Informacje

- **Kontakt:** katarzyna.mazur@umcs.pl
- **Konsultacje:** pokój 412 na 4 piętrze, przed konsultacjami proszę o wiadomość mailową
- **Zasady zaliczenia:**
- **Materiały, aktualności, zmiany terminów zajęć:** `https://kampus.umcs.pl/course/view.php?id=15080`

Teoria

Zadaniem urzędu certyfikacyjnego (Certification Authority, CA), jest poręczenie, że dany klucz publiczny należy do konkretnej osoby bądź podmiotu. Ze strony klienta sygnatura CA znajdująca się w certyfikacie serwera lub klienta jest gwarancją autentyczności tego certyfikatu. Chociaż sami twórcy OpenSSL nie zalecają jego użycia do tworzenia profesjonalnego PKI, to na jego bazie powstają takie projekty jak choćby OpenXPKI. OpenSSL natomiast doskonale nadaje się do celów edukacyjnych – to świetny sposób, żeby w praktyce dowiedzieć się, jak działa CA, oraz stworzyć prywatne CA, które będzie używane w sieci wewnętrznej. Jeżeli chcemy wystawiać certyfikaty np. dla naszych wewnętrznych serwerów HTTPS bądź klientów OpenVPN, to OpenSSL jest rozwiązaniem, które się do tego nada. Narzędzie openssl oferuje niemal wszystko, co z technicznego punktu widzenia jest potrzebne do stworzenia CA. Jednak warto pamiętać, że infrastruktura klucza publicznego to coś więcej niż samo generowanie certyfikatów.

Lista CRL pozwala wycofywać certyfikaty przed czasem upływu ich ważności. Należy ją publikować w ściśle określonych odstępach czasu, a oprogramowanie korzystające z certyfikatów powinno automatycznie sprawdzać jej zawartość (nie zawsze tak się dzieje).

Zadania

5.1 Wykorzystując bibliotekę openssl, utwórz Centrum Autoryzacji, tworząc klucz prywatny, publiczny oraz jego certyfikat.

5.2 Wykorzystując bibliotekę openssl, utwórz Certyfikat dla dowolnego użytkownika.

5.3 Wykorzystując bibliotekę openssl, unieważnij wystawiony certyfikat w zadaniu **5.2**. Wygeneruj aktualną listę CRL.

5.4 Wykorzystując narzędzie easy-rsa utwórz Centrum Autoryzacji, tworząc klucz prywatny, publiczny oraz jego certyfikat.

5.5 Wykorzystując narzędzie easy-rsa utwórz Certyfikat dla dowolnego użytkownika.

5.6 Zainstaluj najnowszą wersję serwera Apache. Zabezpiecz swoją stronę samopodpisanym certyfikatem. Pozbądź się błędu *Warning: Potential Security Risk Ahead*, który pojawia się podczas wchodzenia na stronę zabezpieczoną samopodpisanym certyfikatem.

Linki

- https://roll.urown.net/ca/ca_root_setup.html
- http://wazniak.mimuw.edu.pl/images/b/b4/Bsi_10_lab.pdf
- https://stackoverflow.com/questions/94445/using-openssl-what-does-unable-to-write-random-state-mean
- https://github.com/openssl/openssl/issues/7754
- https://www.cockroachlabs.com/docs/stable/create-security-certificates-openssl.html
- https://developer.mozilla.org/en-US/docs/Mozilla/Projects/NSS/tools/NSS_Tools_crlutil
- https://github.com/OpenVPN/easy-rsa

Odpowiedzi

5.1 Utworzenie struktury katalogów i przygotowanie niezbędnych plików:

```
mkdir root-ca
mkdir -p root-ca/certreqs
mkdir -p root-ca/certs
mkdir -p root-ca/crl
mkdir -p root-ca/newcerts
mkdir -p root-ca/private

cd root-ca/

chmod 700 private
touch root-ca.index
touch root-ca.serial
echo 00 > root-ca.serial
echo 00 > root-ca.crlnum
touch root-ca.cnf
touch private/.rnd

export OPENSSL_CONF=./root-ca/root-ca.cnf
```

Generowanie pary kluczy dla CA:

```
openssl genrsa -out root-ca.key.pem 4096
cp root-ca.key.pem private
chmod 400 private/root-ca.key.pem
```

Generowanie klucza publicznego z klucza prywatnego:

```
openssl rsa -in private/root-ca.key.pem -pubout -out private/root-ca-pub.key.pem
```

Generowanie CSR:

```
openssl req -new -key private/root-ca.key.pem -config root-ca.cnf -out root-ca.req.pem
```

Wyświetlenie CSR:

```
openssl req -verify -in root-ca.req.pem -config root-ca.cnf -noout -text -nameopt multiline
openssl req -verify -in root-ca.req.pem -config root-ca.cnf -noout -text -reqopt no_version,no_pubkey,no_sigdump -nameopt multiline
```

Utworzenie certyfikatu CA (sampodpisanie CSR):

```
openssl ca -selfsign -config root-ca.cnf -in root-ca.req.pem -out root-ca.cert.pem
```

5.2 Utworzenie pary kluczy dla użytkownika:

```
openssl genrsa -out certs/node.key 2048
chmod 400 certs/node.key
```

Utworzenie pliku konfiguracji certyfikatu dla użytkownika:

```
touch node.cnf
```

```
[ req ]
prompt=no
distinguished_name = distinguished_name
req_extensions = extensions

[ distinguished_name ]
organizationName = UMCS
commonName = student

[ extensions ]
subjectAltName = DNS:root
```

Wygenerowanie CSR (żądania podpisania certyfikatu):

```
openssl req -new -key certs/node.key -config node.cnf -out node.csr
```

Podpisanie CSR od użytkownika (podpisanie żądania wydania certyfikatu):

```
openssl x509 -req -days 365 -CA root-ca.cert.pem -CAkey private/root-ca.key.pem -CAcreateserial -CAserial root-ca.serial
-in node.csr -out node.pem
```

Wyświetlenie certyfikatu:

```
openssl x509 -in node.pem -noout -text
```

5.3 Utworzenie listy CRL:

```
openssl ca -config root-ca.cnf -gencrl -out crl/list.crl
```

Cofnięcie certyfikatu (unieważnienie):

```
openssl ca -config root-ca.cnf -revoke [plik z certyfikatem do cofniecia]
openssl ca -config root-ca.cnf -revoke node.pem
```

Wyświetlenie listy CRL:
```
openssl crl -in crl/list.crl -noout -text
```

Importowanie CRL do przeglądarki:
```
sudo apt install libnss3-tools
```

5.4 Instalacja:
```
sudo apt install easy-rsa
```

Instalacja i inicjalizacja PKI:

```
mkdir easy-rsa
ln -s /usr/share/easy-rsa/* easy-rsa/
chmod 700 easy-rsa
cd easy-rsa
./easyrsa init-pki
```

Tworzenie CA:

```
nano vars
```

```
set_var EASYRSA_REQ_COUNTRY    "US"
set_var EASYRSA_REQ_PROVINCE   "NewYork"
set_var EASYRSA_REQ_CITY       "New York City"
set_var EASYRSA_REQ_ORG        "DigitalOcean"
set_var EASYRSA_REQ_EMAIL      "admin@example.com"
set_var EASYRSA_REQ_OU         "Community"
set_var EASYRSA_ALGO           "ec"
set_var EASYRSA_DIGEST         "sha512"
```

```
./easyrsa build-ca
./easyrsa build-ca nopass
```

5.5 Generowanie CSR:
```
./easyrsa gen-req node
```

Podpisanie CSR:
```
./easyrsa sign-req client node
openssl verify -CAfile pki/ca.crt pki/issued/node.crt
```

5.6 Polecenia:

```
sudo apt install apache2

sudo a2enmod ssl

sudo systemctl restart apache2

sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048
-keyout /etc/ssl/private/apache-selfsigned.key -out /etc/ssl/certs/apache-selfsigned.crt

sudo nano /etc/apache2/sites-available/student.conf

<VirtualHost *:443>
   ServerName student
   DocumentRoot /var/www/student

   SSLEngine on
   SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt
   SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key
</VirtualHost>

sudo mkdir /var/www/student

sudo nano /var/www/student/index.html

<html>
<body>
<h1>It worked!</h1>
</body>
</html>

sudo a2ensite student.conf

sudo apache2ctl configtest

sudo systemctl restart apache2
```

Wejście na stronę:
`https://127.0.0.1/index.html`

Pozbycie się błędu:

```
sudo cp /etc/ssl/certs/apache-selfsigned.crt /usr/local/share/ca-certificates/
sudo update-ca-certificates
```

Przekierowanie HTTP do HTTPs:

```
sudo nano /etc/apache2/sites-available/student.conf
```

```
<VirtualHost *:443>
   ServerName student
   DocumentRoot /var/www/student

   SSLEngine on
   SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt
   SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key
</VirtualHost>

<VirtualHost *:80>
    ServerName student
    Redirect / https://student/
</VirtualHost>

sudo apachectl configtest
sudo systemctl reload apache2
```