Kleines SSL Handbuch

Daniel Klaffenbach

18.07.2007

Inhaltsverzeichnis

1 Einleitung

Dieses Dokument kann als Tutorial/HOWTO gesehen werden und richtet sich an versierte Benutzer, die Kenntnisse im Umgang mit der Kommandozeile haben. Es beschreibt wesentliche Schritte, die nötig sind, um SSL Zertifikate mit OpenSSL auszustellen. Anwendung wird dieses Dokument sicherlich in erster Linie im Bereich kleinerer Netzwerke finden, um beispielsweise einen Server zu Hause mit SSL abzusichern. Voraussetzung ist eine installierte Version von OpenSSL. Der Author verwendete zum Zeitpunkt der Erstellung dieses Dokumentes die Version 0.9.8d.
Die benutzten Verzeichnisse können, abhängig von der gewählten Linux-Distribution, unterschiedlich sein.

Zunächst ist es wichtig, Zertifikate zu verstehen und zu wissen, warum man ein Digitales Zertifikat einsetzen möchte. Als Einführung empfiehlt sich den Wikipedia-Artikel über Digitale Zertifikate zu studieren.
Ferner ist es wichtig, mit Zertifikaten richtig umzugehen. Im Bereich des Heimnetzwerkes mag das nicht von besonders großer Relevanz sein, aber sobald ein öffentlich zugänglicher Server per SSL abgesichert wird, sollte besondere Vorsicht geboten sein. Solche Vorsichtsmaßnahmen gehen weit über den Fokues dieses Dokumentes hinaus und werden deshalb nicht näher erläutert. Auf jeden Fall sollten Sie aber darauf achten, dass Ihre Zertifikate nicht allgemein zugänglich sind. Besonders betroffen davon sind die Privaten Schlüssel, die nicht umsonst in dem gesonderten Ordner `private` aufbewahrt werden. Dieser Ordner sollte nur für den Administrator les- und screibbar sein, aber nicht für andere Benutzer und Gruppen.

Der Autor kann keinesfalls für die Richtigkeit und Vollständigkeit der angegeben Informationen garantieren.

2 Selbstsignierte Zertifikate erstellen

2.1 Einen privaten Schlüssel erstellen

2.1.1 Option 1: Einen RSA-Schlüssel erstellen

Ein RSA-Schlüssel kann sowohl zum Verschlüsseln als auch zum Signieren benutzt werden. Einen solchen Schlüssel erstellt man mit folgenden Befehl:

```
openssl genrsa -des3 -out privatekey.pem 2048
```

Falls kein Passwort gewünscht wird, kann man die Option `-des3` auch weglassen. Das wird besonders dann empfohlen, wenn Sie diesen Schlüssel in Kombination mit einem Server-Zertifikat nutzen, weil Sie sonst das Passwort bei jedem Start des entsprechenden Dienstes eingeben müssten.

2.1.2 Option 2: Einen DSA-Schlüssel erstellen

Ein DSA-Schlüssel kann **ausschließlich** zum Signieren benutzt werden. Zuerst müssen Sie die Parameter generieren, aus denen dann der Schlüssel erstellt wird:

```
openssl dsaparam -out dsaparam.pem 2048
```

Danach kann folgendermaßen der DSA-Schlüssel erzeugt werden:

```
openssl gendsa -des3 -out privatekey.pem dsaparam.pem
```

Falls kein Passwort gewünscht wird, kann man die Option `-des3` auch weglassen. Das wird besonders dann empfohlen, wenn Sie diesen Schlüssel in Kombination mit einem Server-Zertifikat nutzen, weil Sie sonst das Passwort bei jedem Start des entsprechenden Dienstes eingeben müssten.

2.2 Ein selbst-signiertes Zertifikat erstellen

Ein selbst signiertes Zertifikat erstellen Sie wie folgt:

```
openssl req -new -x509 -key privatekey.pem -out cert.pem -days 365
```

Dabei ist `privatekey.pem` Ihr im vorherigen Schritt erstellte private Schlüssel und `cert.pem` das gewünschte selbst-signierte Zertifikat. Die Gültigkeit beträgt im angegebenen Beispiel 365 Tage.

3 Die eigene Zertifizierungsstelle

3.1 Eine Zertifizierungsstelle einrichten

Um Zertifikate signieren zu können, benötigt man zunächst eine Zertifizierungsstelle, die so genannte Root-CA. In diesem Beispiel wird die zu erstellende CA einfach als „demoCA“ bezeichnet. Dazu erstellen wir zunächst das nötige Verzeichnis:

```
cd /etc/ssl
mkdir demoCA
```

Nun muss man die SSL-Konfiguration anpassen. Dazu öffnet man mit dem Editor seiner Wahl die Datei `/etc/ssl/openssl.cnf`:

```
 [ ca ]
default_ca      = CA_default            # Der Standard-CA Teil

 [ CA_default ]

dir            = .                      # Verzeichnis unserer CA
database       = $dir/index.txt         # Index Datei
new_certs_dir  = $dir/newcerts          # Verzeihnis für neue Zertifikate

certificate    = $dir/cacert.pem        # Das CA Zertifikat
serial         = $dir/serial            # Seriennummern-Datei
private_key    = $dir/private/cakey.pem # Privater Schlüssel unserer CA
RANDFILE       = $dir/private/.rand     # Datei mit Zufallzahl

default_days   = 365                    # Wie lange das Zertifikat gültig sein soll
default_crl_days= 30                    # Wie lange bis zur nächsten Zertifikatsperrliste
default_md     = sha1                   # MD Algorithmus

policy         = policy_any             # Standard-Richtlinie
email_in_dn    = no                     # EMail nicht in den DN des Zertifikats schreiben

name_opt       = ca_default             # Betreffsanzeigeoption
cert_opt       = ca_default             # Zertifikatsanzeigeoption
copy_extensions = none                  # Erweiterungen nicht von der Anforderung kopieren

 [ policy_any ]
countryName            = supplied
stateOrProvinceName    = optional
organizationName       = optional
organizationalUnitName = optional
commonName             = supplied
emailAddress           = optional

 [ req ]
default_bits            = 2048
default_keyfile         = privkey.pem
distinguished_name      = req_distinguished_name
attributes              = req_attributes
x509_extensions         = v3_ca

 [ req_distinguished_name ]
countryName                     = Name des Landes (2 Buchstaben)
countryName_default             = DE     #DE für Deutschland
countryName_min                 = 2      #Minimal 2 Buchstaben
countryName_max                 = 2      #Maximal 2 Buchstaben

stateOrProvinceName             = Bundesland
stateOrProvinceName_default     = Sachsen #Standard Bundesland
```

```
localityName                    = Ort (z.B. Stadt)
localityName_default            = Chemnitz #Standard Stadt

0.organizationName              = Name der Organisation (z.B. Firma)
0.organizationName_default      = Heimrechenzentrum


organizationalUnitName          = Abteilungsname
#organizationalUnitName_default = Zertifizierungsstelle #brauchen wir nicht unbedingt

commonName                      = Common Name (z.B. Name des Servers)
commonName_max                  = 64

emailAddress                    = Email Adresse
emailAddress_max                = 64
emailAddress_default            = webmaster@domain.org #Ihre Email Adresse

 [ req_attributes ]
challengePassword               = Ein Passwort
challengePassword_min           = 0      #Password nicht notwendig
challengePassword_max           = 20     #Maximal 20 Stellen

 [ v3_req ]
# v3 Erweiterungen, die zur Zertifikatsanforderung hinzugef"ugt werden
basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment

 [ v3_ca ]
# CA Erweiterungen
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid:always,issuer:always
basicConstraints = CA:true
[ crl_ext ]
# CRL Erweiterungen
# Nur issuerAltName und authorityKeyIdentifier sind in einer CRL sinnvoll
# issuerAltName=issuer:copy
authorityKeyIdentifier=keyid:always,issuer:always

[ proxy_cert_ext ]
# Diese Erweiterungen sollten hinzugefügt werden, wenn ein Proxy-
# Zertifikat erstellt wird
basicConstraints=CA:FALSE
nsComment = "OpenSSL Zertifikat"
```

Hinweis: Sie können Einträge wie `Attribut_default` auskommentieren, wenn Sie der Meinung sind, dass Sie in jedem Zertifikat stets andere Angaben machen werden. Sie können aber trotzdem die `Attribut_default` Einträge beibehalten und später abweichende Angaben machen.

Nachdem die OpenSSL-Konfigurationsdatei angepasst wurde wechselt man in das Verzeichnis der Zertifizierungsstelle und generiert den privaten Schlüssel der zukünftigen CA:

```
cd /etc/ssl/demoCA
mkdir private
mkdir newcerts
echo '01' > serial
touch index.txt
touch index.txt.attr
openssl genrsa -des3 -out ./private/cakey.pem -rand ./private/.rand 2048
```

Sie bei diesem Prozess nach einem Passwort gefragt, was Sie sich unbedingt merken sollten. Dieses Passwort brauchen Sie später zum signieren von Zertifikatsanforderungen.
Nun braucht man noch das eigentliche Zertifikat der CA, das wie folgt erstellt wird:

```
openssl req -new -x509 -days 730 -key ./private/cakey.pem -out cacert.pem
```

Nachdem Sie das soeben gewählte Passwort eingegeben haben werden Sie nach Angaben zu Ihrer Zertifizierungsstelle gefragt. Wenn Sie die Konfigurationsdatei `/etc/ssl/openssl.cnf` angepasst haben, brauchen Sie die Angaben nur mit Enter bestätigen. Als `Common Name` können Sie hier momentan einen beliebigen Namen wie „Zertifizierungsstelle“ verwenden.
Damit ist die Erstellung der Zertifizierungsstelle abgeschlossen. Das Zertifikat „cacert.pem“ kann nun in die Browser der Klienten importiert werden, damit diese später bei der Verwendung von abgesicherten Verbindungen keine Warnung ausspucken, dass die Zertifizierungsstelle unbekannt ist.

3.2 Eine Zertifikatsanforderung (Request) erstellen

Um ein Zertifikat von der CA anzufordern, benutzen Sie folgenden Befehl:

```
openssl req -new -keyout ./private/serverkey.pem -out req.csr -days 365
```

Dabei ist `serverkey.pem` der private/geheime Schlüssel; `req.csr` enthält die Zertifikatsanforderung.

Folgen Sie dem Assistenten. Bestätigen Sie die Vorgaben, die Sie in der Datei `/etc/ssl/openssl.cnf` gemacht haben, mit Enter oder ändern Sie die Werte Ihren Wünschen entsprechend.
Wichtig dabei ist, dass Sie bei `Common Name` den vollen Domainnamen (FQDN) des Servers angeben, für den Sie das Zertifikat erstellen, das heißt der Name, unter dem der Server später zu erreichen sein wird (also beispielsweise `www.server.domain.org` im Fall eines Webservers).

3.3 Die Zertifikats-Anforderung unterzeichnen

Da wir nun sowohl eine Zertifizierungsstelle als auch eine Zertifikatsanforderung haben, können wir das gewünschte Server-Zertifikat nun signieren:

```
openssl ca -cert cacert.pem -key ./private/cakey.pem -in req.csr -out
    ./newcerts/servercert.pem
```

Nun wird man gefragt, ob man das Zertifikat wirklich unterschreiben möchte. Darauf antwortet man nun mit „y“.
Im Verzeichnis „newcerts“ erhält man nun 2 neue Zertifikate, „01.pem“ und „servercert.pem“. Diese sind identisch, das heißt eins von beiden können Sie nun in das Verzeichnis des entsprechenden Dienstes verschieben, der abgesichert werden soll.
Da die Schlüssel passwortgeschützt sind, müssten Sie bei jedem Start des abgesicherten Dienstes das Passwort eingeben. Um dem aus dem Weg zu gehen, können Sie das Passwort entfernen:

```
openssl rsa -in ./private/serverkey.pem -out ./private/serverkey2.pem
mv ./private/serverkey2.pem ./private/serverkey.pem
```

4 Quellen

- Linux Manpages
- http://www.openssl.org/docs/apps/ca.html
- http://www.openssl.org/docs/HOWTO/certificates.txt
- http://www.openssl.org/docs/HOWTO/keys.txt