- openssl 1.1.1 及以下版本不支持用临时 csr 用 CA 签发证书，所以需要分成两条命令
- 如果不需要添加扩展的 DNS 和 IP 等信息，`-extfile` 整行可忽略
- 为了在一条命令中添加扩展信息，使用 bash 的支持把配置信息转成文件描述符。因此需要在 bash 下执行，否则需要先把配置信息保存为真实文件。
- 生成 CSR 的参数与前面创建根证书的说明类似，算法也可以改用 ec
- `-CA` 和 `-CAkey` 指定为前面生成的根证书及私钥文件
- `-in` 指定前面生成的 csr 文件，`-out` 为输出证书文件名

其他常用操作：

```
# 转换格式, PEM -> DER , 一般是 Java 需要
openssl x509 -in cert.pem -inform PEM -outform DER -out cert.der

# 转换格式 PEM -> PKCS12 , 一般是 IIS 需要
openssl pkcs12 -inkey cert.key -in cert.crt -export -out cert.pfx

# 某些服务器要求 pem 扩展名, 并且没有额外的配置来指定私钥文件, 组合两个文件即可
copy cert.crt + cert.key cert-key.pem
```

mkcert 快速参考

[mkcert](#) 使用 go 语言开发, 本机使用比较方便。一行命令就可以创建自签名根证书, 并把根证书安装到常用位置。接下来要生成服务需要的证书也都很简单。

安装

下载安装 ([查看最新版本](#))

macOS Linux Windows

```
# macOS
brew install mkcert

# 如果在用 firefox 浏览器
brew install nss
```

使用

1. 首先, 需要创建与安装根证书: `mkcert -install`, 该命令创建 `rootCA-key.pem` 并且:
 - 安装到 Windows 和 macOS 的系统证书存储
 - 安装到 `JAVA_HOME` 环境变量指向的 JAVA 环境
 - 安装到 Firefox (仅 macOS 和 Linux)
 - Windows 系统需要手动进入 选项->隐私与安全->证书->查看证书->证书颁发机构->导入 mkcert 生成的 `rootCA-key.pem` 文件)
2. 创建自签名证书, 例如创建一个支持 `example.com` 通配符及 `localhost` 环回地址的证书:

```
mkcert example.com "*.example.com" example.test localhost 127.0.0.1 :::1  
# 将生成证书文件 ./example.com+5.pem 和私钥 ./example.com+5-key.pem
```

step-ca 快速参考

查看 [step-ca](#) 官方英文文档:

- <https://smallstep.com/docs/step-ca/installation>
- <https://smallstep.com/docs/step-ca/getting-started>

总结

生成证书的工具很多, [openssl](#) 是容易取得的工具, 用起来相对较复杂。本地测试用完即抛可以用 [mkcert](#)。需要更复杂的 CA 功能甚至搭建私有 CA, [step-ca](#) 功能和易用性都相当不错。

当然其它类似工具还有很多没有一一列出。本文仅为个人使用体验, 如有错漏或建议欢迎在评论中指正。

转载许可声明



本作品采用[知识共享-署名-相同方式共享 4.0 国际许可协议](#)进行许可, 转载时请注明原文链接。

1. [传输 | 凤凰架构](#)
2. <https://www.openssl.org/docs/man1.1.1/man1/req.html>



作者
Kompost


如何安装和信任自签名证书 →
2021/11/11

昵称

邮箱

网址

键入内容...

表情 图片 预览 

发送评论

1 条评论 ▾



[小电动](#) 2022-07-27 Firefox 102 Windows 10

我自己测试发现 `mkcert` 已经进入ubuntu 22.04 LTS ppa仓库中，只需要
`sudo apt install mkcert` 就可以完成安装

赞同 (0) 回复

Powered By [Artalk](#)