Certificate

Un certificat este o semnatura digitala care asociaza unei chei publice o identitate (numele unei organizatii sau al unei personae, adresa, s.a.). Certificatul putand fi folosit pentru a verifica faptul ca o cheie publica apartinie unei persoane sau unei organizatii.

Certificatele sunt folosite pentru criptarea cu chei publice pe scara larga. Algoritmii de criptare simetrici au marele dezavantaj ca nu asigura distribuirea sigura a cheilor, problema care este insa rezolvata de criptarea cu chei publice. In principiu criptarea cu chei publice functioneaza in felul urmator: daca o entitate E vrea sa primeasca mesaje criptate nu are decat sa isi publice cheia publica, astfel orice cunoaste respective cheie va putea sa trimite mesaje criptate catre E, mesaje care nu vor putea fi citite de nimeni in afara de E (evident atata timp cat E isi pastreaza in siguranta cheia privata). Singura problema care poate sa apara este aceea ca o alta entitate D publica la randul ei o cheie publica pe care o declara ca fiind cheia publica a entitatii E, in acest fel D va putea primi mesaje private destinate lui E. Pentru a rezolva aceasta problema E nu are decat sa puna cheia sa publica intr-un certificate sis a ceara unei a treia entitati T sa semneze digital respectivul certificate. In acest fel oricine are incredere in T va putea cere o verificare a certificatului lui E. Uzual T este numita Autoritate de Certificate (Certificate Authority (CA)).

Intr-un system de dimensiuni mari insa se poate ca doua entitati care doresc sa comunice intre ele sa detine certificate semnate de autoritati diferite si fiecare din entitati sa nu aiba incredere in autoritatea care a semnat certificatul celeilalte entitati. In aceste conditii este necesar ca si certificatele celor doua autoritati sa fie semnate de o a treia autoritate, care sa fie recunoscuta de ambele entitati. In felul acesta se ajunge la o structura ierarhica de certificate, in varful piramidei aflandu-se un certificat radacina, care reprezinta o autoritate care nu trebuie sa fie autentificata de nimeni.

Daca cheia privata corespunzatoare unui certificat a fost compromisa sau daca asocierea dintre o entitate si cheia publica nu este corecta sau s-a modificat, atunci certificatul trebuie sa fie revocat. Revocarea unui certificat se intampla rar, dar posibilitatea ca un certificate sa fie revocat obliga ca daca un certificate este gasit ca fiind autentic este necesara si o verificare a validitatii respectivului certificat. Verificarea validitatii unui certificate realizandu-se pe baza unor liste de revocare (Certificate Revocation List (CRL)). Asigurarea faptului ca o astfel de lista este tot timpul actuala este o cerinta importanta pentru un system cu de criptare cu chei publice (PKI). O alta modalitate de a verifica daca un certificate este valid consta in trimiterea uncei cereri de validare catre o autoritate utilizand protocolul Online Certificate Status Protocol (OCSP), pentru a afla starea unui certificate. Protocolul OCSP este construit peste protocolul HTTP, serverele OCSP fiind numite “OCSP responders”.

Dar in momentul de fata se pare ca se va impune o nou metoda care permite verificare stari unui anumit certificate si anume XML Key Management System (XKMS), care se bazeaza pe serviciile web.

Cel mai utilizat standard in domeniul certificatelor este X5.09, care a fost introdus in anul 1988, acesta a impus un system ierarhic stric in ceea ce priveste autoritatile de certificate (CAs).

Structura unui certificat X.509 v3:

- Certificate
  - Version
  - Serial Number
  - Algorithm ID
  - Issuer
  - Validity
    - Not Before
    - Not After
  - Subject
  - Subject Public Key Info
    - Public Key Algorithm
    - Subject Public Key
  - Issuer Unique Identifier (Optional)
  - Subject Unique Identifier (Optional)
  - Extensions (Optional)
    - ...
- Certificate Signature Algorithm
- Certificate Signature

Extensii de fisiere certificate:

- `.CER` – certificate codificate CER
- `.DER` – certificate codificate DER
- `.PEM` – certificate codificate Base64, certificatele sunt cuprinse intre "-----BEGIN CERTIFICATE-----" si "-----END CERTIFICATE-----"
- `.P7B`
- `.P7C`
- `.PFX`
- `.P12`

SSL/TLS

O aplicatie importanta a certificatelor o reprezinta Secure Socket Layer (SSL) sau mai nou Transport Layer Security (TLS). SSL este un protocol criptographic care asigura comunicatii sigure pe internet pentru activitati de genul: navigare, posta electronica, instant messaging etc.

Au fost dezvoltate 3 versiuni de SSL (v1.0, v2.0 si v3.0). Ultima versiune a fost lansata in anul 1996, ea fiind baza pentru TLS versiunea 1.0, care a fost definit ca si standardul RFC 2246 in 1999. In primele versiuni ale protocolului au fost folosite chei simetrice pe doar 40 de bitit datorita unei restrictii impuse de Guvernul American asupra exportului de cripto-tehnologie. Aceasta restrictie a fost impusa pentru ca serviciile secrete americane sa poata monitoriza traficul cryptat (realizand o decriptare prin forta

bruta). In prezent aceasta restrictie a fost eliminate astfel ca se folosesc chei cu lungimi de 128 de bitit (chiar mai lungi).

TLS asigura authentificarea si comunicarea de informatii private pe Internet utilizand criptografia. Uzual doar serverul este autentificat, in timp ce clientul ramane neautentificat (persoana, o aplicatie sau un browser de internet). Se poate insa realiza o autentificare mutuala (ambi participanti la comunicare stiind cu cine vorbesc).
Protocolul TLS consta din trei faze principale:

- alegerea algoritmilor de criptare, in present putand fi alesi urmatorii algorithmi:
  - pentru criptari cu chei publice: RSA, Diffie-Hellman, DSA;
  - pentru algoritmi de criptare simetrici: RC2, RC4, IDEA, DES, Triple DES, AES or Camellia;
  - pentru functii hash: MD2, MD4, MD5 or SHA.
- inter-schimbarea cheilor de baza si autentificarea pe baza certificatelor, in aceasta etapa se folosesc algoritmii de criptare cu chei publice;
- transferal datelor, in aceasta etapa se folosesc algoritmii simetrici, cheile folosite pentru criptare fiind cele inter-schimbate in pasul anterior.

TLS consta din interschimbarea de mesje numite inregistrari (records) intre client si server, fiecare inregistrare putand fi comprimata, criptata si marcata cu un numar (message authentication code MAC).

Masuri de siguranta folosite de TSL/SSL:

- clientul foloseste cheia publica a unei CA pentru a valida certificatul serverului.
- clientul verifica, ca autoritatea care a generat certificatul serverului se afla in lista de autoritati in care el are incredere
- clientul verifica perioada de validitate a certificatului primit de la server
- pentru contracararea atacurilor de tipul Man-in-the-Middle, clientul compara numele DNS al serverului cu cel de pe certificate
- numerotarea tuturor inregistrarilor
- folosirea algoritmilor de calculare a hash-urilor impreuna cu o cheie, astfel inainte de a se calcula un hash se realizeaza o criptare a datelor pentru care se doreste calcularea hash-ului si abea apoi se calculeaza hash-ul
- la final se transmite un hash pentru toate datele care au fost transmise
- SSL v3 utilizeaza SHA1 pentru calcularea hash-urilor, metoda fiind considerate mai sigura decat metoda MD5 care este folosita de SSL v2.

Aplicatii in care se foloseste TLS:

- TLS este folosit sub protocoale precum: HTTP, FTP, SMTP, NNTP etc., dar deasupra protocoalelor TCP si UDP.
- TLS este folosit in aplicatii de tunelare pentru a se implementa un VPN (Virtual Private Network)
- TLS este folosit si pentru cryptarea convorbirilor digitale (Voice over IP).

Pentru a utiliza TLS intr-o aplicatie nu este necesara implementarea sandardului, existand la ora acutala o serie de librarii care sunt open source si care pot fi folosite de catre programatori. O astfel de librarie este si OpenSSL (http://www.openssl.org/). Libraria OpenSSL pe langa faptul ca pune la dispozitia programatorului o serie de functii care pot fi folosite pentru a crea o aplicatie client server care sa comunice utilizand o conexiune sigura, contine si o serie de functii care pot fi utilizate pentru a cripta mesaje utilizand diverse metode de criptare, respectiv functii pentru calculul hash-urilor. De asemenea exista si un program care functioneaza in linie de comanda si care perite destarea diverselor functionalitati, el fiind si un exemplu de utilizare a librariei. Dupa pronirea programului OpenSSL, utilizatorul va putea scrie diverse comenzi (help, req, s.a.).

Partea practica

Utilizand tool-ul openssl sa se:

- genereze un certificat self-signed (acest certificate va fi folosit ca si certificate al autoritatii (CA certificate)):

```
req -config openssl.cnf -x509 -newkey rsa:2048 -keyout
cakey.pem -out cacert.pem
parametrii:
   req – permite generarea unei cereri de certificat
   -config – specifica fisierul de configurare utilizat
   -x509 – specifica tipul certificatului
   -newkey – solicita generarea unei noi chei
   rsa:2048 – cheia generata va fi pentru algoritmul RSA si va
              avea o lungime de 2048 biti
   -keyout – specifica fisierul in care va fi stocata cheia
             privata
   -out specifica fisierul in care va fi stocat certificatul

Dupa rularea cumenzii anterioare se vor obtine doua fisiere
cakey.pem – reprezinta cheia privata a autoritatii si
cacert.pem reprezinta certificatul autoritatii (contine cheia
publica a autoritatii), aceste doua fisiere sunt necesare
pentru a putea genera certificate semnate de aceasta
autoritate. La generarea cheii private utilizatorului i se va
cere sa introduca o parola, acea parola va fi folosita pentru
a cripta cheia privata, exista posibilitatea specificarii
algoritmului care va fi folosit pentru a cripta cheia privata,
daca nu se specifica nici un algoritm stunc sa va folosi
implicit algoritm-ul DES. Deschideti cele doua fisiere
generate cu un editor text (notepad) pentru a vedea
continutul. Copiati fisierul cakey.pem in directorul
“CAStore\private\”, iar cacert.pem in “CAStore”.
```

- sa se genereze doua cereri de certificate

pentru a genera o cerere de certificat, mai intai trebuie generate o cheie RSA:

```
genrsa -out keyfile.pem 2048
```

keyfile.pem - represinta numele fisierului in care va fi memorata cheia

Dup ace cheia a fost generata se poate folosi fisierul rezultat pentru a genera o cerere de certificate:

```
req -config openssl.cnf -new -key keyfile.pem -out req.pem
```

comanda este asemanatoare cu cea folosita pentru a genera un certificat auto-semnat, dar ca nu se mai solicita generarea cheii (a fost generata anterior), de asemenea nu mai apare parametrul -x509.

Deschideti cele doua fisiere cerere generate cu un editor text (notepad) pentru a vedea continutul.

- sa se semneze cele doua certificate generate anterior

cele doua cereri generate la punctual anterior pot fi folosite pentru a se genera cate un certificata, semnat de o autoritate.

Pentru a semna un certificate utilizand tool-ul OpenSSL se poate folosi comanda:

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

unde reg.pem reprezinta un fisier care contine o cerere de certificat, iar cert.pem este numele fisierului care va contine certificatul. Pentru a semna un certificat este nevoie de cheia privata a autoritatii, el fiind criptat cu aceasta cheie, de aceea la executarea acestei comenzi se va cere introducerea parolei cu care cheia privata a autoritatii (cakey.pem) a fost criptata.

Deschideti cele doua fisiere certificat generate cu un editor text (notepad) pentru a vedea continutul.

- sa se realizeze comunicarea prin SSL utilizand pentru client unul din certificatele create mai sus, iar pentru server celalalt certificat.

Pornire server:

s_server -cert cert1.pem -key keyfile1.pem -ssl3

Pornire client:

s_client –cert cert2.pem –key keyfile2.pem –Cafile cacert.pem –ssl3

Utilizati alti parametrii pentru a porni server-ul respective clientul.