OpenSSL

A. Instalacja pakietu:

sudo aptitude openssl

B. Zastosowanie pakietu OpenSSL:

- **Tworzenie własnego CA,**
- **Generowanie kluczy, wniosków i certyfikatów.**

<u>Tworzymy własny Urząd Certyfikacji (CA). Wymagania:</u>

- CA powinno być utworzone na bezpiecznym komputerze - najlepiej odłączonym od Internetu lub przynajmniej za Firewallem,
- Regularne robienie kopii bezpieczeństwa wystawionych certyfikatów tak aby w razie potrzeby unieważnić któryś z certyfikatów.

Przechodzimy do pliku **/etc/ssl/openssl.cnf**, odnajdujemy sekcję **[CA_default]** i dokonujemy sprawdzenia poprawności ścieżek składowania certyfikatów i kluczy:

[Ca_default]
dir=/etc/ssl/private # główny katalog, w którym zapisywane są pliki
certs=/etc/ssl/certs # katalog, w który zapisywane są certyfikaty
crl_dir=$dir/crl # katalog z listą unieważnionych certyfikatów
private_key=$dir/ssl-cert-snakeoil.pem # **klucz prywatny CA**
database = $dir /index.txt # baza danych o wystawionych certyfikatach
certfyfikate = $dir/cacert.pem # **Certyfikat CA – do podpisu wniosków**
serial = $dir/serial #plik pomocniczy z bieżącym numerem – inkrementowany po każdym wystawieniu certyfikatu

crl =/$dir/crl.pem # bieżąca lista certyfikatów unieważnionych

Upewniamy się czy istnieje katalog podany w zmiennej dir =/etc/ssl oraz wszystkie jego podkatalogi. Jeśli nie, to je zakładamy. Dla katalogu 'private' ustawiamy uprawnienia tak, aby tylko użytkownik Root mógł do niego wejść

Tworzymy pliki **index.txt** (pusty) oraz **serial** (zawiera wpis 00):

touch /etc/ssl/index.txt
touch /etc/ssl/serial

Tworzenie Urzędu Certyfikacyjnego składa się z dwóch czynności:

- **Wygenerowania klucza prywatnego Urzędu Certyfikacji**
- **Wygenerowanie certyfikatu Urzędu Certyfikacji (klucz publiczny)**

Generujemy klucz prywatny centrum certyfikacji CA (czynność jednorazowa):

W katalogu **/etc/ssl** wydajemy polecenie:

openssl genrsa 1024 -des3 –out private/ssl-cert-snakeoil.pem

genrsa – polecenie generujące klucz prywatny według algorytmu RSA
des3 – polecenie szyfruje klucz prywatny algorytmem des3. Inne opcje: **des, idea**
1024 – długość klucza prywatnego zorientowana bitowo
out – nazwa pliku wyjściowego

Zostaniemy poproszeni o nadanie **hasła do klucza prywatnego**. Po potwierdzeniu hasła klucz zostanie zapisany w katalogu **/etc/ssl/private/cakey.pem**

Generujemy certyfikat którym będziemy posługiwać się do podpisywania innych certyfikatów (np. dla klientów).

openssl req –x509 –new –days 365 –key /etc/ssl/private/ssl-cert-snakeoil.key > /etc/ssl/private/cacert.pem

req – polecenie generujące certyfikat bądź żądanie o podpis certyfikatu
new – opcja generuje nowe żądanie certyfikatu.
x509 - opcja zleca wygenerowanie podpisanego certyfikatu

days - ważność certyfikatu podana w dniach. Działa tylko z opcją x509 (defaultowo 30 dni.
key – określa ścieżkę/nazwę klucza prywatnego

Zostaniemy poproszeni o podanie kilku pól zawartych w certyfikacie. Po podaniu **hasła do klucza prywatnego** certyfikat zostanie zapisany w **/etc/ssl/private/ca cert**

Posiadając klucz prywatny i publiczny dla CA (certyfikat CA) można przystąpić do wystawiania certyfikatów innym podmiotom.

Tworzenie klucza prywatnego dla określonego podmiotu (serwer, klient):

Będąc w katalogu **/etc/ssl** piszemy:

openssl genrsa 1024 –des3 > /etc/ssl/private/serwerkey.key

Program OpenSSL zapyta o hasło – będzie to hasło do klucza prywatnego serwera

Warto standaryzować nazwy tworzonych kluczy i certyfikatów:

- na początku pisząc nazwę podmiotu (tutaj serwer a może być dowolna nazwa: klient, klien1… etc.),
- kolejno nazwę dokument (tutaj **key** a może być dowolna nazwa np. klucz),
- oraz rozszerzenie **.pem**.

Generowanie wniosku o wystawienie certyfikatu:

openssl req –new –key /etc/ssl/private/serwerkey.key > serwerreq.pem

Podajemy hasło klucza prywatnego danego podmiotu (tu serwera). Jeżeli hasło będzie poprawne, zostaniemy zapytani o dane do wniosku (tak jak byliśmy pytani w przypadku certyfikatu dla CA). W przypadku podmiotu, który będzie serwerem bezpośrednio połączonym z Internetem, w polu **'Common Name'** podajemy pełną nazwę domenową (FQDN – 'Fully Qualified Domain Name'). Jest to ważne, ponieważ niektóre programy zgłaszają niezgodność certyfikatu z adresem serwera.

Wystawienie certyfikatu dla określonego podmiotu (serwer, klient):

Aby wystawić certyfikat dla podmiotu wykorzystujemy wygenerowany wcześniej wniosek:

openssl ca –notext –in serwerreq.pem > serwercert.pem

ca – aplikacja wystawiająca/unieważniająca certyfikaty
in – nazwa wniosku o wystawienie certyfikatu

Zostaniemy zapytani o hasło do **klucza prywatnego Urzędu Certyfikacji ssl-cert-snakeoil.pem** (nie mylić z hasłem do klucza prywatnego podmiotu).

Następnie OpenSSL pokaże szczegóły certyfikatów i zapyta czy chcemy go podpisać. Wygenerowaliśmy parę kluczy (prywatny i publiczny - certyfikat) dla podmiotu (serwera). Do zestawienia szyfrowanych połączeń, sieci VPN, IPsec będą potrzebne też klucze dla użytkowników. Generujemy je w sposób analogiczny do powyższego.

Można podejrzeć zawartość plików **index.txt** oraz **serial** które zostały zaktualizowane po podpisaniu wniosku. Plik **index.txt** jest potrzebny przy odwołaniu certyfikatu.

Uwagi:

- W polu **'Common Name'** podajemy dane jednoznacznie identyfikujące użytkownika np. login korporacyjny,
- Hasło zabezpieczające klucz prywatny będzie wpisywane z klawiatury, co nie jest najbezpieczniejszym rozwiązaniem. W takich sytuacjach możemy ściągnąć hasło z klucza prywatnego. Dostęp do klucza powinien mieć jedynie Root.

Ściąganie hasła z klucza prywatnego serwera:

sudo openssl rsa –in /etc/ssl/private/serwerkey.pem > /etc/ssl/private/serwerkey.pem_bezhasla

rsa – narzędzie do przetwarzania kluczy

Unieważnianie certyfikatów:

Do unieważniania certyfikatów służy parametr **revoke** biblioteki openssl, np.:

openssl ca –revoke klient1.pem

revoke – parametr określający nazwę certyfikatu do unieważnienia

Openssl zapyta o hasło do klucza prywatnego Urzędu Certyfikacji i unieważni certyfikat.

Generujemy jeszcze listę CRL, w której są zapisane unieważnione certyfikaty:

openssl ca –gencrl –out crl.pem

gencrl - opcja generuje listę CRL na podstawie informacji w pliku **index.txt**

Inne dostępne polecenia:

asn1parse
Analiza składni sekwencji ASN.1.
ca
Zarządzanie ośrodkami certyfikacji (CA).
ciphers
Opis zestawu dostępnych szyfrów.
crl
Zarządzanie listą unieważnionych certyfikatów (CRL).
crl2pkcs7 Konwersja z CRL do PKCS#7.
dgst
Obliczanie skrótu wiadomości.
dh
Zarządzanie parametrami Diffie-Hellmana. Przedawnione przez dhparam.
dsa
Zarządzanie danymi DSA.
dsaparam Tworzenie parametru DSA.
enc
Szyfrowanie.
errstr
Konwersja numeru błędu na komunikat błędu.
dhparam
Tworzenie i zarządzanie parametrami Diffie-Hellmana.
gendh
Tworzenie parametrów Diffie-Hellmana. Przedawnione przez dhparam.
gendsa
Tworzenie parametrów DSA.
genrsa
Tworzenie parametrów RSA.
passwd
Generowanie skrótu hasła.
pkcs12
Zarządzanie danymi PKCS#12.

pkcs7
Zarządzanie danymi w formacie PKCS#7.
rand
Tworzenie pseudo-losowych bajtów.
req
Zarządzanie żądaniami podpisu certyfikatu X.509 (CSR).
rsa
Zarządzanie danymi RSA.
rsautl
Narzędzie do podpisywania, weryfikowania, szyfrowania i deszyfrowania RSA.
s_client
Implementacja podstawowego klienta SSL/TLS potrafiącego nawiązywać przeźroczyste połączenia z odległym serwerem na SSL/TLS. Służy jedynie do testowania i dostarcza tylko podstawowej funkcjonalności interfejsu, ale wewnętrznie korzysta z niemal pełnych możliwości biblioteki OpenSSL (SSL).
s_server
Implementacja podstawowego serwera SSL/TLS przyjmującego połączenia od odległych klientów obsługujących SSL/TLS. Służy jedynie do testowania i dostarcza tylko podstawowej funkcjonalności interfejsu, ale wewnętrznie korzysta z niemal pełnych możliwości biblioteki OpenSSL ssl. Posiada zarówno własny protokół wiersza poleceń do testowania funkcji SSL, jak i emuluje prosty serwer sieciowy SSL/TLS oparty o HTTP.
s_time
Licznik czasu połączenia SSL.
sess_id
Zarządzanie danymi sesji SSL.
smime
Przetwarzanie poczty S/MIME.
speed
Mierzenie szybkości algorytmu.
verify
Weryfikacja certyfikatu X.509.
version
Informacja o wersji OpenSSL.
x509
Zarządzanie certyfikatami X.509.