HowTo ASIX

Certificats Digitas

Curs 2018 - 2019

Índex de continguts

Aprenentatges treballats 3

Documentació 4

Exemples d'ordres 5

Claus privades RSA 5

Certificats X509 6

Petició de certificació: Request 7

CA 8

Miscel·lània 8

TSL/SSL Conexions segures (HTTPS) 10

Creació/Gestió de certificats digitals 10

Certificas digitals 10

Crear certificats autosignats 10

Crear una CA pròpia: Certificate Authority 12

Crear el certificat del servidor (real) 15

Afegir/modificar/eliminar una passfrase de la clau privada 18

Examinar els continguts de certificats i claus privades 19

Estructura de directoris usada en els exemples 21

Ordre Openssl CA 22

Configuració Openssl 22

Tràfic segur amb TLS/SSL i STARTTLS 24

TLS/SSL i STARTTLS 24

Exemples d'aplicació TLS / SSL / StartTLS 25

HTTPS: Accés web segur 25

Pràctica amb Firefox 25
Pràctica amb s_client 26
Pràctica amb curl 29
Pràctica amb gmail / POP 29
LDAPS: Accés segur al servidor LDAP 31
Utilitats s_client, s_server, ncat 36
Client / Servidor ncat 36
Client / Servidor / Verificar amb openssl 36
Verificar 37
Obtenir / visualitzar 37
Observar certificat de Google 38
IMAP: Accés segur amb TLS / StartTLS 39
Uw-imap 39
Imaps 39
Imap + StartTLS 46
cyrus 47
POP: Accés segur amb TLS / StartTLS 55
SMTP: Accés al correu segur amb StartTLS 56
OpenVPN: Túnels VPN amb TLS 57
Generar certificats propis per a VPN 57
Observar els certificats predefinits 57
Generar-ne de nou amb la seva configuració openssl.conf 58
Crear noves claus manualment 58
Túnel amb comandes OpenVPN 60
Túnel amb Systemctl 61
Extensions 71
Practica extensions: openvpn i ldaps 71
Implementar extensions 71
Usant un fitxer específic d’extensions 71
Usant el fitxer global openssl.conf 74
Descripció de les Extensions 75
STANDARD EXTENSIONS 75
Basic Constraints. 75
Key Usage. 76
Extended Key Usage. 76
Subject Key Identifier. 77
Authority Key Identifier. 77
Subject Alternative Name. 77

Issuer Alternative Name. 78
Authority Info Access. 78
CRL distribution points. 79
Issuing Distribution Point 80
Certificate Policies. 80
Policy Constraints 81
Inhibit Any Policy 82
Name Constraints 82
OCSP No Check 82
Exemples / llistats d'aplicació d'extensions 82
Openssl.conf 91
Part Prèvia: configuració base openssl CA 92
Configurar req 92
Configurar policy 94
Configurar certs 95
Definir noves seccions 96
Underconstruction 98
Documetació RFCs: 98
OpenVPN Samples 102
openssl.conf 102
gen-sample-keys.sh 104
Format dels directoris de certificats 106

Aprenentatges treballats

1. Conceptes generals de Seguretat / Certificats.
    a. Criptografia simètrica / asimètrica / híbrida.
    b. Clau Secreta / Pública, Identitat, Certificat.
    c. Signar, Xifrar. Autenticació, Integritat, No repudi.
    d. Xifrat simètric DES de les claus privades.
    e. Models de seguretat: PKI Public Key Infraestructure.
    f. Models de seguretat:Web of trust.
    g. Entitats de certificació: CA.
    h. TLS / SSL / StartTLS

2. Cerificats digitals.
    a. Certificats autosignats
    b. Claus privades RSA.

    c. Peticions de certificació: request.
    d. Entitats de certificació: CA
    e. Certificats signats per una CA

3. Examinar claus, certificats i peticions.
    a. Consultar els fitxers de text: pem
    b. Consultar les dades de les claus privades RSA.
    c. Consultar les dades dels certificats.
    d. Consultar les dades de les peticions de certificació request.
    e. Afegir / Treure protecció de la clau privada amb passphrase 3DES.

4. Entitat de certificació CA.
    a. Estructura de directoris d'una CA: oficial /etc/pki, personalitzada.
    b. Fitxer de configuració de openssl.cnf.
    c. Policies aplicables: policy_match, policy_anithing. Redefinir la policy.
    d. Req: descripció del DN del certificat. Redefinició i personalització.
    e. Extensions: V3_ca, V3_req. Afegir / treure extensions.

5. Implementació de TLS/SSL (I).
    a. HTTPS. Implementar una seu web amb certificat autosignat.
    b. HTTPS. Implementar una seu web amb certificat avalat per una CA pròpia.
    c. Connexions clients TLS/SSL amb telnet, openssl s_client, curl, ncat.
    d. Implementar Túnels VPN amb TLS/SSL usant certificats propis avalats per una CA.
    e. Implementar Túnels VPN amb TLS/SSL usant systemctl i serveis clients i servidor. Usar certificats propis avalats per una CA.
    f. IMAPS: Implementar l'accés al servidor uw-imap amb IMAPs.
    g. IMAP StartTLS: Implementar l'accés al servidor uw-imap amb IMAP i activar StartTLS.
    h. Connexions client imap i pop amb openssl s_client a serveis locals o a serveis de google.
    i. POPs i POP+StartTLS.
    j. SMTPs i SMTP+StartTLS.
6. Implementació de tLS/SSL (II)
    a. Implementar openVPN amb certificats propis.
    b. Implementar un servidor ldaps amb certificat propi amb subject alternative names.

Documentació

Manual de Madboa (pràctic recomanable!)

- https://www.madboa.com/geek/openssl/

Manual de OpenSSL CookBook

- https://www.feistyduck.com/library/openssl-cookbook/

Documentació OpenSSL

- https://www.openssl.org/docs/
- Pàgines de manual

Mozilla MDN Web docs: Security:

- https://developer.mozilla.org/en-US/docs/Archive/Security

Exemples d'ordres

Exemples de:

- Claus privades RSA: comanda rsa, genrsa. Claus DSA. Formats PEM i DER.
- Certificate Request: comanda req
- Certificats digitals: X509

Claus privades RSA

Claus privades RSA

```
# openssl genrsa -des3 -out ca.key 2048
# openssl genrsa  -out server.key 2048
```

Passfrase des3

```
# openssl rsa -des3 -in server.key -out passfrase.server.key
# openssl rsa -in passfrase.server.key -out deleted-passfrase.server.key
# openssl rsa  -des3 -in passfrase.server.key -out new-passfrase.server.key
```

Llistar

```
# openssl rsa -noout -text -in serverkey.pem
# cat serverkey.pem
```

Conversió PEM / DER

```
# openssl rsa -in key.pem -outform DER -out key.der
# openssl rsa -inform DER -in  key.der  -outform PEM   -out key.pem
# openssl rsa -inform DER -in  key.der  -out key.pem
```

Extreure la clau pública de la privada:

```
# openssl rsa -in key.pem -pubout -out pubkey.pem
# openssl rsa  -noout -text -pubin -in pubkey.pem
```

PEM = capçalera + base64(DER) + peu

```
# cat key.pem
# cat mykey.pem | tail -n +2 | head -n -1 > noheaders.key.pem
# base64 --decode noheaders.key.pem > key.der
```

key.der == mykey.der

```
openssl rsa -in mykey.pem -outform DER -out mykey.der
```

Certificats X509

Certificat autosignat (genera cert i key)

```
# openssl req -new -x509 -nodes  -out servercert.pem  -keyout  serverkey.pem
# openssl req -new -x509 -out servercert.pem  -keyout  passfrasse.serverkey.pem
# openssl req -x509 -newkey rsa:2048 -keyout key.pem -out cert.pem
# openssl req -x509 -nodes -days 365 -sha256 \
                            -subj '/C=US/ST=Oregon/L=Portland/CN=www.madboa.com' \
                            -newkey rsa:2048 -keyout mycert.pem -out mycert.pem
```

Certificat autosignat usant una clau privada existent (cakey.pem)

```
# openssl req -new -x509    -days 365  -key cakey.pem    -out cacert.pem
```

Petició de certificat: Request

```
# openssl req -new -key serverkey.pem  -out serverreq.pem
```

CA Signar un request:/ Generar X509

```
# openssl x509 -CA cacert.pem   -CAkey cakey.pem  -req -in serverreq.pem  \
                      -out servercert.pem [ -CAcreateserial ]
# openssl x509 -CA cacert.pem   -CAkey cakey.pem  -req -in serverreq.pem \
                      -days 365   -extfile ca.conf    -CAcreateserial   -out servercert.pem
```

Definir extensions en un fitxer

```
# cat ca.conf
basicConstraints = critical,CA:FALSE
extendedKeyUsage = serverAuth,emailProtection
```

Llistar

```
# cat servercert.pem
# openssl x509 -noout   -text   -in servercert.pem
# openssl x509 -noout   -issuer    -subject  -purpose   -dates    -in servercert.pem
# openssl x509 -noout   -startdate   -enddate  -serial   -fingerprint    -fingerprint  \
                      -email   -hash   -issuer_hash     -subject_hash
```

Verificar

```
# openssl x509  -noout  -modulus  -in  servercert.pem    | openssl md5
# openssl rsa     -noout  -modulus  -in serverkey.pem  | openssl md5
```

Conversió de format PEM / DER

```
# openssl x509 -in cert.pem -inform PEM -out cert.der -outform DER
```

Convert a certificate to a certificate request:

```
# openssl x509 -x509toreq -in cert.pem -out req.pem -signkey key.pem
```

Convert a certificate request into a self signed certificate using extensions for a CA:

```
# openssl x509 -req -in careq.pem -extfile openssl.cnf -extensions v3_ca \
                  -signkey key.pem -out cacert.pem
```

Sign a certificate request using the CA certificate above and add user certificate extensions:

```
# openssl x509 -req -in req.pem -extfile openssl.cnf -extensions v3_usr \
                  -CA cacert.pem -CAkey key.pem -CAcreateserial
```

```
# openssl x509 -CA cacert.pem -CAkey cakey.pem -req -in serverreq.pem -out
servercert.pem -CAcreateserial   -extensions v3_ca
```

Set a certificate to be trusted for SSL client use and change set its alias to "Steve's Class 1 CA"

```
# openssl x509 -in cert.pem -addtrust clientAuth -setalias "Steve Class-1 CA" -out
trust.pem
```

Petició de certificació: Request

Petició de certificació

```
# openssl req -new -key serverkey.pem  -out serverreq.pem
```

Petició de certificació (generant clau privada)

```
# openssl req -newkey rsa:2048 -keyout key.pem -out req.pem
# openssl req -new -sha256 -newkey rsa:2048 -nodes \
             -subj '/CN=www.mydom.com/O=My Dom, Inc./C=US/ST=Oregon/L=Portland' \
             -keyout mykey.pem -out myreq.pemz
```

Llistar / verify

```
# openssl req -in req.pem -text -verify -noout
# openssl req -in myreq.pem -noout -verify -key mykey.pem
# openssl req -in req.pem -text  -noout
```

CA

Extret de la documentació M08-IOC-annexos

```
openssl ca  -keyfile private/cakey.pem -cert cacert.pem -in perereq.pem \
               -out perecert.pem -days 365 -config openssl.conf

openssl ca -in annareq.pem -out annacert.pem -config openssl.conf

# openssl ca  -in annareq.pem -out new2cert.pem -days 900 \
               -extensions v3_ca -config openssl.conf
# openssl ca -in req.pem -extensions v3_ca -out newcert.pem

# openssl ca -in annareq.pem -config openssl.conf
# openssl ca -in annareq.pem -config openssl.conf -extensions v3_ca

# openssl ca -in usuarireq.pem -config openssl.conf -policy policy_anything
```

Miscel·lània

Exemples extrets de ***Madboa***: https://www.madboa.com/geek/openssl/

Verify

```
$ openssl verify cert.pem
$ openssl verify remote.site.pem
$ openssl verify -CAfile cacert.pem servercert.pem
```

Client connection

```
openssl s_client -connect remote.host:25 -starttls smtp
openssl s_client -connect remote.host:465
openssl s_client -connect remote.host:25 -crlf -starttls smtp

openssl s_client -connect www.massivehost.com:443 -servername www.myhost.com
openssl s_client -connect remote.host:443
openssl s_client -connect remote.host:636
openssl s_client -connect remote.host:993
openssl s_client -connect remote.host:995
```

Server side

```
openssl s_server -cert mycert.pem -www
openssl s_server -accept 443 -cert mycert.pem -WWW
```

Digest

```
openssl dgst -md5 filename
openssl dgst -sha1 filename
openssl dgst -sha256 filenam
openssl list-message-digest-commands
```

Encription

```
openssl enc -base64 -in file.txt
openssl enc -base64 -in file.txt -out file.txt.enc
openssl list-cipher-commands

openssl enc -aes-256-cbc -salt -in file.txt -out file.enc
openssl enc -aes-256-cbc -a -salt -in file.txt -out file.enc

# decrypt
openssl enc -d -aes-256-cbc -in file.enc
openssl enc -d -aes-256-cbc -a -in file.enc

# provide password on command line
openssl enc -aes-256-cbc -salt -in file.txt  -out file.enc -pass pass:mySillyPassword

# provide password in a file
openssl enc -aes-256-cbc -salt -in file.txt -out file.enc -pass
file:/path/to/secret/password.txt
```

Passwords

```
$ openssl passwd MySecret
$ openssl passwd -salt 8E MySecret
$ openssl passwd -1 MySecret
$ openssl passwd -1 -salt sXiKzkus MySecret
```

Prime

```
$ openssl prime 119054759245460753
$ openssl prime -hex 2f
$ openssl prime -generate -bits 64
```

Random

```
openssl rand -base64 128
openssl rand -out random-data.bin 1024
```

S/MIME

```
openssl smime her-cert.pem -encrypt -in my-message.txt
openssl smime her-cert.pem -encrypt -des3 -in my-message.txt
openssl smime her-cert.pem \
  -encrypt \
  -des3 \
  -in my-message.txt \
  -from 'Your Fullname <you@youraddress.com>' \
  -to 'Her Fullname <her@heraddress.com>' \
  -subject 'My encrypted reply' |\
sendmail her@heraddress.com
```

```
Usar openssl_server amb un dels nostres certificats per fer de web
           a)  openssl s_server -cert cert.pem  -www -key key.pem -accept 8080
           b)  ncat --ssl  localhost 8080
               GET / HTTP/1.0

ídem exemple indicant on és el cakey:
       # openssl s_server -cert servercert.pem  -www -key serverkey.pem -accept 8080
       #  openssl s_client -CApath . -connect localhost:8080

Verificar amb openssl verify cert.pem (hem de tenir la ca carregada a etc)
       openssl verify cert.pem
       cert.pem: C = ca, ST = ca, L = Default City, O = Default Company Ltd, CN = e
       error 18 at 0 depth lookup:self signed certificate
       OK
       openssl verify -CAfile cacert.pem servercert.pem
       servercert.pem: OK
```

TSL/SSL Conexions segures (HTTPS)

Creació/Gestió de certificats digitals

Certificas digitals

Crear un certificat auto-signat per fer tests
Crear certificats per ser una pròpia CA.
Crear els certificats del servidor basats en una CA (pròpia o externa)
Afegir/modificar/eliminar una *passfrase* a una clau privada.

Crear certificats autosignats

Vàlid per a fer de CA i per ser un certificat de servidor autosignat (sense que calgui una altra CA).
Genera:

- autosigned.server.cert és el certificat.
- autosigned.server.key és la clau privada (“serverkey”)

La clau privada generada no conté *passfrase*, una frase de seguretat que es demana com un password per poder desxifrar el fitxer. Se li pot afegir/modificar.

```
# Generar el certificat + clau privada autosignats
# openssl req -new -x509 -nodes -out autosigned.server.crt -keyout autosigned.server.key
Generating a 2048 bit RSA private key
..+++
.......+++
writing new private key to 'autosigned.server.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:ca
State or Province Name (full name) []:Barcelona
Locality Name (eg, city) [Default City]:Barcelona
Organization Name (eg, company) [Default Company Ltd]:escola del treball de barcelona
Organizational Unit Name (eg, section) []:departament informatica
Common Name (eg, your name or your server's hostname) []:www.edt.org
Email Address []:admin@edt.org
# ll auto*
-rw-r--r-- 1 root root 1489 29 nov 16:28 autosigned.server.crt
-rw-r--r-- 1 root root 1704 29 nov 16:28 autosigned.server.key

# cat autosigned.server.crt
-----BEGIN CERTIFICATE-----
MIIEHTCCAwWgAwIBAgIJAMf0OqXXwvGYMA0GCSqGSIb3DQEBBQUAMIGkMQ
... output suprimit ...
PgCgnrTzCgSrMdWsvuFyaorcV6u9HaZoMDHkC5F4Bt76UbIZVo8F23s2Fhjl7Tjh
Sg==
-----END CERTIFICATE-----

# cat autosigned.server.key
-----BEGIN PRIVATE KEY-----
MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEAAoIBAQC8VYl8jRqW5Pdm
... output suprimit ...
sbuv4mqD0dQrZHFPPGzPn+g=
-----END PRIVATE KEY-----
```

```
# Afegir passfrase a la clau privada (generem un nou fitxer de clau privada)

# openssl rsa -des3 -in autosigned.server.key -out autosigned.passfrase.server.key
```

```
writing RSA key
Enter PEM pass phrase: serverkey
Verifying - Enter PEM pass phrase: serverkey

# cat autosigned.passfrase.server.key
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,159C3F65D3CECAE4

pZpIBwsjVZoM9w2ZHrhfTrW6bRyvG/yTu3+93E+M9Sord3+CipWR9c9lMdEZyxik
... output suprimit ...
SkiF9OkA+9S2rYNkcnuDt4GXs+afzkMWSIqRRkPCsXXoaJ0n8zjWyQ==
-----END RSA PRIVATE KEY-----
```

Crear una CA pròpia: Certificate Authority

Fer-ho manulment pas a pas:

- generar la clau privada (observar amb cat el contingut físic i amb openssl el lògic)
- generar el certificat x509 propi de la CA.

Usar els scripts ja preparats de openssl (CA.sh o CA.pl).

```
# Crear una entitat CA pròpia

# generar la clau privada, encriptada amb 3des i amb passfrase (format PEM)
# openssl genrsa -des3 -out ca.key 1024
Generating RSA private key, 1024 bit long modulus
...........++++++
.........................++++++
e is 65537 (0x10001)
Enter pass phrase for ca.key: cakey
Verifying - Enter pass phrase for ca.key:  cakey

# generar el certificat x509 pròpi de l'entitat CA (per a 365 dies) en format PEM
# openssl req -new -x509 -nodes -sha1 -days 365 -key ca.key -out ca.crt
Enter pass phrase for ca.key:  cakey
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:ca
```

```
State or Province Name (full name) []:Bercelona
Locality Name (eg, city) [Default City]:Barcelona
Organization Name (eg, company) [Default Company Ltd]:Veritat Absoluta
Organizational Unit Name (eg, section) []:Departament de certificats
Common Name (eg, your name or your server's hostname) []:VeritatAbsoluta
Email Address []:admin@edt.org

# ll
-rw-r--r-- 1 root root 1159 29 nov 17:40 ca.crt
-rw-r--r-- 1 root root  963 29 nov 17:24 ca.key
```

```
# Observar la clau privada de la CA

# mostrar el contingut físic
# cat ca.key
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,770703FF70C7B96F

dx25QunUljFCtQJrSJQAgAtbnpCLhtxkVtozRsDv6SjbwtFbshaxm6hms6tANSmg
... output suprimit ...
8yAB1+vj72huDV2r4PVgXouRJcxCDKjMlrbWhRjJEWqPSgLdNC7z3Q==
-----END RSA PRIVATE KEY-----

# mostrar el contingut lògic
# openssl rsa -noout -text -in ca.key
Enter pass phrase for ca.key:
Private-Key: (1024 bit)
modulus:
    00:de:1c:ec:6c:2e:bf:4d:b6:ca:8d:93:d3:9d:41:
    ... output suprimit ...
    c8:90:13:34:ba:31:d1:b3:f5
publicExponent: 65537 (0x10001)
privateExponent:
    7d:8e:8e:1b:4d:85:b8:f1:a6:a8:c7:b2:ed:07:8d:
    ... output suprimit ...
    0f:03:eb:ef:ed:45:ba:b5
prime1:
    00:f2:44:ed:97:c3:e2:9a:aa:95:ae:67:26:86:0f:
    ... output suprimit ...
    13:50:0d:e0:4b
prime2:
    00:ea:b3:8c:97:c6:a4:95:57:39:e0:de:74:f1:b3:
    ... output suprimit ...
    71:ad:e4:94:bf
```

```
exponent1:
   70:b0:87:23:94:c6:0e:d3:52:14:71:7e:85:d5:5a:
   ... output suprimit ...
   c8:8e:eb:c9
exponent2:
   00:91:af:dc:80:c6:3c:99:bb:28:61:4e:95:57:07:
   ... output suprimit ...
   e0:b3:e9:a4:ef
coefficient:
   5a:92:81:89:a7:83:52:b5:33:16:ed:79:0e:25:c7:
   ... output suprimit ...
   2a:a2:bf:df
```

```
# Observar el certificat x509 de la CA

# mostrar el contingut físic del certificat x509
# cat ca.crt
-----BEGIN CERTIFICATE-----
MIIDKjCCApOgAwIBAgIJANWdpn/8oUijMA0GCSqGSIb3DQEBBQUAMIGtMQswCQYD
... output suprimit ...
7zBltLVl0unEnClxY0jNhWkLdwPz/CKuDCIl6c8XAVCfJRHMhWpi8EGUi4GW2A==
-----END CERTIFICATE-----

# mostrar el contingut lògic del certificat x509
# openssl x509 -noout -text -in ca.crt
Certificate:
   Data:
      Version: 3 (0x2)
      Serial Number:
         d5:9d:a6:7f:fc:a1:48:a3
      Signature Algorithm: sha1WithRSAEncryption
       Issuer: C=ca, ST=Bercelona, L=Barcelona, O=Veritat Absoluta, OU=Departament de certificats, CN=VeritatAbsoluta/emailAddress=admin@edt.org
      Validity
         Not Before: Nov 29 16:40:57 2011 GMT
         Not After : Nov 28 16:40:57 2012 GMT
        Subject: C=ca, ST=Bercelona, L=Barcelona, O=Veritat Absoluta, OU=Departament de certificats, CN=VeritatAbsoluta/emailAddress=admin@edt.org
      Subject Public Key Info:
         Public Key Algorithm: rsaEncryption
            Public-Key: (1024 bit)
            Modulus:
               00:de:1c:ec:6c:2e:bf:4d:b6:ca:8d:93:d3:9d:41:
               ... output suprimit ...
               c8:90:13:34:ba:31:d1:b3:f5
            Exponent: 65537 (0x10001)
      X509v3 extensions:
         X509v3 Subject Key Identifier:
            35:7C:15:36:20:F3:B5:87:E2:C4:C8:71:5A:B2:87:16:7F:B8:13:63
         X509v3 Authority Key Identifier:
            keyid:35:7C:15:36:20:F3:B5:87:E2:C4:C8:71:5A:B2:87:16:7F:B8:13:63

         X509v3 Basic Constraints:
            CA:TRUE
   Signature Algorithm: sha1WithRSAEncryption
      33:39:de:3a:cc:c6:fd:74:a4:5e:40:cd:c9:33:f0:e7:27:32:
      ... output suprimit ...
      96:d8
```

Crear el certificat del servidor (real)

Crear una clau privada per el servidor (o per el servei web desitjat).

Crear una petició de certificat request per enviar a una CA:

- indicar les dades apropiades de qui som quan demanem el certificat.
- assegurar-se de que el CN (common name) és el de la seu web a usar el certificat.

La CA genera el certificat .crt signat per ella mateixa i l'envia al client.

- usar un fitxer de configuració de la CA que undiqui que els certificats a elaborar siguin de tipus “serverAuth”, és a dir, certificats de servidor.
- Es generarà un número de sèrie dels certificats que l'entitat de certificació CA va emetent.

“Et voilà” el servidor HTTP ja disposa d'un servificat que diu que “www.edt.org” és qui diu ser. Per tant si es fa la configuració SSL apropiada es podran fer connexions HTTPS.

```
# Crear una clau privada per al servidor
# és en format PEM, de 1024 bits i xifrada en 3DES. Utilitza passfrase
# podeu mirar a l'apartat “afegir/modificar/eliminar passfrases” si la voleu treure

# openssl genrsa -des3 -out server.key 1024
Generating RSA private key, 1024 bit long modulus
....................++++++
.........++++++
e is 65537 (0x10001)
Enter pass phrase for server.key: serverkey
Verifying - Enter pass phrase for server.key: serverkey

# Generar una petició de certificat request per enviar a l'entitat certificadora CA
# openssl req -new -key server.key -out server.csr
Enter pass phrase for server.key:
:wYou are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:ca
State or Province Name (full name) []:Barcelona
Locality Name (eg, city) [Default City]:Barcelona
Organization Name (eg, company) [Default Company Ltd]:escola del treball de barcelona
Organizational Unit Name (eg, section) []:departament d'informatica
Common Name (eg, your name or your server's hostname) []:www.edt.org
Email Address []:admin@edt.org

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:request password
An optional company name []:edt

# ll
-rw-r--r-- 1 root root  10 29 nov 17:51 key.txt
-rw-r--r-- 1 root root 830 29 nov 17:58 server.csr
-rw-r--r-- 1 root root 963 29 nov 17:50 server.key

# Observar la petició de certificat

# openssl req -noout -text -in server.csr
Certificate Request:
    Data:
        Version: 0 (0x0)
            Subject: C=ca, ST=Barcelona, L=Barcelona, O=escola del treball de barcelona,
OU=departament d'informatica, CN=www.edt.org/emailAddress=admin@edt.org
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (1024 bit)
                Modulus:
                    00:bc:6f:02:72:f2:f9:3f:19:62:2e:d8:46:61:46:
                    ... output suprimit ...
                    2c:6a:47:5b:db:99:14:28:af
                Exponent: 65537 (0x10001)
        Attributes:
            unstructuredName         :unable to print attribute
            challengePassword        :unable to print attribute
    Signature Algorithm: sha1WithRSAEncryption
        10:8d:61:05:7f:12:76:41:e4:d6:09:d4:fc:a6:56:be:36:fa:
         ... output suprimit ...
        ee:99
```

```
# Una entitat CA ha de signar la petició request de certificat i retornar un certificat .crt.
# en aquest cas com que som CA nosaltres mateixos generarem el certificat (com a “Veritat Absoluta”) del client (“www.edt.org”) que ha fet el request.

$ man x509
$ man ca

# Fitxer de configuració de la generació de certificats: indica què certifiquen
# cat ssl/ca/ca.conf
basicConstraints = critical,CA:FALSE
extendedKeyUsage = serverAuth,emailProtection

# L'autoritat CA ha de signar el certificat
# openssl x509 -CA ssl/ca/ca.crt -CAkey ssl/ca/ca.key -req -in ssl/server/server.csr -days 365 -sha1 -extfile ssl/ca/ca.conf  -CAcreateserial -out ssl/server/server.crt
Signature ok
subject=/C=ca/ST=Barcelona/L=Barcelona/O=escola del treball de barcelona/OU=departament d'informatica/CN=www.edt.org/emailAddress=admin@edt.org
Getting CA Private Key
Enter pass phrase for ssl/ca/ca.key: cakey

# Mostrar el nº de sèrie que genera la CA  per a cada certificat que emet.
# cat ssl/ca/ca.srl
F96F36F4897271FF
```

```
# L'entitat li enviarà al client el certificat generat: server.crt
# ll
-rw-r--r-- 1 root root 1184 29 nov 18:09 server.crt
-rw-r--r-- 1 root root  830 29 nov 17:58 server.csr
-rw-r--r-- 1 root root  963 29 nov 17:50 server.key

# El client que ha sol·licitat el certificat pot validar el certificat respecte la seva clau privada
# openssl x509 -noout -modulus -in ssl/server/server.crt | openssl md5
(stdin)= 3b5cc670b2312990f4e53efc37194108
# openssl rsa -noout -modulus -in ssl/server/server.key | openssl md5
Enter pass phrase for ssl/server/server.key: serverkey
(stdin)= 3b5cc670b2312990f4e53efc37194108
```

```
# També pot examinar el contingut del certificat per veure si és realment el seu
# openssl x509 -noout -text -in ssl/server/server.crt
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            f9:6f:36:f4:89:72:71:ff
        Signature Algorithm: sha1WithRSAEncryption
         Issuer: C=ca, ST=Bercelona, L=Barcelona, O=Veritat Absoluta, OU=Departament de certificats, CN=VeritatAbsoluta/emailAddress=admin@edt.org
        Validity
            Not Before: Nov 30 20:24:15 2011 GMT
            Not After : Nov 29 20:24:15 2012 GMT
             Subject: C=ca, ST=Barcelona, L=Barcelona, O=escola del treball de barcelona, OU=departament d'informatica, CN=www.edt.org/emailAddress=admin@edt.org
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (1024 bit)
                Modulus:
                    00:bc:6f:02:72:f2:f9:3f:19:62:2e:d8:46:61:46:
                    ... output suprimit ...
                    2c:6a:47:5b:db:99:14:28:af
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints: critical
                CA:FALSE
            X509v3 Extended Key Usage:
                TLS Web Server Authentication, E-mail Protection
    Signature Algorithm: sha1WithRSAEncryption
        4b:d1:73:d4:56:9b:5e:05:27:75:56:34:49:7d:c5:5f:7c:7d:
        ... output suprimit ...
        08:6e
```

Afegir/modificar/eliminar una passfrase de la clau privada

Afegir a la clau <nom>.key per disposar de seguretat a la clau privada. Sense la passfrase ningú podrà utilitzar la clau privada. Cal la passfrase per desxifrar la clau privada per poder-la usar.

Inconvenient: en engegar Apache demanarà la passfrase necessària per a cada certificat de servidor que en tingui una.

Avantatge: seguretat de la clau privada. Si algú la pot obtenir es pot fer passar per nosaltres.

Accions a saber fer:

- afegir una passfrase a una clau privada que no en té: genera una nova key.

- eliminar una passfrase d'una clau privada que ja en té: genera una nova key no xifrada (perill!).
- modificar una passfrase d'una clau provada que ja en té una: genera una nova key.

```
# Afegir passfrase a la clau privada (generem un nou fitxer de clau privada)

# openssl rsa -des3 -in server.key -out passfrase.server.key
writing RSA key
Enter PEM pass phrase: serverkey
Verifying - Enter PEM pass phrase: serverkey
## mv passfrase.server.key server.key

# Modificar la passfrase existent

# openssl rsa -des3 -in passfrase.server.key -out passfrase.new.server.key
Enter pass phrase for passfrase.server.key:
writing RSA key
Enter PEM pass phrase: serverkey
Verifying - Enter PEM pass phrase: newserverkey
## mv passfrase.new.server.key passfrase.server.key

# Eliminar la passfrase d'una clau privada

# openssl rsa -in passfrase.server.key -out deleted-passfrase.server.key
Enter pass phrase for autosigned.passfrase.server.key: serverkey
writing RSA key
## mv deleted-passfrase.server.key server-key

# Llistat de tot el que s'ha anat generant
# ll
-rw-r--r-- 1 root root 1675 29 nov 16:55 deleted-passfrase.server.key
-rw-r--r-- 1 root root 1743 29 nov 16:48 passfrase.new.server.key
-rw-r--r-- 1 root root 1743 29 nov 16:37 passfrase.server.key
-rw-r--r-- 1 root root 1489 29 nov 16:28 server.crt
-rw-r--r-- 1 root root 1704 29 nov 16:28 server.key
```

Examinar els contingauts de certificats i claus privades

Examinar el contingut de certificats.
Examinar el contingut de claus privades.
Verificar si corresponen com a parella “certificat / clau-privada”

```
# Examinar el contingut de certificats:

# openssl x509 -noout -text -in autosigned.server.crt
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            c7:f4:3a:a5:d7:c2:f1:98
        Signature Algorithm: sha1WithRSAEncryption
               Issuer: C=ca, ST=Barcelona, L=Barcelona, O=escola del treball de barcelona,
OU=depaca, CN=www.edt.org/emailAddress=admin@edt.org
        Validity
            Not Before: Nov 29 15:28:02 2011 GMT
            Not After : Dec 29 15:28:02 2011 GMT
             Subject: C=ca, ST=Barcelona, L=Barcelona, O=escola del treball de barcelona,
OU=depaca, CN=www.edt.org/emailAddress=admin@edt.org
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:bc:55:89:7c:8d:1a:96:e4:f7:66:91:87:e9:63:
                    ... output suprimit ...
                    86:35
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                3F:3A:CC:C3:50:4C:28:89:B4:07:76:B3:3A:45:C9:40:63:40:E1:12
            X509v3 Authority Key Identifier:
                keyid:3F:3A:CC:C3:50:4C:28:89:B4:07:76:B3:3A:45:C9:40:63:40:E1:12

            X509v3 Basic Constraints:
                CA:TRUE
    Signature Algorithm: sha1WithRSAEncryption
        32:fd:29:72:57:81:ff:ae:55:d9:46:87:df:3b:31:8c:27:12:
        ... output suprimit ...
        ed:38:e1:4a
```

```
# Mostrar el contingut de la clau privada

# openssl rsa -noout -text -in autosigned.server.key
Private-Key: (2048 bit)
modulus:
    00:bc:55:89:7c:8d:1a:96:e4:f7:66:91:87:e9:63:
    ... output suprimit ...
    86:35
publicExponent: 65537 (0x10001)
privateExponent:
    40:3a:33:8f:04:58:03:09:c6:cd:75:e8:11:d1:b3:
    ... output suprimit ...
    41
prime1:
    00:f5:7b:53:1f:8e:53:d5:e0:0c:19:2c:25:91:a5:
    ... output suprimit ...
    53:8e:50:bd:1e:7e:72:e9:a9
prime2:
    00:c4:67:5d:0c:aa:76:c3:35:3a:e0:c8:96:f4:f9:
    ... output suprimit ...
    d6:a6:17:09:bd:9f:b4:07:ad
exponent1:
    49:24:68:bd:03:44:59:7a:7b:40:58:d6:0c:d2:83:
    ... output suprimit ...
    71:2d:ff:5b:81:a3:ad:99
exponent2:
    00:83:6f:70:d3:d3:18:1b:56:fa:0a:07:f3:0e:0a:
    ... output suprimit ...
    88:de:29:b8:b9:0f:b1:59:19
coefficient:
    5f:44:60:85:5c:44:41:92:91:da:c2:c4:70:d8:ed:
    ... output suprimit ...
    64:71:4f:3c:6c:cf:9f:e8
```

```
# Verificar que el certificat i la clau-privada són conjuntats, es corresponen

# openssl x509 -noout -modulus -in autosigned.server.crt | openssl md5
(stdin)= db5c2f5add8d40d76b9ce4b962d94ab8
# openssl rsa -noout -modulus -in autosigned.server.key | openssl md5
(stdin)= db5c2f5add8d40d76b9ce4b962d94ab8
```

Estructura de directoris usada en els exemples

```
# Des d'un directori de proves (/tmp/ssl) s'ha generat:

# tree
.
├── autosigned
```

```
│   ├── autosigned.deleted-passfrase.server.key
│   ├── autosigned.passfrase.new.server.key
│   ├── autosigned.passfrase.server.key
│   ├── autosigned.server.crt
│   ├── autosigned.server.key
│   ├── dn.txt
│   └── key.txt
├── ca
│   ├── ca.conf
│   ├── ca.crt
│   ├── ca.exemple.conf
│   ├── ca.key
│   ├── ca.srl
│   ├── dn.txt
│   └── key.txt
└── server
    ├── key.txt
    ├── server.crt
    ├── server.csr
    └── server.key
```

Ordre Openssl CA

Actuar com a CA amb l'ordre openssl CA
Llistar /etc/PKI/CA
Generar el fitxer index.txt
Generar el fitxer de serial (amb valor 01)

```
# si cal fer:
  touch /etc/pki/CA/index.txt
  echo "01" > /etc/pki/CA/serial

openssl ca  -keyfile private/cakey.pem -cert cacert.pem -in perereq.pem \
              -out perecert.pem -days 365 -config openssl.conf

openssl ca -in annareq.pem -out annacert.pem -config openssl.conf
```

Configuració Openssl

Consulteu la documentació de:

- ❏ Apunts de m11: doc_m08_ioc_correu_annexos (Annex global dels apunts IOC de correu) https://sites.google.com/site/asixm11edt/home/uf1nf1_filesystem_security/doc_m08_ioc_correu_annexos.pdf?attredirects=0&d=1)

❏ man x509
❏ man x509v3_config
❏ man openssl.conf

Explicació del fitxer /etc/pki/tls/openssl.cnf

- fer-ne una còpia al directori de treball i personalitzar la còpia.
- practicar diferents policies (match, anything, crear-ne una). Establir la que és per defecte / indicar-la a la línia de comandes.
- Opcions de req i de reg-distinguished-name. Generar noves opcions amb valors predeterminats
- Extensions: usar extensions ja definides: v3_ca , usr_cert, definit per nosaltres. Indicar-ho al fitxer de configuració o passar-ho com a argument.
- Extensions: generar un fitxer de configuració amb les constraints / extensions a usar.

```
# openssl ca  -in annareq.pem -out new2cert.pem -days 900 \
                  -extensions v3_ca -config openssl.conf
# openssl ca -in req.pem -extensions v3_ca -out newcert.pem

# openssl ca -in annareq.pem -config openssl.conf
# openssl ca -in annareq.pem -config openssl.conf -extensions v3_ca

# openssl ca -in usuarireq.pem -config openssl.conf -policy policy_anything
```

Tràfic segur amb TLS/SSL i STARTTLS

TLS/SSL i STARTTLS

Tràfic de dades ‘data in motion’

❏ tràfic client servidor segur en un medi insegur i sense secret compartit
❏ criptografia asimètrica clau privada / clau pública
❏ certificats per validar la identitat, usualment del servidor i opcionalment del client.
❏ algorisme diffie-hellman per generar un secret compartit.
❏ tràfic ‘in motion’ amb criptografia simètrica, secret compartit generat amb criptografia asimètrica.
❏ observar el handsake de SSL/TLS i la negociació de paràmetres.

Ports:

❏ ports ‘trafic pla’ per exemple 80, 110, 143, 386
❏ ports privilegiats segurs 443, 995 993, 636
❏ connexions segures a ports no segurs usant STARTTLS.

Exemples d’aplicació TLS / SSL / StartTLS

HTTPS: Accés web segur

Objectiu de la pràctica:

- Funcionament dels certificats web de servidor autosignats.
- Funcionament dels certificats d’entitat.

Pràctica amb Firefox

- Engegar el docker ***edtasixm11/https*** i configurar el */etc/hosts* del docker i del host apuntant a la ip del docker les dues seus web virtuals www.m11.cat i www.admin.cat.

- Engegar el servei amb la utilitat ***httpon*** (és un àlias disponible per root). Verificar amb l’ordre ***httpd -S*** que les seus estan en marxa.

- Autosignat:
  Connectar amb ***firefox*** via https a una web amb un certificat autosignat: ***https://www.m11.cat***. Observar el certificat. Acceptar el certificat. Següents connexions ja sense excepció de seguretat. Eliminar el certificat de la llista de certificats de servidor. En tornar a connectar es torna a produir l’excepció.

- Servidor amb certificat de CA:
  Connectar amb el ***firefox*** a la web ***https://www.admin.cat*** que disposa d’un certificat de servidor expedit per l’entitat Veritat Absoluta. Es genera una excepció de seguretat. Importar el certificat de servidor. Observar que ja no es genera l’excepció. Eliminar el certificat de servidor i de nou es genera l’excepció de seguretat.

- Entitat CA:
  Es vol incorporar el certificat de la CA al navegador firefox. D’aquesta manera un cop incorporat qualsevol accés a la web ***https://www.admin.cat*** serà validat automàticament.

  Per incorporar el certificat farem un ***trick***, en el container fer ***cat /var/www/certs/cacert.pem*** i seleccionar-ho amb el mouse. En el host crear un fitxer nou amb ***vim cacert.pem*** i copiar-hi el contingut (acabem de copiar textualment un certificat!).

  Al ***firefox*** importar a la pestanya d’***Entitats*** el certificat cacert.pem. Observar que apareix una nova entitat anomenada Veritat Absoluta. Ara en accedir a

https://www.admin.cat ja no es genera l’excepció de seguretat.

Pràctica amb s_client

- Des d’una consola del host connectar amb openssl s_client al servidors web segur. Per exemple a les seus virtuals del docker de l’exercici anterior.

```
[root@hp01 m11]# openssl  s_client -connect 172.17.0.2:443
CONNECTED(00000003)
depth=0 C = ca, ST = barcelona, L = barcelona, O = escola del treball de barcelona, OU
= informatica, CN = www.m11.cat, emailAddress = admin@edt.cat
verify error:num=18:self signed certificate
verify return:1
depth=0 C = ca, ST = barcelona, L = barcelona, O = escola del treball de barcelona, OU
= informatica, CN = www.m11.cat, emailAddress = admin@edt.cat
verify error:num=10:certificate has expired
notAfter=Apr 14 18:45:58 2016 GMT
verify return:1
depth=0 C = ca, ST = barcelona, L = barcelona, O = escola del treball de barcelona, OU
= informatica, CN = www.m11.cat, emailAddress = admin@edt.cat
notAfter=Apr 14 18:45:58 2016 GMT
verify return:1
---
Certificate chain
 0 s:/C=ca/ST=barcelona/L=barcelona/O=escola del treball de
barcelona/OU=informatica/CN=www.m11.cat/emailAddress=admin@edt.cat
   i:/C=ca/ST=barcelona/L=barcelona/O=escola del treball de
barcelona/OU=informatica/CN=www.m11.cat/emailAddress=admin@edt.cat
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIEJzCCAw+gAwIBAgIJAJomYm95Fsx1MA0GCSqGSIb3DQEBCwUAMIGpMQswCQYD
VQQGEwJjYTESMBAGA1UECAwJYmFyY2Vsb25hMRIwEAYDVQQHDAliYXJjZWxvbmEx
KDAmBgNVBAoMH2VzY29sYSBkZWwgdHJlYmFsbCBkZSBiYXJjZWxvbmExFDASBgNV
BAsMC2luZm9ybWF0aWNhMRQwEgYDVQQDDAt3d3cubTExLmNhdDEcMBoGCSqGSIb3
DQEJARYNYWRtaW5AZWR0LmNhdDAeFw0xNjAzMTUxODQ1NThaFw0xNjA0MTQxODQ1
NThaMIGpMQswCQYDVQQGEwJjYTESMBAGA1UECAwJYmFyY2Vsb25hMRIwEAYDVQQH
DAliYXJjZWxvbmExKDAmBgNVBAoMH2VzY29sYSBkZWwgdHJlYmFsbCBkZSBiYXJj
ZWxvbmExFDASBgNVBAsMC2luZm9ybWF0aWNhMRQwEgYDVQQDDAt3d3cubTExLmNh
dDEcMBoGCSqGSIb3DQEJARYNYWRtaW5AZWR0LmNhdDCCASIwDQYJKoZIhvcNAQEB
BQADggEPADCCAQoCggEBAMX1deS6NR/SQ1ukCi89Ikc9mVQO5GeRV3ASBtrsSL2T
cjdbaRgWEunwSXxwKBnPO3H3ViZYEqUFayy44FLyE4gz+lFo1eoLD7SofXuxU1MQ
DqU7a3NLsmyKEQpnbsVx/BkPJ09dWPfwN/OeKlNqe81V91Kaj1L4babhIZD3kXlk
hb1SNOKFqGXDXzkLrgKnWvlAW2mjoP+F56QwbaFVHxNXnZWj7HLWsdg0HH2AyXDk
EoUblmY/ud+7Yfo9b9cQ7CpY0/StinhAaBWvHEWnOPR/5UjFy/XGx6zFqmt5ETf3
bd4iipA+XO+KCL2dw3s6NkwzjXU9HmdaiD1w+puH5cECAwEAAaNQME4wHQYDVR0O
BBYEFHyaR2tzMAeAT/MYHeCFcYenlSMnMB8GA1UdIwQYMBaAFHyaR2tzMAeAT/MY
HeCFcYenlSMnMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQELBQADggEBAAS+Kxbo
3L0TKU/vGDMXMKH3938iw14PAKd++7IVQK17l7wP8AHmuDlWbBqfLJNbbfgNLxKw
li8ZjqndfDwJMf9Ht8KaRCKAzQ+iIfHihTdMwzIbI2VoMOH8A6WiQOOZKfg6xL+z
CZ5dGJ6GyfiwHl2VLGE7Mp+E7V9vZyKpl6iei/Nz/lVBkpvyFI8IHDuUaE4n+73u
```

```
MxvFLtNoofBI175ActxNc5EFse2rAaP3lhC/wX/PdXxAIepqleNi+8mBDhp0JHCk
O6IvET7VGZPydAdRkhuRzSdUUmLdKmIyMlyI2FRkhMJZsQbUeOKKzA1dQHXbWzw3
sOSHlaGtguIzAE8=
-----END CERTIFICATE-----
subject=/C=ca/ST=barcelona/L=barcelona/O=escola del treball de
barcelona/OU=informatica/CN=www.m11.cat/emailAddress=admin@edt.cat
issuer=/C=ca/ST=barcelona/L=barcelona/O=escola del treball de
barcelona/OU=informatica/CN=www.m11.cat/emailAddress=admin@edt.cat
---
No client certificate CA names sent
Server Temp Key: ECDH, prime256v1, 256 bits
---
SSL handshake has read 1758 bytes and written 333 bytes
---
New, TLSv1/SSLv3, Cipher is ECDHE-RSA-AES128-GCM-SHA256
Server public key is 2048 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
SSL-Session:
        Protocol  : TLSv1.2
        Cipher: ECDHE-RSA-AES128-GCM-SHA256
        Session-ID:
F1F8B2CA0CA89AE39D66D9B6971A8071FECBB2A4851E088070F35E8994D273E7
        Session-ID-ctx:
        Master-Key:
D680940AD8ADA21123C402995AC4391DD6DB8FC56519970C0AC34498AF65A2FA7
DBBBFE6F2F776C20CEECD9222FF6F91
        Key-Arg   : None
        Krb5 Principal: None
        PSK identity: None
        PSK identity hint: None
        TLS session ticket lifetime hint: 300 (seconds)
        TLS session ticket:
        0000 - c2 c1 94 df d9 7c bf f6-34 fc 29 4c 37 c0 0a 7e   .....|..4.)L7..~
        0010 - a7 5e 89 00 56 c9 04 d1-55 a9 80 11 15 ae 42 c4   .^..V...U.....B.
        0020 - b2 6f d6 f0 a0 02 7d bb-7c c3 24 0f ba 4a 39 9d   .o....}.|.$..J9.
        0030 - 7e 67 31 ee 7f 7e f0 5d-ef c2 15 a6 a7 2c 31 45   ~g1..~.].....,1E
        0040 - ae 1f 8b e2 3d f0 84 5a-72 fc 52 64 89 ec e4 61   ....=..Zr.Rd...a
        0050 - 1b a7 d4 9e a9 aa b9 06-07 56 3f bd ad a8 5c 6e   .........V?...\n
        0060 - c5 34 17 49 b6 b2 81 8b-e6 2d 20 44 63 3b 1a a7   .4.I.....- Dc;..
        0070 - 98 68 99 a1 cf fb 17 e3-14 0b 58 a9 a4 df a9 82   .h........X.....
        0080 - 6e af d6 ad 1e c8 b4 17-f9 a0 d2 3b e0 a0 fd 9f   n.........;....
        0090 - 52 18 9d d0 eb 56 48 fe-f5 39 60 a7 5a 5d b0 fb   R....VH..9`.Z]..
        00a0 - fd a8 bf a6 fd 36 e2 06-1e 37 f2 86 75 29 3f 2d   .....6...7..u)?-
        00b0 - cc eb 98 ab d4 d1 1f 06-ef a8 65 27 58 f6 2d eb   ..........e'X.-.
        Start Time: 1490557377
        Timeout   : 300 (sec)
        Verify return code: 10 (certificate has expired)
---
GET / HTTP/1.1
Host: www.m11.cat

HTTP/1.1 200 OK
Date: Sun, 26 Mar 2017 19:43:05 GMT
Server: Apache/2.4.17 (Fedora) OpenSSL/1.0.1k-fips
Last-Modified: Tue, 15 Mar 2016 18:25:53 GMT
ETag: "76-52e1a87078e40"
Accept-Ranges: bytes
Content-Length: 118
Content-Type: text/html; charset=UTF-8

<html>
<title>my first https page</title>
<body>
  <h1> TLS / SSL </h1>
  My Test site - $(hostname)
</body>
</html>

closed
```

```
…
---
GET / HTTP/1.1
Host: www.admin.cat

HTTP/1.1 200 OK
Date: Sun, 26 Mar 2017 19:48:33 GMT
Server: Apache/2.4.17 (Fedora) OpenSSL/1.0.1k-fips
Last-Modified: Tue, 15 Mar 2016 19:01:51 GMT
ETag: "7d-52e1b07a805c0"
Accept-Ranges: bytes
Content-Length: 125
Content-Type: text/html; charset=UTF-8

<html>
<title>my second https page</title>
<body>
  <h1> CA - TLS / SSL </h1>
  Test using Veritat Absoluta
```

```
</body>
</html>
```

Pràctica amb curl

```
[root@hp01 m11]# curl -v -ssl http://www.m11.cat
<html>
<title>my first https page</title>
<body>
  <h1> TLS / SSL </h1>
  My Test site - $(hostname)
</body>
</html>

[root@hp01 m11]# curl -v -ssl https://www.m11.cat
```

Pràctica amb gmail / POP

https://support.google.com/mail/answer/7104828?authuser=1&hl=en&authuser=1&visit_id=1-636261549854277284-763462414&rd=1

```
[root@hp01 m11]# openssl  s_client -connect pop.gmail.com:995
CONNECTED(00000003)
depth=3 C = US, O = Equifax, OU = Equifax Secure Certificate Authority
verify return:1
depth=2 C = US, O = GeoTrust Inc., CN = GeoTrust Global CA
verify return:1
depth=1 C = US, O = Google Inc, CN = Google Internet Authority G2
verify return:1
depth=0 C = US, ST = California, L = Mountain View, O = Google Inc, CN =
pop.gmail.com
verify return:1
---
Certificate chain
 0 s:/C=US/ST=California/L=Mountain View/O=Google Inc/CN=pop.gmail.com
   i:/C=US/O=Google Inc/CN=Google Internet Authority G2
 1 s:/C=US/O=Google Inc/CN=Google Internet Authority G2
   i:/C=US/O=GeoTrust Inc./CN=GeoTrust Global CA
 2 s:/C=US/O=GeoTrust Inc./CN=GeoTrust Global CA
   i:/C=US/O=Equifax/OU=Equifax Secure Certificate Authority
---
```

```
Server certificate
-----BEGIN CERTIFICATE-----
MIIEfjCCA2agAwIBAgIIPR/udz0XEV8wDQYJKoZIhvcNAQELBQAwSTELMAkGA1UE
BhMCVVMxEzARBgNVBAoTCkdvb2dsZSBJbmMxJTAjBgNVBAMTHEdvb2dsZSBJbnRl
cm5ldCBBdXRob3JpdHkgRzIwHhcNMTcwMzE2MDg1NTU0WhcNMTcwNjA4MDg1NDAw
WjBnMQswCQYDVQQGEwJVUzETMBEGA1UECAwKQ2FsaWZvcm5pYTEWMBQGA1UEBwwN
TW91bnRhaW4gVmlldzETMBEGA1UECgwKR29vZ2xlIEluYzEWMBQGA1UEAwwNcG9w
LmdtYWlsLmNvbTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBALf69xkP
njg9zKktauDvgGBUX5d4iTOgupseSH+dUNC+n6mLirhBXXYTGKZikUPs2YoSIkdX
vZqF1ZwFYtPxOAUn/Pmywb595BoSwR8Lle3I+MRZd3sMPelmrD9bafBd6WBDQTf4
YwvsjQsJpH8XR8FiRRCJfRHdGqbkcXvgHvvpZcJX6XZTCmQGoDaA52zhfdVo1zBY
icuV9eqsQW7lMpxskf7Emm7vrOpr3RFP1PVlLQ/XQJzOHCtewkOHZl4H0nkd0MfP
QFuulB1WS7sHzHP4C6hTWdZ/ae2rrnnriLrDKbLqdsf31XzAS2Bz5eX7sBV3WsuV
AOaASC13MxilsrUCAwEAAaOCAUowggFGMB0GA1UdJQQWMBQGCCsGAQUFBwMBBggr
BgEFBQcDAjAYBgNVHREEETAPgg1wb3AuZ21haWwuY29tMGgGCCsGAQUFBwEBBFww
WjArBggrBgEFBQcwAoYfaHR0cDovL3BraS5nb29nbGUuY29tL0dJQUcyLmNydDAr
BggrBgEFBQcwAYYfaHR0cDovL2NsaWVudHMxLmdvb2dsZS5jb20vb2NzcDAdBgNV
HQ4EFgQUGCtFVo1P9r2B64VWflrlhiLo2+QwDAYDVR0TAQH/BAIwADAfBgNVHSME
GDAWgBRK3QYWG7z2aLV29YG2u2IauIqBLzAhBgNVHSAEGjAYMAwGCisGAQQB1nkC
BQEwCAYGZ4EMAQICMDAGA1UdHwQpMCcwJaAjoCGGH2h0dHA6Ly9wa2kuZ29vZ2xl
LmNvbS9HSUFHMi5jcmwwDQYJKoZIhvcNAQELBQADggEBAFuNfOwmWQuPf+apVpXa
/U3DsIIYQaKd+S6DWsefx2GXTZyxfR6inNe1hP0SEYmBdcqQNO8DZbjQWor0nbmY
L5J2v0l0lwfJ6Ey4gvQOPBAPA8FY9QebrCnVMWY3wssLrogkYytosA/ayQj7xec/
v+Q1kG3PGIDk3+qEZnclyLj17k7FdX0gRj2yOrdV23xVgVDUuMSRgHyEZ8M+u0MO
v7/Ba5cBAy06FwB9D9KgoG5MmFCSpO8ScSYt8ghz/r720HG3M/X5+Nb+lEqfOHMO
DuJ1FcVEt/OcSSEULjnuc7/uqJeCs1tW/jt5aY7QeFdPPh/68wK/RYuwpn4I8KgL
g2Y=
-----END CERTIFICATE-----
subject=/C=US/ST=California/L=Mountain View/O=Google Inc/CN=pop.gmail.com
issuer=/C=US/O=Google Inc/CN=Google Internet Authority G2
---
No client certificate CA names sent
Server Temp Key: ECDH, prime256v1, 256 bits
---
SSL handshake has read 3725 bytes and written 333 bytes
---
New, TLSv1/SSLv3, Cipher is ECDHE-RSA-AES128-GCM-SHA256
Server public key is 2048 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
SSL-Session:
        Protocol  : TLSv1.2
        Cipher: ECDHE-RSA-AES128-GCM-SHA256
        Session-ID:
F612D003A779872AD2FCDE148A25728689B1049E0811B7FD68844AFC1E7A8241
        Session-ID-ctx:
        Master-Key:
389D144DE177D986BE4781489616B926A7C614CC17D74DDE552653FFBD56308D5
97601C6CBE19EEEF35605CB54F0A8A3
        Key-Arg   : None
        Krb5 Principal: None
        PSK identity: None
        PSK identity hint: None
```

```
    TLS session ticket lifetime hint: 100800 (seconds)
    TLS session ticket:
    0000 - 28 4a 56 30 01 fb b2 a0-ab 76 06 d6 46 5e e3 7b   (JV0.....v..F^.{
    0010 - d0 4e 0f 5c 40 2e 88 82-15 49 b5 34 ab fc 66 c2   .N.\@....I.4..f.
    0020 - 09 57 ae 36 09 f8 fc 62-cb d4 de 27 61 4a 04 fa   .W.6...b...'aJ..
    0030 - 73 f9 7d 86 ed 0d 4d 92-f0 86 cf 0f eb 62 df 76   s.}...M......b.v
    0040 - 42 f2 78 01 a1 59 4f 14-4e af 3f 67 43 9f a0 6f   B.x..YO.N.?gC..o
    0050 - 8d 4a 51 3f ea 8d 6c 97-80 d0 84 d7 1e 3d 9e 4f   .JQ?..l......=.O
    0060 - ea dd c6 32 d0 46 77 a5-95 b6 df 26 97 87 33 34   ...2.Fw....&..34
    0070 - 8c 39 42 a0 15 cf 57 19-e7 0c 39 5f f6 79 12 ad   .9B...W...9_.y..
    0080 - e1 ec 34 d2 f8 04 36 f3-e7 7c 8b 84 e6 06 6c a3   ..4...6..|....l.
    0090 - c3 1c 28 22 49 d5 01 5f-90 4d 36 51 c9 86 74 89   ..("I.._.M6Q..t.
    00a0 - 3b 7c 0e 93                                       ;|..

    Start Time: 1490558954
    Timeout   : 300 (sec)
    Verify return code: 0 (ok)
---
+OK Gpop ready for requests from 88.3.81.73 64mb28206634ljj
USER edtasixm14
+OK send PASS
PASS xxxxx
+OK Welcome.
STAT
+OK 8 87440
LIST
+OK 8 messages (87440 bytes)
1 6928
2 7758
3 4844
4 5364
5 5071
6 35142
7 12120
8 10213
.
QUIT
DONE
```

trick aprofiteu per fer cut&paste del certificat de google i examineu-lo amb openssl

LDAPS: Accés segur al servidor LDAP

HowTo ASIX Certificats Digitals

Objectius:

- Usant el docker ***edtasixm06/ldapserver*** afegir-li certificats digitals per permetre connexions segures ***ldaps*** amb TLS/SSL.
- Des de qualsevol host client realitzar consultes LDAP segures al port de ***ldaps***.
- Configurar LDAP per acceptar connexions segures al propi port ***ldap*** realitzant ***StartTLS***.
- Des de qualsevol host client realitzar consultes ldap segures al port de ***ldap***.

Server:

```
# cat /opt/docker/slapd-tls.conf
#
# See slapd.conf(5) for details on configuration options.
# This file should NOT be world readable.
include         /etc/openldap/schema/corba.schema
include         /etc/openldap/schema/core.schema
include         /etc/openldap/schema/cosine.schema
include         /etc/openldap/schema/duaconf.schema
include         /etc/openldap/schema/dyngroup.schema
include         /etc/openldap/schema/inetorgperson.schema
include         /etc/openldap/schema/java.schema
include         /etc/openldap/schema/misc.schema
include         /etc/openldap/schema/nis.schema
include         /etc/openldap/schema/openldap.schema
include         /etc/openldap/schema/ppolicy.schema
include         /etc/openldap/schema/collective.schema
include     /etc/openldap/schema/samba.schema


# Allow LDAPv2 client connections.  This is NOT the default.
allow bind_v2
pidfile         /var/run/openldap/slapd.pid
TLSCACertificateFile        /etc/openldap/certs/ca.crt
TLSCertificateFile          /etc/openldap/certs/server.crt
TLSCertificateKeyFile       /etc/openldap/certs/server.key
TLSVerifyClient     never
TLSCipherSuite      HIGH:MEDIUM:LOW:+SSLv2

# ----------------database {0} config ----------------------------------
database config
rootdn "cn=Sysadmin,cn=config"
rootpw syskey
# ----------------------------------------------------------------------

# ----------------database {1} edt.org ----------------------------------
database bdb
suffix "dc=edt,dc=org"
rootdn "cn=Manager,dc=edt,dc=org"
rootpw secret
directory /var/lib/ldap
#index objectClass eq, press
access to * by self write by * read
# ----------------------------------------------------------------------

# ----------------enable monitoring ------------------------------------
database monitor
access to * by dn.exact="cn=Manager,dc=edt,dc=org" read by * none
# ---------------- end database monitor --------------------------------
```

Client:

```
# cat /etc/openldap/ldap.conf
#
# LDAP Defaults
#

# See ldap.conf(5) for details
# This file should be world readable but not world writable.

#BASE   dc=example,dc=com
#URI    ldap://ldap.example.com ldap://ldap-master.example.com:666
BASE dc=edt,dc=org
URI ldaps://ldap.edt.org

#SIZELIMIT    12
#TIMELIMIT    15
#DEREF        never

#TLS_CACERTDIR    /etc/openldap/certs
TLS_CACERT /etc/openldap/certs/ca.crt

# Turning this off breaks GSSAPI used with krb5 when rdns = false
SASL_NOCANON    on
```

Search amb debug usant ldaps (TLS):

```
$ ldapsearch  -LLL -x -H ldaps://172.17.0.2 -b 'dc=edt,dc=org' -d-1
          > /tmp/out.out     2> /tmp/err.out
```

Usant StartTLS:

```
$ ldapsearch  -LLL -x -Z -H ldap://172.17.0.2 -b 'dc=edt,dc=org' 'cn=pere*'
```

Problemes? Mirem que pot estar passant:

1) En el client no s'ha definit apropiadament el cacert i per tant en connectar no pot verificar el issuer del certificat que el client rep del servidor. Cal assegurar-se de copiar el certificat de la ca al directori on s'ha indicat en la directiva client: TLS_CACERT /etc/openldap/certs/ca.crt

```
$ ldapsearch -x -LLL -Z -b 'dc=edt,dc=org' -h 172.17.0.2 dn
ldap_start_tls: Connect error (-11)
   additional info: TLS error -8172:Peer's certificate issuer has been marked as not trusted by the user.
ldap_result: Can't contact LDAP server (-1)
```

```
$ ldapsearch -d1 -x -LLL -Z -b 'dc=edt,dc=org' -h 172.17.0.2 dn  | less
ldap_create
ldap_url_parse_ext(ldap://172.17.0.2)
...
```

2) No estem cridant al servidor amb el nom de host FQDN apropiat i no fa match amb el que identifica el certificat:

```
$ ldapsearch -vx -LLL -Z -b 'dc=edt,dc=org' -h 172.17.0.2 dn
ldap_initialize( ldap://172.17.0.2 )
ldap_start_tls: Connect error (-11)
   additional info: TLS error -8157:Certificate extension not found.
ldap_result: Can't contact LDAP server (-1)
```

```
]$ ldapsearch -d1 -x -LLL -Z -b 'dc=edt,dc=org' -h 172.17.0.2 dn
ldap_create
ldap_url_parse_ext(ldap://172.17.0.2)
...
TLS: loaded CA certificate file /var/tmp/m11/tls18/tls18:ldaps/cacert.pem.
TLS: certificate
[E=ldap@edt.org,CN=ldap.edt.org,OU=informatica,O=edt,L=barcelona,ST=barcelona,C=ca] is valid
TLS certificate verification: subject:
E=ldap@edt.org,CN=ldap.edt.org,OU=informatica,O=edt,L=barcelona,ST=barcelona,C=ca, issuer:
E=veritat@edt.org,CN=VeritatAbsoluta,OU=informatica,O=edt,L=barcelona,ST=barcelona,C=ca, cipher: AES-128-GCM, security level: high, secret key bits: 128, total key bits: 128, cache hits: 0, cache misses: 0, cache not reusable: 0
TLS: hostname (172.17.0.2) does not match common name in certificate (ldap.edt.org).
```

```
TLS: can't connect: TLS error -8157:Certificate extension not found..
ldap_err2string
ldap_start_tls: Connect error (-11)
   additional info: TLS error -8157:Certificate extension not found.
```

3) Si el client està ben configurat i la resolució al /etc/hosts també, la consulta TLS/SSL funcionarà.

```
 cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
172.17.0.2 ldap.edt.org
```

```
ldapsearch -x -LLL -ZZ -b 'dc=edt,dc=org' -h ldap.edt.org dn  | head -n2
dn: dc=edt,dc=org

ldapsearch -x -d1 -LLL -ZZ -b 'dc=edt,dc=org' -h ldap.edt.org dn  2> ldap.log
```

Test amb altres clients:

```
$ openssl s_client -connect ldap.edt.org:636
CONNECTED(00000003)
depth=1 C = ca, ST = barcelona, L = barcelona, O = edt, OU = informatica, CN = VeritatAbsoluta, emailAddress = veritat@edt.org
verify error:num=19:self signed certificate in certificate chain
---
Certificate chain
 0 s:/C=ca/ST=barcelona/L=barcelona/O=edt/OU=informatica/CN=ldap.edt.org/emailAddress=ldap@edt.org
   i:/C=ca/ST=barcelona/L=barcelona/O=edt/OU=informatica/CN=VeritatAbsoluta/emailAddress=veritat@edt.org
 1 s:/C=ca/ST=barcelona/L=barcelona/O=edt/OU=informatica/CN=VeritatAbsoluta/emailAddress=veritat@edt.org
   i:/C=ca/ST=barcelona/L=barcelona/O=edt/OU=informatica/CN=VeritatAbsoluta/emailAddress=veritat@edt.org
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIDnjCCAoYCCQDBXaOOex9i0zANBgkqhkiG9w0BAQsFADCBkzELMAkGA1UEBhMC
Y2ExEjAQBgNVBAgMCWJhcmNlbG9uYTESMBAGA1UEBwwJYmFyY2Vsb25hMQwwCgYD
VQQKDANlZHQxFDASBgNVBAsMC2luZm9ybWF0aWNhMRgwFgYDVQQDDA9WZXJpdGF0
QWJzb2x1dGExHjAcBgkqhkiG9w0BCQEWD3Zlcml0YXRAZWR0Lm9yZzAeFw0xOTAz
MjIxOTUxMjJaFw0yOTAzMTkxOTUxMjJaMIGNMQswCQYDVQQGEwJjYTESMBAGA1UE
CAwJYmFyY2Vsb25hMRIwEAYDVQQHDAliYXJjZWxvbmExDDAKBgNVBAoMA2VkdDEU
MBIGA1UECwwLaW5mb3JtYXRpY2ExFTATBgNVBAMMDGxkYXAuZWR0Lm9yZzEbMBkG
CSqGSIb3DQEJARYMbGRhcEBlZHQub3JnMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8A
MIIBCgKCAQEAkt50uVwPyF3/0aNm70lC5E8311zAcuK4kKl4hd9HJM2pjiR2O2CY
nW61c/BqriuQAvtNNgu7t83KZNbFO/Svd3+fDJeP7QDGMpxskB3Ha/2HeBXqTt0o
8LNT8MjCw0QCaZE3+k2v2mdCCq9KocJ0M+GbsT/RDAs7NZyYFP3R8nboHYa4otrK
aREEcn10T7NF1kQOXF0l5lEnXbSbf45MwhNfEVN9L0TtZOHnKAq5SGI9QigOWDtd
qY5haS8rxA0ZTDO9qjjkC79gWATr0FGkiunB6wpKk/aqFe6v7WgQeJQZ/0ssbluW
dEF4bspoTHv7HRBvS4vQDBDUHi1DoeYwXQIDAQABMA0GCSqGSIb3DQEBCwUAA4IB
AQCouaftY/7Gsoa+SGGItGAyavmIN8t8XqKEOV/F7Q8s/D35ODvSJEGLNnq3JW6C
A+blmny3sOObKujF5N6xUUevHkidT7TtyyZbKa6OCX0QPSwhalenCmzP1i+m1nkf
P3EikJRCdAYaSUzM58Raex5H8lh7cXx8XgMMiLDFblo8QGZhujpHd5KvVYTxnlt4
DiYFoFv7biJmG7Fq/g2bmMIsSK+2LnogJfHoBbEO+rzMCCtgpupkwsyu/U7GX05S
RDYeiITfo+Mhf7GXlQfAUFE8+cnvmZN9JPSoxZl4CFynzul3ZF1EeSqPLVtvO/hn
GKHICPJDeMgahCyjaTEn/N+4
-----END CERTIFICATE-----
subject=/C=ca/ST=barcelona/L=barcelona/O=edt/OU=informatica/CN=ldap.edt.org/emailAddress=ldap@edt.org
issuer=/C=ca/ST=barcelona/L=barcelona/O=edt/OU=informatica/CN=VeritatAbsoluta/emailAddress=veritat@edt.org
---
No client certificate CA names sent
Peer signing digest: SHA256
Server Temp Key: X25519, 253 bits
---
SSL handshake has read 2416 bytes and written 347 bytes
Verification error: self signed certificate in certificate chain
---
```

```
New, TLSv1.2, Cipher is ECDHE-RSA-AES128-GCM-SHA256
Server public key is 2048 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
No ALPN negotiated
SSL-Session:
        Protocol  : TLSv1.2
        Cipher    : ECDHE-RSA-AES128-GCM-SHA256
        Session-ID: 0015C9F72B2C179CEB148158A65BB2FAFAC439C050F90ED3FD02653F3E24355D
        Session-ID-ctx:
        Master-Key: 1474E37C0826EF2DB95FF29A23474604C90006C4854341D85935BEF5B35BF7A37BE727F720EAE81E9D3EBB6AD2BBF5BF
        PSK identity: None
        PSK identity hint: None
        SRP username: None
        Start Time: 1553672648
        Timeout   : 7200 (sec)
        Verify return code: 19 (self signed certificate in certificate chain)
        Extended master secret: no
---
```

Utilitats s_client, s_server, ncat

Client / Servidor ncat

```
$ ncat --ssl -k -l 8080
```

```
$ ncat --ssl  localhost 8080
$ openssl s_client localhost 8080
```

Client / Servidor / Verificar amb openssl

Engegar un servidor web

```
$ openssl s_server -key key.pem -cert cert.pem -accept 44330 -www
```

Es genera un error ja que el client no dona per bò el issuer del certificat

```
$ openssl s_client  -connect localhost:44330
```

Ara el client valida apropiadament el certificat del issuer:

```
$ openssl s_client -CAfile cacert.pem -connect localhost:44330
```

```
$ curl -ssl --cacert cacert.pem -v  https://localhost:44330
```

Connectar amb s_client a un servidor web amb virtual hosts:

```
$ openssl s_client  -servername www.web1.org  -connect 172.17.0.2:443

$ openssl s_client -CAfile cacert.pem  -servername www.web1.org  -connect
```

```
172.17.0.2:443
```

Connectar a gmail

```
openssl  s_client -connect pop.gmail.com:995
```

Podem usar openssl s_client per connectar amb starttls si és tracta d’un dels següents serveis:

```
$ openssl s_client -starttls  -servername www.web1.org  -connect 172.17.0.2:443
s_client: Value must be one of:
   smtp
   pop3
   imap
   ftp
   xmpp
   xmpp-server
   telnet
   irc
```

Verificar

```
openssl verify servercert.auto1.pem
servercert.auto1.pem: C = ca, ST = barcelona, L = barcelona, O = edt, OU = informatica,
CN = www.auto1.cat, emailAddress = auto1@edt.org
error 18 at 0 depth lookup:self signed certificate
OK
```

```
openssl verify servercert.web1.pem
servercert.web1.pem: C = ca, ST = barcelona, L = barcelona, O = edt, OU = informatica,
CN = www.web1.org, emailAddress = web1@edt.org
error 20 at 0 depth lookup:unable to get local issuer certificate
```

```
openssl verify -CAfile cacert.pem servercert.pem
servercert.web1.pem: OK
```

```
openssl verify -CAfile servercert.auto1.pem   servercert.auto1.pem
servercert.auto1.pem: OK
```

Obtenir / visualitzar

```
openssl s_client  -connect 172.17.0.2:443 2> /dev/null | openssl x509 -noout -text
```

```
$ openssl s_client  -connect pop.gmail.com:993 2> /dev/null | openssl x509 -noout -text
```

Descarregar i desar el certificat remot:

```
openssl s_client  -servername www.web1.org  -connect 172.17.0.2:443 < /dev/null 2>
/dev/null  | openssl x509 -outform PEM > downloaded.cert.pem
```

Observar certificat de Google

Per exemple podem fer: (en plan bast)

- consultar amb openssl _client la web i copiar manualment el text del certificat en PEM.
- copiar-lo i desar-lo en un fitxer.
- Examinar el fitxer

```
$  openssl  s_client -connect pop.gmail.com:995
# fer cut&paste del PEM

$ openssl x509 -noout -text -in /tmp/cert.pem
```

```
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number:
      ca:25:81:38:4e:d1:43:d7:03:00:00:00:00:cb:92:a9
  Signature Algorithm: sha256WithRSAEncryption
    Issuer: C = US, O = Google Trust Services, CN = GTS CA 1O1
    Validity
      Not Before: Feb 17 12:18:39 2021 GMT
      Not After : May 12 12:18:38 2021 GMT
    Subject: C = US, ST = California, L = Mountain View, O = Google LLC, CN = pop.gmail.com
...
    X509v3 extensions:
      X509v3 Key Usage: critical
        Digital Signature, Key Encipherment
      X509v3 Extended Key Usage:
        TLS Web Server Authentication
      X509v3 Basic Constraints: critical
        CA:FALSE
      X509v3 Subject Key Identifier:
        00:62:5B:DF:4B:D2:3B:31:09:4A:E6:4E:41:F7:A7:F8:7D:97:8B:4D
      X509v3 Authority Key Identifier:
        keyid:98:D1:F8:6E:10:EB:CF:9B:EC:60:9F:18:90:1B:A0:EB:7D:09:FD:2B

      Authority Information Access:
        OCSP - URI:http://ocsp.pki.goog/gts1o1core
```

```
            CA Issuers - URI:http://pki.goog/gsr2/GTS1O1.crt

        X509v3 Subject Alternative Name:
            DNS:pop.gmail.com
        X509v3 Certificate Policies:
            Policy: 2.23.140.1.2.2
            Policy: 1.3.6.1.4.1.11129.2.5.3

        X509v3 CRL Distribution Points:

            Full Name:
              URI:http://crl.pki.goog/GTS1O1core.crl
```

O també fer-ho directament amb:

```
openssl s_client -connect www.gmail.com:443 < /dev/null 2> /dev/null | openssl x509 -noout -text
```

IMAP: Accés segur amb TLS / StartTLS

Objectius:

- Configurar el servidor IMAP per acceptar connexions segures via IMAPs i IMAP amb StartTLS.
- Configurar al xinetd el servidor de uw-imap
- Configurar el servei de Cyrus Imap.

Uw-imap

Imaps

```
[root@d01 ~]# rpm -ql uw-imap

[root@d01 ~]# rpm -ql uw-imap
/etc/pam.d/imap
/etc/pam.d/pop
/etc/pki/tls/certs/imapd.pem
/etc/pki/tls/certs/ipop3d.pem
/etc/xinetd.d/imap
/etc/xinetd.d/imaps
/etc/xinetd.d/ipop2
/etc/xinetd.d/ipop3
/etc/xinetd.d/pop3s
/usr/sbin/imapd
```

```
/usr/sbin/ipop2d
/usr/sbin/ipop3d
/usr/share/doc/uw-imap
/usr/share/doc/uw-imap/SSLBUILD
/usr/share/man/man8/imapd.8uw.gz
/usr/share/man/man8/ipopd.8uw.gz
```

```
[root@d01 ~]# openssl x509 -noout -text -in /etc/pki/tls/certs/imapd.pem
Certificate:
        Data:
        Version: 3 (0x2)
        Serial Number:
        f7:df:94:d4:f9:22:58:75
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=--, ST=SomeState, L=SomeCity, O=SomeOrganization,
OU=SomeOrganizationalUnit,
CN=localhost.localdomain/emailAddress=root@localhost.localdomain
        Validity
        Not Before: Mar 28 11:56:19 2017 GMT
        Not After : Mar 28 11:56:19 2018 GMT
        Subject: C=--, ST=SomeState, L=SomeCity, O=SomeOrganization,
OU=SomeOrganizationalUnit,
CN=localhost.localdomain/emailAddress=root@localhost.localdomain
        Subject Public Key Info:
        Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                00:cc:be:9f:a2:97:c5:f9:b3:64:d5:3c:63:6a:c6:
                f7:e0:b0:fa:a9:e3:5e:82:e9:00:06:f8:ef:02:fd:
                a7:af:81:d7:b4:33:df:e8:33:a1:fc:8b:f3:d6:3a:
                f9:74:c1:70:04:a9:de:83:8f:4f:5c:71:db:02:66:
                9e:cb:fc:fa:dc:c9:73:17:67:30:41:e9:05:0e:30:
                b3:15:88:c2:ac:5d:4b:71:1c:9b:cb:ba:b3:ba:2c:
                da:0a:c2:99:ac:15:af:30:26:32:91:44:65:31:b0:
                76:ee:3e:d4:28:4a:b6:e9:57:f6:57:33:d4:5c:7d:
                77:f4:ee:9a:14:f7:19:0b:46:57:2c:fe:5c:64:ec:
                65:da:9b:17:69:fe:58:3e:27:c2:91:f5:aa:25:19:
                8d:6b:39:a1:8c:22:56:02:ec:41:4e:44:c4:20:74:
                6a:a6:07:3e:0d:be:9e:be:2a:2a:a3:7e:7b:e7:4b:
                16:9e:a9:ab:c0:64:fd:ad:65:df:9c:65:fb:ed:4a:
                39:a6:b7:83:39:e3:48:9b:9e:c4:d3:e1:0e:79:01:
                36:50:8a:ac:07:33:0d:3a:04:c1:c0:1b:e6:4a:36:
                41:9d:58:e8:a5:c3:79:b2:0a:7a:9e:77:20:ad:ce:
                d3:f5:f1:b2:57:5b:fc:53:5a:87:64:87:83:32:06:
                44:f1
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                21:75:38:63:A2:2D:FF:7D:DC:C2:6C:7A:48:5E:87:59:67:23:F9:38
            X509v3 Authority Key Identifier:
                keyid:21:75:38:63:A2:2D:FF:7D:DC:C2:6C:7A:48:5E:87:59:67:23:F9:38

            X509v3 Basic Constraints:
                CA:TRUE
            Signature Algorithm: sha256WithRSAEncryption
            af:53:35:9a:58:40:46:7d:d7:0b:0c:ff:5e:7d:8d:84:10:70:
            f7:3f:ce:09:e1:39:81:c2:33:1f:30:60:87:07:8f:37:c6:bb:
            08:d8:dc:8c:62:9e:e6:ea:a2:4d:38:3a:db:bd:67:a4:79:e7:
            98:8e:4b:17:8d:92:d9:0c:9d:a4:a1:c2:a6:72:04:ee:90:37:
            c6:47:de:df:3f:e2:f1:96:e2:ad:3f:4b:99:c9:25:1b:40:0e:
            64:a3:83:26:57:b5:4c:1b:d9:2e:92:f1:18:05:e7:3d:39:7f:
            03:43:b3:65:d5:2f:40:37:ec:26:5a:15:39:8f:9a:4a:8c:4b:
            de:c4:3b:d2:94:80:20:9e:dc:ad:d8:0c:b7:8b:7a:fb:f8:aa:
            87:30:c3:c9:1b:1f:7d:ce:5f:ba:b9:91:f2:bb:bc:77:94:80:
            2b:7e:36:43:e6:5d:8d:cc:29:e7:08:da:8a:e0:d2:33:52:03:
            ca:03:75:d1:fb:d6:af:c0:39:0e:01:af:6d:35:b1:f7:82:16:
            21:6e:b4:6f:8d:4a:91:22:37:cd:6e:ba:30:73:dd:75:1a:11:
            24:18:aa:b1:68:2a:a4:d1:0a:60:9a:e1:fd:22:fa:a6:84:d3:
            b5:5c:19:fe:64:4a:15:12:93:b6:29:3c:3e:9b:85:8c:59:7a:
            52:2f:2d:ba
```

```
[root@d01 ~]# openssl x509 -noout -purpose -in /etc/pki/tls/certs/imapd.pem
Certificate purposes:
SSL client : Yes
SSL client CA : Yes
SSL server : Yes
SSL server CA : Yes
Netscape SSL server : Yes
Netscape SSL server CA : Yes
S/MIME signing : Yes
S/MIME signing CA : Yes
S/MIME encryption : Yes
S/MIME encryption CA : Yes
CRL signing : Yes
CRL signing CA : Yes
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
OCSP helper CA : Yes
Time Stamp signing : No
Time Stamp signing CA : Yes
```

```
root@d01 ~]# cat /etc/xinetd.d/imaps
# default: off
# description: The IMAPS service allows remote users to access their mail \
#       using an IMAP client with SSL support such as Netscape \
#       Communicator or fetchmail.
service imaps
{
   socket_type          = stream
   wait                 = no
   user                 = root
   server               = /usr/sbin/imapd
   log_on_success  += HOST DURATION
   log_on_failure    += HOST
   disable              = no
}
```

```
[root@d01 ~]# systemctl restart xinetd

[root@d01 ~]# systemctl status xinetd
● xinetd.service - Xinetd A Powerful Replacement For Inetd
   Loaded: loaded (/usr/lib/systemd/system/xinetd.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2017-03-28 14:01:39 CEST; 6s ago
        Docs: man:xinetd
        man:xinetd.conf
        man:xinetd.log
  Process: 6361 ExecReload=/usr/bin/kill -HUP $MAINPID (code=exited, status=0/SUCCESS)
  Process: 6593 ExecStart=/usr/sbin/xinetd -stayalive -pidfile /var/run/xinetd.pid (code=exited,
status=0/SUCCE
 Main PID: 6594 (xinetd)
        Tasks: 1 (limit: 512)
   CGroup: /system.slice/xinetd.service
        └─6594 /usr/sbin/xinetd -stayalive -pidfile /var/run/xinetd.pid

Mar 28 14:01:39 d01.informatica.escoladeltreball.org xinetd[6594]: removing echo
Mar 28 14:01:39 d01.informatica.escoladeltreball.org xinetd[6594]: removing echo
Mar 28 14:01:39 d01.informatica.escoladeltreball.org xinetd[6594]: removing imap
Mar 28 14:01:39 d01.informatica.escoladeltreball.org xinetd[6594]: removing pop2
Mar 28 14:01:39 d01.informatica.escoladeltreball.org xinetd[6594]: removing pop3
Mar 28 14:01:39 d01.informatica.escoladeltreball.org xinetd[6594]: removing tcpmux
Mar 28 14:01:39 d01.informatica.escoladeltreball.org xinetd[6594]: removing time
Mar 28 14:01:39 d01.informatica.escoladeltreball.org xinetd[6594]: removing time
Mar 28 14:01:39 d01.informatica.escoladeltreball.org xinetd[6594]: xinetd Version 2.3.15 started
with libwrap l
Mar 28 14:01:39 d01.informatica.escoladeltreball.org xinetd[6594]: Started working: 2 available
services
```

```
[root@d01 ~]# nmap localhost

Starting Nmap 7.40 ( https://nmap.org ) at 2017-03-28 14:04 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000080s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 996 closed ports
PORT  STATE SERVICE
22/tcp  open  ssh
631/tcp open  ipp
993/tcp open  imaps
995/tcp open  pop3s
```

```
[root@d01 ~]# openssl s_client -connect localhost:993
CONNECTED(00000003)
depth=0 C = --, ST = SomeState, L = SomeCity, O = SomeOrganization, OU =
SomeOrganizationalUnit, CN = localhost.localdomain, emailAddress =
root@localhost.localdomain
verify error:num=18:self signed certificate
verify return:1
depth=0 C = --, ST = SomeState, L = SomeCity, O = SomeOrganization, OU =
SomeOrganizationalUnit, CN = localhost.localdomain, emailAddress =
root@localhost.localdomain
verify return:1
---
Certificate chain
 0
s:/C=--/ST=SomeState/L=SomeCity/O=SomeOrganization/OU=SomeOrganizationalUnit/CN=loc
alhost.localdomain/emailAddress=root@localhost.localdomain

i:/C=--/ST=SomeState/L=SomeCity/O=SomeOrganization/OU=SomeOrganizationalUnit/CN=loc
alhost.localdomain/emailAddress=root@localhost.localdomain
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIESzCCAzOgAwIBAgIJAPffINT5Ilh1MA0GCSqGSIb3DQEBCwUAMIG7MQswCQYD
VQQGEwItLTESMBAGA1UECAwJU29tZVN0YXRlMREwDwYDVQQHDAhTb21lQ2l0eTEZ
MBcGA1UECgwQU29tZU9yZ2FuaXphdGlvbjEfMB0GA1UECwwWU29tZU9yZ2FuaXph
dGlvbmFsVW5pdDEeMBwGA1UEAwwVbG9jYWxob3N0LmxvY2FsZG9tYWluMSkwJwYJ
KoZIhvcNAQkBFhpyb290QGxvY2FsaG9zdC5sb2NhbGRvbWFpbjAeFw0xNzAzMjgx
MTU2MTlaFw0xODAzMjgxMTU2MTlaMIG7MQswCQYDVQQGEwItLTESMBAGA1UECAwJ
U29tZVN0YXRlMREwDwYDVQQHDAhTb21lQ2l0eTEZMBcGA1UECgwQU29tZU9yZ2Fu
aXphdGlvbjEfMB0GA1UECwwWU29tZU9yZ2FuaXphdGlvbmFsVW5pdDEeMBwGA1UE
AwwVbG9jYWxob3N0LmxvY2FsZG9tYWluMSkwJwYJKoZIhvcNAQkBFhpyb290QGxv
Y2FsaG9zdC5sb2NhbGRvbWFpbjCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoC
ggEBAMy+n6KXxfmzZNU8Y2rG9+Cw+qnjXoLpAAb47wL9p6+B17Qz3+gzofyL89Y6
+XTBcASp3oOPT1xx2wJmnsv8+tzJcxdnMEHpBQ4wsxWIwqxdS3Ecm8u6s7os2grC
mawVrzAmMpFEZTGwdu4+1ChKtulX9lcz1Fx9d/TumhT3GQtGVyz+XGTsZdqbF2n+
WD4nwpH1qiUZjWs5oYwiVgLsQU5ExCB0aqYHPg2+nr4qKqN+e+dLFp6pq8Bk/a1I
35xl++1KOaa3gznjSJuexNPhDnkBNlCKrAczDToEwcAb5ko2QZ1Y6KXDebIKep53
lK3O0/Xxsldb/FNah2SHgzIGRPECAwEAAaNQME4wHQYDVR0OBBYEFCF1OGOiLf99
3MJsekheh1lnI/k4MB8GA1UdIwQYMBaAFCF1OGOiLf993MJsekheh1lnI/k4MAwG
```

```
A1UdEwQFMAMBAf8wDQYJKoZIhvcNAQELBQADggEBAK9TNZpYQEZ91wsM/159jYQQ
cPc/zgnhOYHCMx8wYIcHjzfGuwjY3Ixinubqok04Otu9Z6R555iOSxeNktkMnaSh
wqZyBO6QN8ZH3t8/4vGW4q0/S5nJJRtADmSjgyZXtUwb2S6S8RgF5z05fwNDs2XV
L0A37CZaFTmPmkqMS97EO9KUgCCe3K3YDLeLevv4qocww8kbH33OX7q5kfK7vHeU
gCt+NkPmXY3MKecI2org0jNSA8oDddH71q/AOQ4Br201sfeCFiFutG+NSpEiN81u
ujBz3XUaESQYqrFoKqTRCmCa4f0i+qaE07VcGf5kShUSk7YpPD6bhYxZeIIvLbo=
-----END CERTIFICATE-----
subject=/C=--/ST=SomeState/L=SomeCity/O=SomeOrganization/OU=SomeOrganizationalUnit/
CN=localhost.localdomain/emailAddress=root@localhost.localdomain
issuer=/C=--/ST=SomeState/L=SomeCity/O=SomeOrganization/OU=SomeOrganizationalUnit/C
N=localhost.localdomain/emailAddress=root@localhost.localdomain
---
No client certificate CA names sent
---
SSL handshake has read 1416 bytes and written 519 bytes
---
New, TLSv1/SSLv3, Cipher is AES256-GCM-SHA384
Server public key is 2048 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
No ALPN negotiated
SSL-Session:
        Protocol  : TLSv1.2
        Cipher : AES256-GCM-SHA384
        Session-ID:
DC16FEB483EA34D09D0407DD8148BA73FBE517E50A620D049456580FB957B353
        Session-ID-ctx:
        Master-Key:
373180FA7A6D9E51D5522BB2F39F198E7DE83EDA3FA43CCC7925DAA2E94BED5C68C865
B0FCD26BC2AA333AA9842A72B0
        Key-Arg   : None
        Krb5 Principal: None
        PSK identity: None
        PSK identity hint: None
        TLS session ticket lifetime hint: 300 (seconds)
        TLS session ticket:
        0000 - 25 41 00 13 83 ec 52 a0-f2 ec 8d 30 76 0f 90 10   %A....R....0v...
        0010 - 97 a9 29 ea b6 ed 58 00-95 89 51 ea 05 33 9d 92   ..)...X...Q..3..
        0020 - cb ab fd d8 41 c2 86 7e-cd bd 32 b7 20 a0 32 f7   ....A..~..2. .2.
        0030 - 27 1c 75 bf 9c cc 0b e2-eb 9a 57 f7 55 76 83 32   '.u.......W.Uv.2
        0040 - ea cc f8 af b5 9f 7b 55-92 d3 97 c4 ac 5f 92 cd   ......{U....._..
        0050 - a2 79 1a d1 53 2a 19 33-59 75 72 12 2a f7 f4 ee   .y..S*.3Yur.*...
        0060 - 44 1a 69 f8 b4 63 eb 70-22 bc 80 83 d9 13 f1 6b   D.i..c.p"......k
        0070 - e1 98 7b eb 08 56 54 d8-c8 01 4b e1 f0 fb 16 c8   ..{..VT...K.....
        0080 - 1a b1 87 bb 43 f1 a7 d9-f7 e6 b0 ea ed 71 a6 72   ....C........q.r
        0090 - 2d 6e e5 fb c1 9f de 8a-80 9c c2 bb e3 f1 48 d6   -n............H.

        Start Time: 1490702704
        Timeout   : 300 (sec)
        Verify return code: 18 (self signed certificate)
---
* OK [CAPABILITY IMAP4REV1 I18NLEVEL=1 LITERAL+ SASL-IR LOGIN-REFERRALS
AUTH=GSSAPI AUTH=PLAIN AUTH=LOGIN] localhost IMAP4rev1 2007f.404 at Tue, 28 Mar
2017 14:05:04 +0200 (CEST)

a001 LOGIN m11pere pere
a001 OK [CAPABILITY IMAP4REV1 I18NLEVEL=1 LITERAL+ IDLE UIDPLUS NAMESPACE
CHILDREN MAILBOX-REFERRALS BINARY UNSELECT ESEARCH WITHIN SCAN SORT
THREAD=REFERENCES THREAD=ORDEREDSUBJECT MULTIAPPEND] User m11pere
authenticated

a004 SELECT inbox
* 0 EXISTS
* 0 RECENT
* OK [UIDVALIDITY 1490702954] UID validity status
* OK [UIDNEXT 1] Predicted next UID
* FLAGS (\Answered \Flagged \Deleted \Draft \Seen)
* OK [PERMANENTFLAGS ()] Permanent flags
a004 OK [READ-WRITE] SELECT completed

a005 logout
* BYE d01.informatica.escoladeltreball.org IMAP4rev1 server terminating connection
a005 OK LOGOUT completed
read:errno=0
```

```
[root@d01 ~]# imtest localhost
S: * OK [CAPABILITY IMAP4REV1 I18NLEVEL=1 LITERAL+ SASL-IR LOGIN-REFERRALS
STARTTLS AUTH=GSSAPI] localhost IMAP4rev1 2007f.404 at Tue, 28 Mar 2017 14:20:40
+0200 (CEST)
Authentication failed. generic failure
Security strength factor: 0

a001 CAPABILITY
* CAPABILITY IMAP4REV1 I18NLEVEL=1 LITERAL+ IDLE UIDPLUS NAMESPACE
CHILDREN MAILBOX-REFERRALS BINARY UNSELECT ESEARCH WITHIN SCAN SORT
THREAD=REFERENCES THREAD=ORDEREDSUBJECT MULTIAPPEND SASL-IR
LOGIN-REFERRALS STARTTLS AUTH=GSSAPI
a001 OK CAPABILITY completed

a002 LOGIN m11pere pere
a002 OK [CAPABILITY IMAP4REV1 I18NLEVEL=1 LITERAL+ IDLE UIDPLUS NAMESPACE
CHILDREN MAILBOX-REFERRALS BINARY UNSELECT ESEARCH WITHIN SCAN SORT
THREAD=REFERENCES THREAD=ORDEREDSUBJECT MULTIAPPEND] User m11pere
authenticated

a003 SELECT inbox
```

```
* 0 EXISTS
* 0 RECENT
* OK [UIDVALIDITY 1490703692] UID validity status
* OK [UIDNEXT 1] Predicted next UID
* FLAGS (\Answered \Flagged \Deleted \Draft \Seen)
* OK [PERMANENTFLAGS ()] Permanent flags
a003 OK [READ-WRITE] SELECT completed

a004 LOGOUT
* BYE d01.informatica.escoladeltreball.org IMAP4rev1 server terminating connection
a004 OK LOGOUT completed
Connection closed.
```

Imap + StartTLS

```
[root@d01 ~]# cat /etc/xinetd.d/imap
# default: off
# description: The IMAP service allows remote users to access their mail using \
#              an IMAP client such as Mutt, Pine, fetchmail, or Netscape \
#              Communicator.
service imap
{
   socket_type          = stream
   wait                 = no
   user                 = root
   server               = /usr/sbin/imapd
   log_on_success    += HOST DURATION
   log_on_failure    += HOST
   disable              = no
}
```

```
root@d01 ~]# systemctl  restart xinetd

[root@d01 ~]# nmap localhost
Starting Nmap 7.40 ( https://nmap.org ) at 2017-03-28 14:13 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000090s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 995 closed ports
PORT  STATE SERVICE
22/tcp  open  ssh
143/tcp open  imap
631/tcp open  ipp
```

```
993/tcp open  imaps
995/tcp open  pop3s
```

```
[root@d01 ~]# telnet localhost 143
Trying ::1...
Connected to localhost.
Escape character is '^]'.
* OK [CAPABILITY IMAP4REV1 I18NLEVEL=1 LITERAL+ SASL-IR LOGIN-REFERRALS
STARTTLS AUTH=GSSAPI] localhost IMAP4rev1 2007f.404 at Tue, 28 Mar 2017 14:13:50
+0200 (CEST)

a001 CAPABILITY
* CAPABILITY IMAP4REV1 I18NLEVEL=1 LITERAL+ IDLE UIDPLUS NAMESPACE
CHILDREN MAILBOX-REFERRALS BINARY UNSELECT ESEARCH WITHIN SCAN SORT
THREAD=REFERENCES THREAD=ORDEREDSUBJECT MULTIAPPEND SASL-IR
LOGIN-REFERRALS STARTTLS AUTH=GSSAPI
a001 OK CAPABILITY completed

a002 STARTTLS
a002 OK STARTTLS completed
```

cyrus

```
Installed:
  cyrus-imapd.x86_64 2.4.18-2.fc24           cyrus-imapd-utils.x86_64 2.4.18-2.fc24
lm_sensors-libs.x86_64 3.4.0-4.fc24          net-snmp-agent-libs.x86_64 1:5.7.3-13.fc24
```

```
root@d01 ~]# imtest  -h
Usage: imtest [options] hostname
  -p port  : port to use (default=standard port for protocol)
  -z       : timing test
  -k #     : minimum protection layer required
  -l #     : max protection layer (0=none; 1=integrity; etc)
  -u user  : authorization name to use
  -a user  : authentication name to use
  -w pass  : password to use (if not supplied, we will prompt)
  -v       : verbose
  -m mech  : SASL mechanism to use
             ("login" for IMAP LOGIN)
  -f file  : pipe file into connection after authentication
  -r realm : realm
  -s       : Enable imap over SSL (imaps)
  -t file  : Enable TLS. file has the TLS public and private keys
           (specify "" to not use TLS for authentication)
  -q       : Enable imap COMPRESSion (before last authentication attempt)
  -c       : enable challenge prompt callbacks
           (enter one-time password instead of secret pass-phrase)
  -n       : number of auth attempts (default=1)
  -I file  : output my PID to (file) (useful with -X)
  -x file  : open the named socket for the interactive portion
  -X file  : same as -X, except close all file descriptors & dameonize
```

```
[root@d01 ~]# pop3test  -h
Usage: pop3test [options] hostname
  -p port  : port to use (default=standard port for protocol)
  -k #     : minimum protection layer required
  -l #     : max protection layer (0=none; 1=integrity; etc)
  -u user  : authorization name to use
  -a user  : authentication name to use
  -w pass  : password to use (if not supplied, we will prompt)
  -v       : verbose
  -m mech  : SASL mechanism to use
           ("user" for USER/PASS, "apop" for APOP)
  -f file  : pipe file into connection after authentication
  -r realm : realm
  -s       : Enable pop3 over SSL (pop3s)
  -t file  : Enable TLS. file has the TLS public and private keys
           (specify "" to not use TLS for authentication)
  -c       : enable challenge prompt callbacks
           (enter one-time password instead of secret pass-phrase)
  -n       : number of auth attempts (default=1)
  -I file  : output my PID to (file) (useful with -X)
  -x file  : open the named socket for the interactive portion
  -X file  : same as -X, except close all file descriptors & dameonize
```

```
root@d01 ~]# rpm -ql cyrus-imapd
/etc/cron.daily/cyrus-imapd
/etc/cyrus.conf
/etc/imapd.conf
/etc/logrotate.d/cyrus-imapd
/etc/pam.d/csync
/etc/pam.d/imap
/etc/pam.d/lmtp
/etc/pam.d/mupdate
/etc/pam.d/nntp
/etc/pam.d/pop
/etc/pam.d/sieve
```

```
/etc/pki/cyrus-imapd
/etc/pki/cyrus-imapd/cyrus-imapd.pem
/etc/sysconfig/cyrus-imapd
/usr/lib/cyrus-imapd
/usr/lib/cyrus-imapd/arbitron
/usr/lib/cyrus-imapd/arbitronsort.pl
/usr/lib/cyrus-imapd/chk_cyrus
/usr/lib/cyrus-imapd/convert-sieve.pl
/usr/lib/cyrus-imapd/ctl_cyrusdb
/usr/lib/cyrus-imapd/ctl_deliver
/usr/lib/cyrus-imapd/ctl_mboxlist
/usr/lib/cyrus-imapd/cvt_cyrusdb
/usr/lib/cyrus-imapd/cvt_cyrusdb_all
/usr/lib/cyrus-imapd/cyr_dbtool
/usr/lib/cyrus-imapd/cyr_df
/usr/lib/cyrus-imapd/cyr_expire
/usr/lib/cyrus-imapd/cyr_sequence
/usr/lib/cyrus-imapd/cyr_synclog
/usr/lib/cyrus-imapd/cyr_systemd_helper
/usr/lib/cyrus-imapd/cyr_userseen
/usr/lib/cyrus-imapd/cyrdump
/usr/lib/cyrus-imapd/cyrfetchnews
/usr/lib/cyrus-imapd/cyrus-master
/usr/lib/cyrus-imapd/deliver
/usr/lib/cyrus-imapd/dohash
/usr/lib/cyrus-imapd/fud
/usr/lib/cyrus-imapd/idled
/usr/lib/cyrus-imapd/imapd
/usr/lib/cyrus-imapd/ipurge
/usr/lib/cyrus-imapd/lmtpd
/usr/lib/cyrus-imapd/lmtpproxyd
/usr/lib/cyrus-imapd/masssievec
/usr/lib/cyrus-imapd/mbexamine
/usr/lib/cyrus-imapd/mbpath
/usr/lib/cyrus-imapd/migrate-metadata
/usr/lib/cyrus-imapd/mkimap
/usr/lib/cyrus-imapd/mknewsgroups
/usr/lib/cyrus-imapd/mupdate
/usr/lib/cyrus-imapd/mupdate-loadgen.pl
/usr/lib/cyrus-imapd/nntpd
/usr/lib/cyrus-imapd/notifyd
/usr/lib/cyrus-imapd/pop3d
/usr/lib/cyrus-imapd/proxyd
/usr/lib/cyrus-imapd/ptdump
/usr/lib/cyrus-imapd/ptexpire
/usr/lib/cyrus-imapd/ptloader
/usr/lib/cyrus-imapd/quota
/usr/lib/cyrus-imapd/reconstruct
```

```
/usr/lib/cyrus-imapd/rehash
/usr/lib/cyrus-imapd/sievec
/usr/lib/cyrus-imapd/sieved
/usr/lib/cyrus-imapd/smmapd
/usr/lib/cyrus-imapd/squatter
/usr/lib/cyrus-imapd/sync_client
/usr/lib/cyrus-imapd/sync_reset
/usr/lib/cyrus-imapd/sync_server
/usr/lib/cyrus-imapd/timsieved
/usr/lib/cyrus-imapd/tls_prune
/usr/lib/cyrus-imapd/translatesieve
/usr/lib/cyrus-imapd/undohash
/usr/lib/cyrus-imapd/unexpunge
/usr/lib/cyrus-imapd/upgradesieve
/usr/lib/systemd/system/cyrus-imapd.service
/usr/share/cyrus-imapd
/usr/share/cyrus-imapd/rpm
/usr/share/cyrus-imapd/rpm/db.cfg
/usr/share/cyrus-imapd/rpm/magic
/usr/share/doc/cyrus-imapd
/usr/share/doc/cyrus-imapd/README
/usr/share/doc/cyrus-imapd/README.rpm
...
/usr/share/doc/cyrus-imapd/text/specs
/usr/share/licenses/cyrus-imapd
/usr/share/licenses/cyrus-imapd/COPYRIGHT
/usr/share/man/man5/cyrus.conf.5.gz
/usr/share/man/man5/imapd.conf.5.gz
….
/var/lib/imap
/var/lib/imap/backup
/var/lib/imap/db
/var/lib/imap/log
/var/lib/imap/md5
/var/lib/imap/meta
/var/lib/imap/msg
/var/lib/imap/proc
/var/lib/imap/ptclient
/var/lib/imap/quota
/var/lib/imap/rpm
/var/lib/imap/sieve
/var/lib/imap/socket
/var/lib/imap/sync
/var/lib/imap/user
/var/spool/imap
```

```
[root@d01 ~]# cat /etc/cyrus.conf
```

```
# standard standalone server implementation

START {
  # do not delete this entry!
  recover    cmd="ctl_cyrusdb -r"

  # this is only necessary if using idled for IMAP IDLE
  idled           cmd="idled"
}

# UNIX sockets start with a slash and are put into /var/lib/imap/sockets
SERVICES {
  # add or remove based on preferences
  imap            cmd="imapd" listen="imap" prefork=5
  imaps           cmd="imapd -s" listen="imaps" prefork=1
  pop3            cmd="pop3d" listen="pop3" prefork=3
  pop3s           cmd="pop3d -s" listen="pop3s" prefork=1
  sieve           cmd="timsieved" listen="sieve" prefork=0

  # these are only necessary if receiving/exporting usenet via NNTP
#  nntp           cmd="nntpd" listen="nntp" prefork=3
#  nntps          cmd="nntpd -s" listen="nntps" prefork=1

  # at least one LMTP is required for delivery
#  lmtp           cmd="lmtpd" listen="lmtp" prefork=0
  lmtpunix    cmd="lmtpd" listen="/var/lib/imap/socket/lmtp" prefork=1

  # this is only necessary if using notifications
#  notify    cmd="notifyd" listen="/var/lib/imap/socket/notify" proto="udp" prefork=1
}

EVENTS {
  # this is required
  checkpoint    cmd="ctl_cyrusdb -c" period=30

  # this is only necessary if using duplicate delivery suppression,
  # Sieve or NNTP
  delprune    cmd="cyr_expire -E 3" at=0400

  # this is only necessary if caching TLS sessions
  tlsprune    cmd="tls_prune" at=0400
}
```

```
[root@d01 ~]# cat /etc/imapd.conf
configdirectory: /var/lib/imap
```

```
partition-default: /var/spool/imap
admins: cyrus
sievedir: /var/lib/imap/sieve
sendmail: /usr/sbin/sendmail
hashimapspool: true
sasl_pwcheck_method: saslauthd
sasl_mech_list: PLAIN LOGIN
allowplaintext: no
defaultdomain: mail
tls_cert_file: /etc/pki/cyrus-imapd/cyrus-imapd.pem
tls_key_file: /etc/pki/cyrus-imapd/cyrus-imapd.pem
tls_ca_file: /etc/pki/tls/certs/ca-bundle.crt
# uncomment this if you're operating in a DSCP environment (RFC-4594)
# qosmarking: af13
```

```
[root@d01 ~]# cat /etc/sysconfig/cyrus-imapd
# Options to cyrus-master
CYRUSOPTIONS=""

# Mailbox list dumps are rotated n times via cron.daily
#ROTATE=6
```

```
[root@d01 ~]# saslauthd -v
saslauthd 2.1.26
authentication mechanisms: getpwent kerberos5 pam rimap shadow ldap httpform
```

```
[root@d01 ~]# openssl x509 -noout -text -in /etc/pki/cyrus-imapd/cyrus-imapd.pem
Certificate:
        Data:
        Version: 3 (0x2)
        Serial Number:
        c2:63:fb:c1:b8:00:4e:e2
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=--, ST=SomeState, L=SomeCity, O=SomeOrganization,
OU=SomeOrganizationalUnit,
CN=localhost.localdomain/emailAddress=root@localhost.localdomain
        Validity
        Not Before: Mar 28 11:48:34 2017 GMT
        Not After : Mar 28 11:48:34 2018 GMT
        Subject: C=--, ST=SomeState, L=SomeCity, O=SomeOrganization,
OU=SomeOrganizationalUnit,
CN=localhost.localdomain/emailAddress=root@localhost.localdomain
```

```
Subject Public Key Info:
Public Key Algorithm: rsaEncryption
        Public-Key: (2048 bit)
        Modulus:
        00:c6:ec:71:bd:e7:2c:22:b7:84:3b:2b:61:00:c9:
        2e:b4:26:de:5d:35:8e:73:49:42:47:51:e0:5f:64:
        4c:4e:20:06:03:66:3d:91:5c:ed:fd:c0:29:ea:8b:
        97:3f:bc:55:d7:91:4c:b3:42:b2:00:cc:9a:b3:cc:
        f2:a5:92:53:db:5b:96:60:e0:b3:cf:14:1b:36:7d:
        53:d2:56:6e:0f:7d:28:94:41:ef:6a:b5:92:e3:1e:
        bf:b3:9c:d7:99:28:6d:20:f8:5a:43:b5:a0:8e:7d:
        dc:3e:83:7d:f3:e9:89:1d:5d:12:7a:13:e3:04:1f:
        27:30:1b:e6:77:52:d7:4d:3a:5b:21:e7:98:f4:40:
        14:69:15:0e:d8:43:5a:a3:b6:23:cb:6f:b4:f4:99:
        aa:63:c2:57:89:98:a2:9b:92:e9:c3:ed:e3:7d:85:
        47:40:89:5e:5c:f8:96:a9:b3:b0:26:d4:ad:e2:10:
        7e:93:5b:c2:0f:fc:25:b1:87:61:60:fb:b6:4a:24:
        0f:2d:ed:9a:02:1c:68:82:6e:16:11:01:bf:1c:46:
        58:b6:5b:44:28:39:ca:16:5f:fe:b7:f4:18:29:fb:
        32:b4:6e:21:18:24:6a:e5:f4:e8:a7:36:10:a0:37:
        6d:1a:0b:28:30:fd:b9:9c:d8:96:d8:b1:1e:11:af:
        ff:7b
        Exponent: 65537 (0x10001)
X509v3 extensions:
X509v3 Subject Key Identifier:
        4B:91:F3:2E:F0:F3:50:D2:DB:14:BB:C2:53:2B:A9:50:55:8F:52:80
X509v3 Authority Key Identifier:
        keyid:4B:91:F3:2E:F0:F3:50:D2:DB:14:BB:C2:53:2B:A9:50:55:8F:52:80

X509v3 Basic Constraints:
        CA:TRUE
Signature Algorithm: sha256WithRSAEncryption
a7:a3:c8:3d:2c:45:d9:48:e2:bd:75:78:75:87:8b:2f:7a:be:
e7:90:02:81:b6:70:3a:04:0f:43:c4:0a:d1:eb:f9:4b:35:1e:
d9:e2:8e:a8:6f:f2:f5:8a:d1:07:8c:c8:16:98:e8:22:77:00:
4f:f6:e8:63:8b:bd:02:23:2c:0a:9d:ae:27:4d:eb:64:45:ad:
ea:55:5a:ff:76:ae:c6:d0:c7:c8:a5:77:48:41:4e:c9:19:7a:
99:58:e2:08:39:c3:48:da:bb:83:d3:00:42:a1:81:51:cd:f7:
4f:e0:9a:19:79:cc:0f:f8:b0:a0:bc:e1:e1:03:3d:6d:4f:31:
e9:43:51:c7:15:f6:53:5b:27:e3:17:c0:29:49:7e:90:f6:e8:
90:3d:ea:81:49:c0:d6:df:5c:bf:3c:26:f0:75:3b:5b:0d:f2:
ff:17:c1:1a:4a:cf:1a:fe:bf:c3:8e:c0:97:35:df:86:36:c3:
f5:49:37:6c:b5:95:ce:7e:26:29:63:f2:c1:54:ac:a1:96:15:
71:f1:21:55:9f:ed:43:1e:28:eb:51:40:57:e5:31:45:0d:12:
90:d7:b0:02:c6:66:f5:36:39:3f:2c:03:2f:c4:5f:c2:d5:30:
ae:3b:99:54:6f:ef:52:6f:ad:8a:ab:c1:56:f4:eb:bc:a3:ab:
```

```
         c6:50:8c:f0
```

https://cyrusimap.org/

https://www.cyrusimap.org/docs/cyrus-imapd/2.4.7/install-configure.php

https://cyrusimap.org/imap/developer/basicserver.html

```
[root@d01 ~]# /usr/lib/cyrus-imapd/cyrus-master -d

[root@d01 ~]# nmap localhost
Starting Nmap 7.40 ( https://nmap.org ) at 2017-03-28 14:52 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000090s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 994 closed ports
PORT STATE SERVICE
22/tcp  open  ssh
110/tcp open  pop3
143/tcp open  imap
631/tcp open  ipp
993/tcp open  imaps
995/tcp open  pop3s
```

```
[root@d01 ~]# telnet localhost 143
Trying ::1...
Connected to localhost.
Escape character is '^]'.
* OK [CAPABILITY IMAP4rev1 LITERAL+ ID ENABLE STARTTLS LOGINDISABLED]
d01.informatica.escoladeltreball.org Cyrus IMAP v2.4.18-Fedora-RPM-2.4.18-2.fc24
server ready

a001 STARTTLS
a001 OK Begin TLS negotiation now
```

```
[root@d01 ~]# imtest localhost 993
S: * OK [CAPABILITY IMAP4rev1 LITERAL+ ID ENABLE STARTTLS
LOGINDISABLED] d01.informatica.escoladeltreball.org Cyrus IMAP
v2.4.18-Fedora-RPM-2.4.18-2.fc24 server ready
Authentication failed. generic failure
```

```
Security strength factor: 0
```

```
[root@d01 ~]# openssl s_client -connect localhost:993
CONNECTED(00000003)
depth=0 C = --, ST = SomeState, L = SomeCity, O = SomeOrganization, OU =
SomeOrganizationalUnit, CN = localhost.localdomain, emailAddress =
root@localhost.localdomain
verify error:num=18:self signed certificate
verify return:1
depth=0 C = --, ST = SomeState, L = SomeCity, O = SomeOrganization, OU =
SomeOrganizationalUnit, CN = localhost.localdomain, emailAddress =
root@localhost.localdomain
verify return:1
---
Certificate chain
 0
s:/C=--/ST=SomeState/L=SomeCity/O=SomeOrganization/OU=SomeOrganizationalUnit
/CN=localhost.localdomain/emailAddress=root@localhost.localdomain

i:/C=--/ST=SomeState/L=SomeCity/O=SomeOrganization/OU=SomeOrganizationalUnit/
CN=localhost.localdomain/emailAddress=root@localhost.localdomain
---
Server certificate
…
---
* OK [CAPABILITY IMAP4rev1 LITERAL+ ID ENABLE AUTH=PLAIN AUTH=LOGIN
SASL-IR] d01.informatica.escoladeltreball.org Cyrus IMAP
v2.4.18-Fedora-RPM-2.4.18-2.fc24 server ready
```

POP: Accés segur amb TLS / StartTLS

Objectius:

- Configurar el servidor POP per acceptar connexions segures via POPs i amb POP i StartTLS.
- Configurar el servei POPs i POP de uw-imap dins de xinetd.

SMTP: Accés al correu segur amb StartTLS

Objectius:

- Exemple connexió amb gmail al port 25 i establir starttls

```
$ telnet pop.gmail.com 25

Trying 66.102.1.108...
Connected to pop.gmail.com.
Escape character is '^]'.
220 smtp.gmail.com ESMTP u62sm10532282wma.15 - gsmtp
HELO localhost
250 smtp.gmail.com at your service
MAIL FROM: usuari
530 5.7.0 Must issue a STARTTLS command first. u62sm10532282wma.15 - gsmtp
STARTTLS
220 2.0.0 Ready to start TLS
```

- Configurar el servidor SMTP per acceptar connexions segures al port 25 amb StartTLS.

OpenVPN: Túnels VPN amb TLS

Objectius:

- Generar certificats de servidor TLS per al host que juga el rol de servidor OpenVPN. Certificats signats per una entitat CA com per exemple Veritat Absoluta.

- Generar certificats client, també de la mateixa CA. Per al host que realitza el paper de client OpenVPN.

Podeu consultar els fitxers d'aquest apartat al github de edtasixm11/tls18.

Generar certificats propis per a VPN

Evidentment no es recomana usar en producció els certificats predefinits que incorpoerava les versions anteriors de openvpn. Actualment podem generar els certificats de servidor i client usant dues estratègies diferents:

❏ La utilitat de generació de keys que proporciona openvpn en el directori samples.
❏ Crear manualment amb openssl les keys i certificats, tenint en compte que han d'incorporar les extensions pertinents.

Observar els certificats predefinits

Primerament observem els certificats predefinits que es proporcionaven en versions anteriors de openvpn (fedora24). Així podrem observar el cn amb el que estan emesos i les extensions que incorporen. El certificat de servidor i el certificat de client són diferents en les extensions que incorporen.

Certificat de servidor:

```
  $ openssl x509 -noout -text -in server.crt
Certificate:
          Data:
          Version: 3 (0x2)
          Serial Number: 1 (0x1)
          Signature Algorithm: sha256WithRSAEncryption
          Issuer: C = KG, ST = NA, L = BISHKEK, O = OpenVPN-TEST, emailAddress = me@myhost.mydomain
          Validity
          Not Before: Oct 22 21:59:52 2014 GMT
          Not After : Oct 19 21:59:52 2024 GMT
          Subject: C = KG, ST = NA, O = OpenVPN-TEST, CN = Test-Server, emailAddress = me@myhost.mydomain
          Subject Public Key Info:
          Public Key Algorithm: rsaEncryption
          Public-Key: (2048 bit)
              ….
          X509v3 extensions:
             X509v3 Basic Constraints:
                  CA:FALSE
             Netscape Cert Type:
                  SSL Server
```

```
            Netscape Comment:
                        OpenSSL Generated Server Certificate
            X509v3 Subject Key Identifier:
                        B3:9D:81:E6:16:92:64:C4:86:87:F5:29:10:1B:5E:2F:74:F7:ED:B1
            X509v3 Authority Key Identifier:
                        keyid:2B:40:E5:C9:7D:F5:F4:96:38:E9:2F:E3:2F:D9:40:64:C9:8E:05:9B
                       DirName:/C=KG/ST=NA/L=BISHKEK/O=OpenVPN-TEST/emailAddress=me@myhost.mydomain
                        serial:A1:4E:DE:FA:90:F2:AE:81
            X509v3 Extended Key Usage:
            TLS Web Server Authentication
            X509v3 Key Usage:
            Digital Signature, Key Encipherment
```

Certificat de client predefinit:

```
# openssl x509 -noout -text -in  client.crt
Certificate:
            Data:
            Version: 3 (0x2)
            Serial Number: 2 (0x2)
            Signature Algorithm: sha256WithRSAEncryption
            Issuer: C=KG, ST=NA, L=BISHKEK, O=OpenVPN-TEST/emailAddress=me@myhost.mydomain
            Validity
            Not Before: Oct 22 21:59:53 2014 GMT
            Not After : Oct 19 21:59:53 2024 GMT
            Subject: C=KG, ST=NA, O=OpenVPN-TEST, CN=Test-Client/emailAddress=me@myhost.mydomain
            Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            Public-Key: (2048 bit)
            X509v3 extensions:
               X509v3 Basic Constraints:
                      CA:FALSE
              X509v3 Subject Key Identifier:
                        D2:B4:36:0F:B1:FC:DD:A5:EA:2A:F7:C7:23:89:FA:E3:FA:7A:44:1D
              X509v3 Authority Key Identifier:
                      keyid:2B:40:E5:C9:7D:F5:F4:96:38:E9:2F:E3:2F:D9:40:64:C9:8E:05:9B
                        DirName:/C=KG/ST=NA/L=BISHKEK/O=OpenVPN-TEST/emailAddress=me@myhost.mydomain
            serial:A1:4E:DE:FA:90:F2:AE:81
```

Generar-ne de nou amb la seva configuració openssl.conf

Fitxers d’exemple en fedora:24:

- /usr/share/doc/openvpn/sample/sample-keys/openssl.conf
- /usr/share/doc/openvpn/sample/sample-keys/gen-sample-keys.sh

podeu veure el contingut d’aquests fitxers a l’annex.

Executar l’script de creació en un directori on hi hagi el fitxer openssl.conf

```
$ gen-sample-keys.sh
```

Crear noves claus manualment

Podem crear claus i certificats manualment amb openssl de la CA, el servidor i de els clients. Això sí, cal que les extesions que incorporen siguin les apropiades. Es pot consultar quines extensions cals mirant l'apartat inicial.

Per poder generar els certificats amb les extensions podem generar un fitxer d'extensions específic per a server i un per a client. Les directives que cal usar en cada cas les podem copiar del fitxer *openssl.conf* de *samples*.

Per exemple, el fitxer ext.server.conf que conté les extensions a usar en el certificat de servidor és:

```
# cat ext.server.conf
basicConstraints       = CA:FALSE
nsCertType             = server
nsComment              = "OpenSSL Generated Server Certificate"
subjectKeyIdentifier   = hash
authorityKeyIdentifier = keyid,issuer:always
extendedKeyUsage            = serverAuth
keyUsage               = digitalSignature, keyEncipherment
```

El fitxer que conté les extensions a usar en el certificat client és per exemple ext.client.conf, amb les següents directives:

```
# cat ext.client.conf
basicConstraints       = CA:FALSE
subjectKeyIdentifier = hash
authorityKeyIdentifier  = keyid,issuer:always
```

Per generar la key, el req i el certificat de servidor (disposem ja de la CA), fem:

```
$ openssl  genrsa -out serverkey.vpn.pem
$ openss req -new -key serverkey.vpn.pem -out serverreq.vpn.pem
$ openssl x509 -CAkey cakey.pem -CA cacert.pem -req -in serverreq.vpn.pem -days
3650 -CAcreateserial -extfile ext.server.conf -out servercert.vpn.pem
```

Per generar la key, el req i el certificat del primer client fem:

```
$ openssl  genrsa -out clientkey.1vpn.pem
$ openssl req -new -key clientkey.1vpn.pem -out clientreq.1vpn.pem
$ openssl x509 -CAkey cakey.pem -CA cacert.pem -req -in clientreq.1vpn.pem -days
3650 -CAcreateserial -extfile ext.client.conf -out clientcert.1vpn.pem
```

Finalment engegar en el servidor i en el client (o clients, cada un d'ells amb el seu propi certificat) el servei openvpn:

```
# systemctl  start openvpn-server@server.service
```

```
# systemctl  start openvpn-client@client.service
```

Túnel amb comandes OpenVPN

```
[root@d01 vpn]# openvpn --remote d02 --dev tun0 --ifconfig 10.4.0.1 10.4.0.2
--tls-server --dh dh2048.pem --ca cacert.pem --cert servercert.pem   --key
serverkey.pem --reneg-sec 60

Mon Mar 27 10:30:25 2017 OpenVPN 2.3.14 x86_64-redhat-linux-gnu [SSL (OpenSSL)]
[LZO] [EPOLL] [PKCS11] [MH] [IPv6] built on Dec  7 2016
Mon Mar 27 10:30:25 2017 library versions: OpenSSL 1.0.2k-fips  26 Jan 2017, LZO
2.08
Mon Mar 27 10:30:25 2017 NOTE: your local LAN uses the extremely common subnet
address 192.168.0.x or 192.168.1.x.  Be aware that this might create routing conflicts if
you connect to the VPN server from public locations such as internet cafes that use the
same subnet.
Mon Mar 27 10:30:25 2017 WARNING: file 'serverkey.pem' is group or others accessible
Mon Mar 27 10:30:25 2017 TUN/TAP device tun0 opened
Mon Mar 27 10:30:25 2017 do_ifconfig, tt->ipv6=0, tt->did_ifconfig_ipv6_setup=0
Mon Mar 27 10:30:25 2017 /usr/sbin/ip link set dev tun0 up mtu 1500
Mon Mar 27 10:30:25 2017 /usr/sbin/ip addr add dev tun0 local 10.4.0.1 peer 10.4.0.2
Mon Mar 27 10:30:25 2017 UDPv4 link local (bound): [undef]
Mon Mar 27 10:30:25 2017 UDPv4 link remote: [AF_INET]192.168.0.22:1194
Mon Mar 27 10:31:55 2017 Initialization Sequence Completed
```

```
[root@d02 vpn]# openvpn --remote d01 --dev tun0 --ifconfig 10.4.0.2 10.4.0.1
--tls-client --ca cacert.pem --cert clientcert.pem --key clientkey.pem --reneg-sec 60

Mon Mar 27 10:31:54 2017 OpenVPN 2.3.14 x86_64-redhat-linux-gnu [SSL (OpenSSL)]
[LZO] [EPOLL] [PKCS11] [MH] [IPv6] built on Dec  7 2016
Mon Mar 27 10:31:54 2017 library versions: OpenSSL 1.0.2k-fips  26 Jan 2017, LZO
2.08
Mon Mar 27 10:31:54 2017 WARNING: No server certificate verification method has
been enabled.  See http://openvpn.net/howto.html#mitm for more info.
Mon Mar 27 10:31:54 2017 WARNING: file 'clientkey.pem' is group or others accessible
Mon Mar 27 10:31:54 2017 TUN/TAP device tun0 opened
Mon Mar 27 10:31:54 2017 do_ifconfig, tt->ipv6=0, tt->did_ifconfig_ipv6_setup=0
Mon Mar 27 10:31:54 2017 /usr/sbin/ip link set dev tun0 up mtu 1500
Mon Mar 27 10:31:54 2017 /usr/sbin/ip addr add dev tun0 local 10.4.0.2 peer 10.4.0.1
Mon Mar 27 10:31:54 2017 UDPv4 link local (bound): [undef]
Mon Mar 27 10:31:54 2017 UDPv4 link remote: [AF_INET]192.168.0.21:1194
Mon Mar 27 10:31:54 2017 WARNING: INSECURE cipher with block size less than 128
bit (64 bit).  This allows attacks like SWEET32.  Mitigate by using a --cipher with a larger
```

```
block size (e.g. AES-256-CBC).
Mon Mar 27 10:31:54 2017 WARNING: INSECURE cipher with block size less than 128
bit (64 bit).  This allows attacks like SWEET32.  Mitigate by using a --cipher with a larger
block size (e.g. AES-256-CBC).
Mon Mar 27 10:31:54 2017 [servidor] Peer Connection Initiated with
[AF_INET]192.168.0.21:1194
Mon Mar 27 10:31:55 2017 Initialization Sequence Completed
```

```
[root@d01 vpn]# nc -kl 60000
hola que tal remot!
```

```
[root@d02 ~]# telnet 10.4.0.1 60000
Trying 10.4.0.1...
Connected to 10.4.0.1.
Escape character is '^]'.
hola que tal remot!
```

Túnel amb Systemctl

```
[root@d01 openvpn]# systemctl start openvpn@server.service

[root@d01 openvpn]# systemctl status openvpn@server.service
● openvpn@server.service - OpenVPN Robust And Highly Flexible Tunneling
Application On server
   Loaded: loaded (/usr/lib/systemd/system/openvpn@.service; disabled; vendor preset:
disabled)
   Active: active (running) since Mon 2017-03-27 11:58:40 CEST; 2h 32min ago
  Process: 4785 ExecStart=/usr/sbin/openvpn --daemon --writepid
/var/run/openvpn/%i.pid --cd /etc/openvpn/ --config %i.conf (co
 Main PID: 4788 (openvpn)
        Tasks: 1 (limit: 512)
   CGroup: /system.slice/system-openvpn.slice/openvpn@server.service
          └─4788 /usr/sbin/openvpn --daemon --writepid /var/run/openvpn/server.pid --cd
/etc/openvpn/ --config server.conf

Mar 27 14:30:52 d01.informatica.escoladeltreball.org openvpn[4788]: GET INST BY REAL: 192.168.0.22:46155 [succeeded]
Mar 27 14:30:52 d01.informatica.escoladeltreball.org openvpn[4788]: client/192.168.0.22:46155 UDPv4 READ [69] from
[AF_INET]192
Mar 27 14:30:52 d01.informatica.escoladeltreball.org openvpn[4788]: client/192.168.0.22:46155 TLS: tls_pre_decrypt, key_id=2, l
Mar 27 14:30:52 d01.informatica.escoladeltreball.org openvpn[4788]: client/192.168.0.22:46155 DECRYPT IV: e3b8b7f9 620af938
7ec
Mar 27 14:30:52 d01.informatica.escoladeltreball.org openvpn[4788]: client/192.168.0.22:46155 DECRYPT TO: 000000b2 2a187bf3
641
Mar 27 14:30:52 d01.informatica.escoladeltreball.org openvpn[4788]: client/192.168.0.22:46155 PID_TEST [0] [SSL-2]
[>EEEEEEEEEE
Mar 27 14:30:52 d01.informatica.escoladeltreball.org openvpn[4788]: client/192.168.0.22:46155 RECEIVED PING PACKET
Mar 27 14:30:52 d01.informatica.escoladeltreball.org openvpn[4788]: PO_CTL rwflags=0x0001 ev=6 arg=0x56075c2dc190
Mar 27 14:30:52 d01.informatica.escoladeltreball.org openvpn[4788]: PO_CTL rwflags=0x0001 ev=7 arg=0x56075c2dc068
Mar 27 14:30:52 d01.informatica.escoladeltreball.org openvpn[4788]: I/O WAIT TR|Tw|SR|Sw [10/0]
```

```
[root@d02 openvpn]# systemctl start openvpn@client.service

[root@d02 openvpn]# systemctl status openvpn@client.service
● openvpn@client.service - OpenVPN Robust And Highly Flexible Tunneling Application
On client
   Loaded: loaded (/usr/lib/systemd/system/openvpn@.service; disabled; vendor preset:
disabled)
   Active: active (running) since Mon 2017-03-27 12:00:14 CEST; 2h 32min ago
  Process: 2312 ExecStart=/usr/sbin/openvpn --daemon --writepid
/var/run/openvpn/%i.pid --cd /etc/openvpn/ --config %i.conf (co
 Main PID: 2313 (openvpn)
        Tasks: 1 (limit: 512)
   CGroup: /system.slice/system-openvpn.slice/openvpn@client.service
           └─2313 /usr/sbin/openvpn --daemon --writepid /var/run/openvpn/client.pid --cd
/etc/openvpn/ --config client.conf

Mar 27 13:00:14 d02.informatica.escoladeltreball.org openvpn[2313]: Data Channel Decrypt: Using 160 bit message hash 'SHA1' for
Mar 27 13:00:14 d02.informatica.escoladeltreball.org openvpn[2313]: Control Channel: TLSv1.2, cipher TLSv1/SSLv3
DHE-RSA-AES256
Mar 27 14:00:14 d02.informatica.escoladeltreball.org openvpn[2313]: VERIFY OK: depth=1, C=ca, ST=catalunya, L=barcelona,
O=veri
Mar 27 14:00:15 d02.informatica.escoladeltreball.org openvpn[2313]: VERIFY OK: depth=0, C=ca, ST=catalunya, L=barcelona,
O=serv
Mar 27 14:00:15 d02.informatica.escoladeltreball.org openvpn[2313]: Data Channel Encrypt: Cipher 'AES-256-CBC' initialized with
Mar 27 14:00:15 d02.informatica.escoladeltreball.org openvpn[2313]: Data Channel Encrypt: Using 160 bit message hash 'SHA1' for
Mar 27 14:00:15 d02.informatica.escoladeltreball.org openvpn[2313]: Data Channel Decrypt: Cipher 'AES-256-CBC' initialized with
Mar 27 14:00:15 d02.informatica.escoladeltreball.org openvpn[2313]: Data Channel Decrypt: Using 160 bit message hash 'SHA1' for
Mar 27 14:00:15 d02.informatica.escoladeltreball.org openvpn[2313]: Control Channel: TLSv1.2, cipher TLSv1/SSLv3
DHE-RSA-AES256
Mar 27 14:32:18 d02.informatica.escoladeltreball.org systemd[1]: Started OpenVPN Robust And Highly Flexible Tunneling
Application
```

```
[root@d01 vpn]# ncat -kl 60000
hola server ncat
que tal?
molt bé gràcies
```

```
[root@d01 vpn]# ncat -kl 60000
hola server ncat
que tal?
molt bé gràcies
```

```
[root@d01 vpn]# ip address show tun0
10: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc
fq_codel state UNKNOWN group default qlen 100
```

```
        link/none
        inet 10.8.0.1 peer 10.8.0.2/32 scope global tun0
        valid_lft forever preferred_lft forever
        inet6 fe80::5551:e2f:73f6:9e38/64 scope link flags 800
        valid_lft forever preferred_lft forever
```

```
[root@d02 ~]# ip address show tun0
6: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc
fq_codel state UNKNOWN group default qlen 100
        link/none
        inet 10.8.0.6 peer 10.8.0.5/32 scope global tun0
        valid_lft forever preferred_lft forever
        inet6 fe80::bcb7:281a:fc12:10da/64 scope link flags 800
        valid_lft forever preferred_lft forever
```

Fitxers de configuració:

servei:

```
[root@d01 openvpn]# cat /usr/lib/systemd/system/openvpn@.service
[Unit]
Description=OpenVPN Robust And Highly Flexible Tunneling Application On %I
After=network.target

[Service]
PrivateTmp=true
Type=forking
PIDFile=/var/run/openvpn/%i.pid
ExecStart=/usr/sbin/openvpn --daemon --writepid /var/run/openvpn/%i.pid --cd
/etc/openvpn/ --config %i.conf

[Install]
WantedBy=multi-user.target
```

server.conf

```
#################################################
# Sample OpenVPN 2.0 config file for            #
# multi-client server.                          #
#                                               #
# This file is for the server side              #
# of a many-clients <-> one-server              #
# OpenVPN configuration.                        #
#                                               #
# OpenVPN also supports                         #
# single-machine <-> single-machine             #
# configurations (See the Examples page         #
# on the web site for more info).            #
#                                            #
# This config should work on Windows         #
# or Linux/BSD systems.  Remember on         #
# Windows to quote pathnames and use         #
# double backslashes, e.g.:                  #
# "C:\\Program Files\\OpenVPN\\config\\foo.key" #
#                                            #
# Comments are preceded with '#' or ';'      #
#################################################

# Which local IP address should OpenVPN
# listen on? (optional)
;local a.b.c.d

# Which TCP/UDP port should OpenVPN listen on?
# If you want to run multiple OpenVPN instances
# on the same machine, use a different port
# number for each one.  You will need to
# open up this port on your firewall.
port 1194

# TCP or UDP server?
;proto tcp
proto udp

# "dev tun" will create a routed IP tunnel,
# "dev tap" will create an ethernet tunnel.
# Use "dev tap0" if you are ethernet bridging
# and have precreated a tap0 virtual interface
# and bridged it with your ethernet interface.
# If you want to control access policies
# over the VPN, you must create firewall
# rules for the the TUN/TAP interface.
# On non-Windows systems, you can give
# an explicit unit number, such as tun0.
# On Windows, use "dev-node" for this.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
;dev tap
dev tun

# Windows needs the TAP-Win32 adapter name
# from the Network Connections panel if you
# have more than one.  On XP SP2 or higher,
# you may need to selectively disable the
# Windows firewall for the TAP adapter.
# Non-Windows systems usually don't need this.
;dev-node MyTap

# SSL/TLS root certificate (ca), certificate
# (cert), and private key (key).  Each client
# and the server must have their own cert and
# key file.  The server and all clients will
# use the same ca file.
#
# See the "easy-rsa" directory for a series
# of scripts for generating RSA certificates
# and private keys.  Remember to use
# a unique Common Name for the server
# and each of the client certificates.
#
# Any X509 key management system can be used.
# OpenVPN can also use a PKCS #12 formatted key file
# (see "pkcs12" directive in man page).
ca cacert.pem
```

```
cert servercert.pem
key serverkey.pem

# Diffie hellman parameters.
# Generate your own with:
#   openssl dhparam -out dh2048.pem 2048
dh dh2048.pem

# Network topology
# Should be subnet (addressing via IP)
# unless Windows clients v2.0.9 and lower have to
# be supported (then net30, i.e. a /30 per client)
# Defaults to net30 (not recommended)
;topology subnet

# Configure server mode and supply a VPN subnet
# for OpenVPN to draw client addresses from.
# The server will take 10.8.0.1 for itself,
# the rest will be made available to clients.
# Each client will be able to reach the server
# on 10.8.0.1. Comment this line out if you are
# ethernet bridging. See the man page for more info.
server 10.8.0.0 255.255.255.0

# Maintain a record of client <-> virtual IP address
# associations in this file.  If OpenVPN goes down or
# is restarted, reconnecting clients can be assigned
# the same virtual IP address from the pool that was
# previously assigned.
ifconfig-pool-persist ipp.txt

# Configure server mode for ethernet bridging.
# You must first use your OS's bridging capability
# to bridge the TAP interface with the ethernet
# NIC interface.  Then you must manually set the
# IP/netmask on the bridge interface, here we
# assume 10.8.0.4/255.255.255.0.  Finally we
# must set aside an IP range in this subnet
# (start=10.8.0.50 end=10.8.0.100) to allocate
# to connecting clients.  Leave this line commented
# out unless you are ethernet bridging.
;server-bridge 10.8.0.4 255.255.255.0 10.8.0.50 10.8.0.100

# Configure server mode for ethernet bridging
# using a DHCP-proxy, where clients talk
# to the OpenVPN server-side DHCP server
# to receive their IP address allocation
# and DNS server addresses.  You must first use
# your OS's bridging capability to bridge the TAP
# interface with the ethernet NIC interface.
# Note: this mode only works on clients (such as
# Windows), where the client-side TAP adapter is
# bound to a DHCP client.
;server-bridge

# Push routes to the client to allow it
# to reach other private subnets behind
# the server.  Remember that these
# private subnets will also need
# to know to route the OpenVPN client
# address pool (10.8.0.0/255.255.255.0)
# back to the OpenVPN server.
;push "route 192.168.10.0 255.255.255.0"
;push "route 192.168.20.0 255.255.255.0"

# To assign specific IP addresses to specific
# clients or if a connecting client has a private
# subnet behind it that should also have VPN access,
# use the subdirectory "ccd" for client-specific
# configuration files (see man page for more info).

# EXAMPLE: Suppose the client
# having the certificate common name "Thelonious"
# also has a small subnet behind his connecting
# machine, such as 192.168.40.128/255.255.255.248.
# First, uncomment out these lines:
;client-config-dir ccd
;route 192.168.40.128 255.255.255.248
# Then create a file ccd/Thelonious with this line:
#   iroute 192.168.40.128 255.255.255.248
# This will allow Thelonious' private subnet to
# access the VPN.  This example will only work
# if you are routing, not bridging, i.e. you are
# using "dev tun" and "server" directives.

# EXAMPLE: Suppose you want to give
# Thelonious a fixed VPN IP address of 10.9.0.1.
# First uncomment out these lines:
;client-config-dir ccd
;route 10.9.0.0 255.255.255.252
# Then add this line to ccd/Thelonious:
#   ifconfig-push 10.9.0.1 10.9.0.2

# Suppose that you want to enable different
# firewall access policies for different groups
# of clients.  There are two methods:
# (1) Run multiple OpenVPN daemons, one for each
#             group, and firewall the TUN/TAP interface
#             for each group/daemon appropriately.
# (2) (Advanced) Create a script to dynamically
#             modify the firewall in response to access
#             from different clients.  See man
#             page for more info on learn-address script.
;learn-address ./script

# If enabled, this directive will configure
# all clients to redirect their default
# network gateway through the VPN, causing
# all IP traffic such as web browsing and
# and DNS lookups to go through the VPN
# (The OpenVPN server machine may need to NAT
# or bridge the TUN/TAP interface to the internet
# in order for this to work properly).
;push "redirect-gateway def1 bypass-dhcp"

# Certain Windows-specific network settings
# can be pushed to clients, such as DNS
# or WINS server addresses.  CAVEAT:
# http://openvpn.net/faq.html#dhcpcaveats
# The addresses below refer to the public
# DNS servers provided by opendns.com.
;push "dhcp-option DNS 208.67.222.222"
;push "dhcp-option DNS 208.67.220.220"

# Uncomment this directive to allow different
# clients to be able to "see" each other.
# By default, clients will only see the server.
# To force clients to only see the server, you
# will also need to appropriately firewall the
# server's TUN/TAP interface.
client-to-client

# Uncomment this directive if multiple clients
# might connect with the same certificate/key
# files or common names.  This is recommended
# only for testing purposes.  For production use,
# each client should have its own certificate/key
# pair.
#
# IF YOU HAVE NOT GENERATED INDIVIDUAL
# CERTIFICATE/KEY PAIRS FOR EACH CLIENT,
# EACH HAVING ITS OWN UNIQUE "COMMON NAME",
# UNCOMMENT THIS LINE OUT.
;duplicate-cn

# The keepalive directive causes ping-like
# messages to be sent back and forth over
# the link so that each side knows when
# the other side has gone down.
# Ping every 10 seconds, assume that remote
# peer is down if no ping received during
# a 120 second time period.
keepalive 10 120

# For extra security beyond that provided
# by SSL/TLS, create an "HMAC firewall"
# to help block DoS attacks and UDP port flooding.
#
# Generate with:
#   openvpn --genkey --secret ta.key
#
# The server and each client must have
# a copy of this key.
# The second parameter should be '0'
# on the server and '1' on the clients.
;tls-auth ta.key 0 # This file is secret

# Select a cryptographic cipher.
# This config item must be copied to
# the client config file as well.
# Note that 2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the ncp-cipher option in the manpage
cipher AES-256-CBC

# Enable compression on the VPN link and push the
# option to the client (2.4+ only, for earlier
# versions see below)
;compress lz4-v2
;push "compress lz4-v2"

# For compression compatible with older clients use comp-lzo
# If you enable it here, you must also
# enable it in the client config file.
;comp-lzo

# The maximum number of concurrently connected
# clients we want to allow.
;max-clients 100

# It's a good idea to reduce the OpenVPN
# daemon's privileges after initialization.
#
# You can uncomment this out on
# non-Windows systems.
;user nobody
;group nobody

# The persist options will try to avoid
# accessing certain resources on restart
# that may no longer be accessible because
# of the privilege downgrade.
persist-key
persist-tun

# Output a short status file showing
# current connections, truncated
# and rewritten every minute.
status openvpn-status.log

# By default, log messages will go to the syslog (or
# on Windows, if running as a service, they will go to
# the "\Program Files\OpenVPN\log" directory).
# Use log or log-append to override this default.
# "log" will truncate the log file on OpenVPN startup,
# while "log-append" will append to it.  Use one
# or the other (but not both).
;log         openvpn.log
;log-append  openvpn.log

# Set the appropriate level of log
# file verbosity.
#
# 0 is silent, except for fatal errors
# 4 is reasonable for general usage
# 5 and 6 can help to debug connection problems
# 9 is extremely verbose
verb 3

# Silence repeating messages.  At most 20
# sequential messages of the same message
# category will be output to the log.
;mute 20

# Notify the client that when the server restarts so it
# can automatically reconnect.
explicit-exit-notify 1
```

client.conf

```
##############################################
# Sample client-side OpenVPN 2.0 config file #
# for connecting to multi-client server.     #
#                                            #
# This configuration can be used by multiple #
# clients, however each client should have   #
# its own cert and key files.                #
#                                            #
# On Windows, you might want to rename this  #
# file so it has a .ovpn extension           #
##############################################

# Specify that we are a client and that we
# will be pulling certain config file directives
# from the server.
client

# Use the same setting as you are using on
# the server.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
;dev tap
dev tun

# Windows needs the TAP-Win32 adapter name
# from the Network Connections panel
# if you have more than one.  On XP SP2,
# you may need to disable the firewall
# for the TAP adapter.
;dev-node MyTap

# Are we connecting to a TCP or
# UDP server?  Use the same setting as
# on the server.
;proto tcp
proto udp

# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.
remote 192.168.0.21 1194
;remote my-server-2 1194

# Choose a random host from the remote
# list for load-balancing.  Otherwise
# try hosts in the order specified.
;remote-random

# Keep trying indefinitely to resolve the
# host name of the OpenVPN server.  Very useful
# on machines which are not permanently connected
# to the internet such as laptops.
#resolv-retry infinite

# Most clients don't need to bind to
# a specific local port number.
nobind

# Downgrade privileges after initialization (non-Windows only)
;user nobody
;group nobody

# Try to preserve some state across restarts.
persist-key
persist-tun

# If you are connecting through an
# HTTP proxy to reach the actual OpenVPN
# server, put the proxy server/IP and
# port number here.  See the man page
# if your proxy server requires
# authentication.
;http-proxy-retry # retry on connection failures
;http-proxy [proxy server] [proxy port #]

# Wireless networks often produce a lot
# of duplicate packets.  Set this flag
# to silence duplicate packet warnings.
;mute-replay-warnings

# SSL/TLS parms.
# See the server config file for more
# description.  It's best to use
# a separate .crt/.key file pair
# for each client.  A single ca
# file can be used for all clients.
ca cacert.pem
cert clientcert.pem
key clientkey.pem

# Verify server certificate by checking that the
# certicate has the correct key usage set.
# This is an important precaution to protect against
# a potential attack discussed here:
#  http://openvpn.net/howto.html#mitm
#
# To use this feature, you will need to generate
# your server certificates with the keyUsage set to
#   digitalSignature, keyEncipherment
# and the extendedKeyUsage to
#   serverAuth
# EasyRSA can do this for you.
#remote-cert-tls server

# If a tls-auth key is used on the server
# then every client must also have the key.
;tls-auth ta.key 1

# Select a cryptographic cipher.
# If the cipher option is used on the server
# then you must also specify it here.
# Note that 2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the ncp-cipher option in the manpage
cipher AES-256-CBC


# Enable compression on the VPN link.
# Don't enable this unless it is also
# enabled in the server config file.
#comp-lzo

# Set log file verbosity.
verb 3

# Silence repeating messages
;mute 20
```

Extensions

Podem definir en els certificats les extensions que incorporen tot explicitant el *KeyUsage* i el *Extended Key Usage*. Podem definir les extensions com a blocs en el fitxer de configuració o podem crear fitxers específics de definició. Una bona manera de practicar-les és fent les pràctiques de openvpn i de ldaps.

Practica extensions: openvpn i ldaps

Com a pràctica per implementar extensions es proposa:

- ❏ Crear un conjunt de claus propis de client i servidor per a openVPN. usant una entitat CA com per exemple VeritatAbsoluta generar un certificat de servidor i varis certificats clients.
  Caldrà que el servidor incorpori unes determinades extensions de servidor i el client les apropiades per fer de client.
  Podeu consultar l'apartat "Generar certificats propis per a VPN" d'aquest mateix HowTo.

- ❏ Una segona pràctica és afegir al certificat del servidor ldap "Subject Alterante Name" per tal de que el certificat no només verifiqui el seu FQDN ldap.edt.org sinó també altres sinònims com per exemple localhost, l'adreça ip 172.17.0.2, etc.

Implementar extensions

Per implementar extensions en un certificat podem fer-ho de dues maneres, amb el fitxer de configuració global o amb un fitxer de configuració específic.

- ❏ Usant un fitxer de configuració openssl.conf podem definir-hi seccions diverses, cada una d'elles descrivint un conjunt d'extensions que volem aplicar segons sigui el tipus de certificat a generar. En executar l'ordre openssl podem usar l'opció -extensions per indicar el nom de les seccions a usar.

- ❏ Creant un fitxer específic que contingui el conjunt d'extensions a aplicar per a un tipus de certificat concret. Amb l'opció de l'ordre openssl -extfile l'indiquem el fitxer de extensions a usar.

Usant un fitxer específic d'extensions

Generar el certificat de servidor ldaps que correrpa en un cpontainer anomenat ldap.edt.org però que també ha de validar connexions realitzades a l’adreça ip 172.17.0.2, 127.0.0.1, al nom loopback i al nom ldaps://mysecureldapserver.org.

Caldrà definir un fitxer de extensions on es defineixi el subjectAltName per a cada un dels noms alternatius amb els que és vol usar el certificat:

Fitxer ext.alternate.conf

```
basicConstraints=CA:FALSE
extendedKeyUsage=serverAuth
subjectAltName=IP:172.17.0.2,IP:127.0.0.1,email:copy,URI:ldaps://mysecureldapserver.org
```

Generar el certificat de servidor:

```
$ openssl x509 -CAkey cakey.pem -CA cacert.pem -req -in serverreq.ldap.pem -days 3650 -CAcreateserial -extfile ext.alternate.conf -out servercert.alternate.pem
Signature ok
subject=/C=ca/ST=barcelona/L=barcelona/O=edt/OU=informatica/CN=ldap.edt.org/emailAddress=ldap@edt.org
Getting CA Private Key
```

Observar que el certificat incorpora aquestes extensions:

```
# openssl x509 -noout -text -in servercert.alternate.pem
Certificate:
        Data:
        Version: 3 (0x2)
        Serial Number:
        c1:5d:a3:8e:7b:1f:62:d4
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=ca, ST=barcelona, L=barcelona, O=edt, OU=informatica, CN=VeritatAbsoluta/emailAddress=veritat@edt.org
        Validity
        Not Before: Mar 28 17:41:40 2019 GMT
        Not After : Mar 25 17:41:40 2029 GMT
        Subject: C=ca, ST=barcelona, L=barcelona, O=edt, OU=informatica, CN=ldap.edt.org/emailAddress=ldap@edt.org
        Subject Public Key Info:
        Public Key Algorithm: rsaEncryption
        Public-Key: (2048 bit)
…
        Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Extended Key Usage:
                TLS Web Server Authentication
        X509v3 Subject Alternative Name:
                IP Address:172.17.0.2, IP Address:127.0.0.1, email:ldap@edt.org, URI:ldaps://mysecureldapserver.org
```

Test del funcionament del certificat ab un client ldapsearch:

```
# ldapsearch -x -LLL -h 172.17.0.2 -s base -b 'dc=edt,dc=org' dn
dn: dc=edt,dc=org

# ldapsearch -x -LLL -Z -h 172.17.0.2 -s base -b 'dc=edt,dc=org' dn
dn: dc=edt,dc=org
```

```
# ldapsearch -x -LLL -ZZ -h 172.17.0.2 -s base -b 'dc=edt,dc=org' dn
dn: dc=edt,dc=org

# ldapsearch -x -LLL -H ldaps://172.17.0.2  -s base -b 'dc=edt,dc=org' dn
dn: dc=edt,dc=org

# ldapsearch -x -LLL -H ldaps://172.17.0.2:636  -s base -b 'dc=edt,dc=org' dn
dn: dc=edt,dc=org

# ldapsearch -x -LLL -ZZ -H ldaps://172.17.0.2 -s base -b 'dc=edt,dc=org' dn
ldap_start_tls: Operations error (1)
        additional info: TLS already started
```

```
# cat /etc/hosts
172.17.0.2 ldap.edt.org mysecureldapserver.org

# ldapsearch -x -LLL -H ldaps://ldap.edt.org -s base -b 'dc=edt,dc=org' dn
dn: dc=edt,dc=org
```

Test del certificat externament:

```
# openssl s_client -connect 172.17.0.2:636 < /dev/null 2> /dev/null | openssl x509 -noout
-tex
Certificate:
        Data:
        Version: 3 (0x2)
        Serial Number:
        c1:5d:a3:8e:7b:1f:62:d4
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=ca, ST=barcelona, L=barcelona, O=edt, OU=informatica, CN=VeritatAbsoluta/emailAddress=veritat@edt.org
        Validity
        Not Before: Mar 28 17:41:40 2019 GMT
        Not After : Mar 25 17:41:40 2029 GMT
        Subject: C=ca, ST=barcelona, L=barcelona, O=edt, OU=informatica, CN=ldap.edt.org/emailAddress=ldap@edt.org
        Subject Public Key Info:
        Public Key Algorithm: rsaEncryption
        Public-Key: (2048 bit)
       ...
        Exponent: 65537 (0x10001)
        X509v3 extensions:
        X509v3 Basic Constraints:
        CA:FALSE
        X509v3 Extended Key Usage:
        TLS Web Server Authentication
        X509v3 Subject Alternative Name:
        IP Address:172.17.0.2, IP Address:127.0.0.1, email:ldap@edt.org, URI:ldaps://mysecureldapserver.org
```

Test des de l’interior del container ldap:

```
$ docker exec -it ldap.edt.org /bin/bash

# ldapsearch -x -LLL -ZZ -h 127.0.0.1 -s base
```

```
dn: dc=edt,dc=org
dc: edt
description: Escola del treball de Barcelona
objectClass: dcObject
objectClass: organization
o: edt.org

[root@ldap docker]# ldapsearch -x -LLL -H ldaps://127.0.0.1 -s base dn
dn: dc=edt,dc=org

[root@ldap docker]# ldapsearch -x -LLL -H ldaps://localhost -s base dn
dn: dc=edt,dc=org
```

Recordeu que en el client ldap.conf s’ha configurat:
TLS_CACERT /opt/docker/cacert.pem

Usant el fitxer global openssl.conf

En el fitxer de configuració openssl.conf (atenció, potser el de /etc/pki/tls es diu openssl.cnf) podem definir seccions que podem cridar específicament per afegir extensions al certificat. De fet podem afegir extensions d’allò que creguem portú i cridar-les en fer el req o en cer el cert.

Atenció:

- Si usem l’ordre openssl CA utilitza el fitxer de configuració gloval /etc/pki/tls/openssl.cnf.
- Si volem practicar podem usar una còpia local del fitxer anomenant-la openssl.conf. Llavors cal usar el paràmetre: -extfile openssl.conf.

Per indicar quina extensió o extensions de les que hi ha definides al fitxer de configuració volem aplicar s’utilitza:

- l’opció -extensions <nom-extensió>.

Exemple de seccions en el fitxer local openssl.conf:

```
[ my_client ]
basicConstraints       = CA:FALSE
subjectKeyIdentifier = hash
authorityKeyIdentifier  = keyid,issuer:always

[ my_server ]
basicConstraints       = CA:FALSE
nsCertType             = server
nsComment              = "OpenSSL Generated Server Certificate"
subjectKeyIdentifier   = hash
authorityKeyIdentifier = keyid,issuer:always
extendedKeyUsage               = serverAuth
keyUsage               = digitalSignature, keyEncipherment

[ my_edt ]
```

```
basicConstraints      = CA:FALSE
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid:always,issuer
basicConstraints = critical,CA:true
keyUsage = cRLSign, keyCertSign
```

Exemple de generació d’un certificat usant el fitxer de configuració local openssl.conf i la secció [my_client]:

```
$ openssl x509 -CAkey ../cakey.pem  -CA ../cacert.pem  -req -in req.pem
-CAcreateserial -extfile /etc/pki/tls/openssl.cnf -extensions my_client  -out cert.pem

$ openssl x509 -noout -text -in cert.pem
        X509v3 extensions:
        X509v3 Basic Constraints:
        CA:FALSE
        X509v3 Subject Key Identifier:
        B0:FA:45:1F:F7:FA:CD:C7:AB:D5:43:C3:03:89:7E:E5:D3:1B:21:FC
        X509v3 Authority Key Identifier:
        keyid:C7:58:F2:1C:98:6E:50:77:C6:7C:4D:2C:AD:39:62:39:7E:DB:72:83
        DirName:/C=ca/ST=barcelona/L=barcelona/O=edt/OU=informatica/CN=VeritatAbsoluta/emailAddress=veritat@edt.org
        serial:B2:29:33:DB:CB:F3:92:A3
```

Exemples de creació de certificats autosignats usant el fitxer de configuració *openssl.conf* i seleccionant explícitament l’extension a usar:

```
# openssl req -new -x509 -key cakey.pem -config openssl.cnf -extensions  usr_cert  -out
cert1.pem
$ openssl x509 -noout -text -in cert1.pem

# openssl req -new -x509 -key cakey.pem -config openssl.cnf -extensions  v3_ca  -out
cert1.pem
$ openssl x509 -noout -text -in cert1.pem
```

Descripció de les Extensions

STANDARD EXTENSIONS

The following sections describe each supported extension in detail.

Basic Constraints.

This is a multi valued extension which indicates whether a certificate is a CA certificate. The first (mandatory) name is CA followed by TRUE or FALSE. If CA is TRUE then an optional pathlen name followed by an non-negative value can be included.

For example:
basicConstraints=CA:TRUE
basicConstraints=CA:FALSE
basicConstraints=critical,CA:TRUE, pathlen:0

A CA certificate must include the basicConstraints value with the CA field set to TRUE. An end user certificate must either set CA to FALSE or exclude the extension entirely. Some software may require the inclusion of basicConstraints with CA set to FALSE for end entity certificates.
The pathlen parameter indicates the maximum number of CAs that can appear below this one in a chain. So if you have a CA with a pathlen of zero it can only be used to sign end user certificates and not further CAs.

Key Usage.

Key usage is a multi valued extension consisting of a list of names of the permitted key usages.

The supporte names are: digitalSignature, nonRepudiation, keyEncipherment, dataEncipherment, keyAgreement, keyCertSign, cRLSign, encipherOnly and decipherOnly.

Examples:
keyUsage=digitalSignature, nonRepudiation
keyUsage=critical, keyCertSign

Extended Key Usage.

This extensions consists of a list of usages indicating purposes for which the certificate public key can be used for,

These can either be object short names of the dotted numerical form of OIDs. While any OID can be used only certain values make sense. In particular the following PKIX, NS and MS values are meaningful:

Value	Meaning
-----	-------
serverAuth	SSL/TLS Web Server Authentication.
clientAuth	SSL/TLS Web Client Authentication.
codeSigning	Code signing.
emailProtection	E-mail Protection (S/MIME).
timeStamping	Trusted Timestamping
msCodeInd	Microsoft Individual Code Signing (authenticode)
msCodeCom	Microsoft Commercial Code Signing (authenticode)
msCTLSign	Microsoft Trust List Signing
msSGC	Microsoft Server Gated Crypto

msEFS	Microsoft Encrypted File System
nsSGC	Netscape Server Gated Crypto

Examples:

```
extendedKeyUsage=critical,codeSigning,1.2.3.4
extendedKeyUsage=nsSGC,msSGC
```

Subject Key Identifier.

This is really a string extension and can take two possible values. Either the word hash which will automatically follow the guidelines in RFC3280 or a hex string giving the extension value to include. The use of the hex string is strongly discouraged.

Example:

```
subjectKeyIdentifier=hash
```

Authority Key Identifier.

The authority key identifier extension permits two options. keyid and issuer: both can take the optional value "always".

If the keyid option is present an attempt is made to copy the subject key identifier from the parent certificate. If the value "always" is present then an error is returned if the option fails.

The issuer option copies the issuer and serial number from the issuer certificate. This will only be done if the keyid option fails or is not included unless the "always" flag will always include the value.

Example:

```
authorityKeyIdentifier=keyid,issuer
```

Subject Alternative Name.

The subject alternative name extension allows various literal values to be included in the configuration file. These include email (an email address) URI a uniform resource indicator, DNS (a DNS domain name), RID (a registered ID: OBJECT IDENTIFIER), IP (an IP address), dirName (a distinguished name) and otherName.

The email option include a special 'copy' value. This will automatically include and email addresses contained in the certificate subject name in the extension.

The IP address used in the IP options can be in either IPv4 or IPv6 format.

The value of dirName should point to a section containing the distinguished name to use as a set of name value pairs. Multi values AVAs can be formed by prefacing the name with a + character.

otherName can include arbitrary data associated with an OID: the value should
be
the OID followed by a semicolon and the content in standard ASN1_generate_nconf(3) format.

Examples:

```
subjectAltName=email:copy,email:my@other.address,URI:http://my.url.here/
subjectAltName=IP:192.168.7.1
subjectAltName=IP:13::17
subjectAltName=email:my@other.address,RID:1.2.3.4
subjectAltName=otherName:1.2.3.4;UTF8:some other identifier

subjectAltName=dirName:dir_sect
[dir_sect]
C=UK
O=My Organization
OU=My Unit
CN=My Name
```

Issuer Alternative Name.

The issuer alternative name option supports all the literal options of subject alternative name. It does not support the email:copy option because that would not make sense. It does support an additional issuer:copy option that will copy all the subject alternative name values from the issuer certificate (if possible).

Example:

```
issuserAltName = issuer:copy
```

Authority Info Access.

The authority information access extension gives details about how to access certain information relating to the CA. Its syntax is accessOID;location where location has the same syntax as subject alternative name (except that email:copy is not supported). accessOID can be any valid OID but only certain values are meaningful, for example OCSP and caIssuers.

Example:

```
authorityInfoAccess = OCSP;URI:http://ocsp.my.host/
authorityInfoAccess = caIssuers;URI:http://my.ca/ca.html
```

CRL distribution points.

This is a multi-valued extension whose options can be either in name:value pair using the same form as subject alternative name or a single value representing a section name containing all the distribution point fields.

For a name:value pair a new DistributionPoint with the fullName field set to the given value both the cRLissuer and reasons fields are omitted in this case.

In the single option case the section indicated contains values for each field.
In this section:

If the name is "fullname" the value field should contain the full name of the distribution point in the same format as subject alternative name.

If the name is "relativename" then the value field should contain a section name whose contents represent a DN fragment to be placed in this field.

The name "CRLIssuer" if present should contain a value for this field in subject alternative name format.

If the name is "reasons" the value field should consist of a comma separated field containing the reasons. Valid reasons are: "keyCompromise", "CACompromise", "affiliationChanged", "superseded", "cessationOfOperation", "certificateHold", "privilegeWithdrawn" and "AACompromise".

Simple examples:

```
crlDistributionPoints=URI:http://myhost.com/myca.crl
crlDistributionPoints=URI:http://my.com/my.crl,URI:http://oth.com/my.crl
```

Full distribution point example:

```
crlDistributionPoints=crldp1_section

[crldp1_section]

fullname=URI:http://myhost.com/myca.crl
CRLissuer=dirName:issuer_sect
reasons=keyCompromise, CACompromise
```

```
[issuer_sect]
C=UK
O=Organisation
CN=Some Name
```

Issuing Distribution Point

This extension should only appear in CRLs. It is a multi valued extension whose syntax is similar to the "section" pointed to by the CRL distribution points extension with a few differences.

The names "reasons" and "CRLissuer" are not recognized.

The name "onlysomereasons" is accepted which sets this field. The value is in the same format as the CRL distribution point "reasons" field.

The names "onlyuser", "onlyCA", "onlyAA" and "indirectCRL" are also accepted
the
values should be a boolean value (TRUE or FALSE) to indicate the value of the corresponding field.

Example:

```
issuingDistributionPoint=critical, @idp_section
[idp_section]
fullname=URI:http://myhost.com/myca.crl
indirectCRL=TRUE
onlysomereasons=keyCompromise, CACompromise
[issuer_sect]
C=UK
O=Organisation
CN=Some Name
```

Certificate Policies.

This is a raw extension. All the fields of this extension can be set by using the appropriate syntax.

If you follow the PKIX recommendations and just using one OID then you just include the value of that OID. Multiple OIDs can be set separated by commas, for example:

```
certificatePolicies= 1.2.4.5, 1.1.3.4
```

If you wish to include qualifiers then the policy OID and qualifiers need to be specified in a separate section: this is done by using the @section syntax instead of a literal OID value.

The section referred to must include the policy OID using the name policyIdentifier, cPSuri qualifiers can be included using the syntax:

```
CPS.nnn=value
```

userNotice qualifiers can be set using the syntax:

```
userNotice.nnn=@notice
```

The value of the userNotice qualifier is specified in the relevant section. This section can include explicitText, organization and noticeNumbers options. explicitText and organization are text strings, noticeNumbers is a comma separated list of numbers. The organization and noticeNumbers options (if included) must BOTH be present. If you use the userNotice option with IE5 then you need the 'ia5org' option at the top level to modify the encoding: otherwise it will not be interpreted properly.

Example:

```
certificatePolicies=ia5org,1.2.3.4,1.5.6.7.8,@polsect
[polsect]
policyIdentifier = 1.3.5.8
CPS.1="http://my.host.name/"
CPS.2="http://my.your.name/"
userNotice.1=@notice
[notice]
explicitText="Explicit Text Here"
organization="Organisation Name"
noticeNumbers=1,2,3,4
```

The ia5org option changes the type of the organization field. In RFC2459 it can only be of type DisplayText. In RFC3280 IA5Strring is also permissible. Some software (for example some versions of MSIE) may require ia5org.

Policy Constraints

This is a multi-valued extension which consisting of the names requireExplicitPolicy or inhibitPolicyMapping and a non negative intger value. At least one component must be present.

Example:

```
policyConstraints = requireExplicitPolicy:3
```

Inhibit Any Policy

This is a string extension whose value must be a non negative integer.

Example:
inhibitAnyPolicy = 2

Name Constraints

The name constraints extension is a multi-valued extension. The name should begin with the word permitted or excluded followed by a ;. The rest of the name and the value follows the syntax of subjectAltName except email:copy is not supported and the IP form should consist of an IP addresses and subnet mask separated by a /.

Examples:
nameConstraints=permitted;IP:192.168.0.0/255.255.0.0
nameConstraints=permitted;email:.somedomain.com
nameConstraints=excluded;email:.com

OCSP No Check

The OCSP No Check extension is a string extension but its value is ignored.
Example:
noCheck = ignored

Exemples / llistats d'aplicació d'extensions

```
basicConstraints=CA:FALSE
extendedKeyUsage=serverAuth,clientAuth,emailProtection
subjectAltName=IP:192.168.1.40,IP:127.0.0.1,IP:127.0.0.1,email:nom1@edt.org,email:nom2@edt.org,URI:https://www.edt.org

#subjectAltName=IP:192.168.1.40
#subjectAltName=IP:127.0.0.1
#subjectAltName=IP:172.17.0.1
#subjectAltName=email:nom1@edt.org
#subjectAltName=email:nom2@edt.org
#subjectAltName=URI:https://www.edt.org
#subjectAltName=dirName:edtorg_sec
#[edtorg_sec]
#C=ca
#ST=barcelona
```

```
#L=barcelona
#O=edt
#OU=inf
#CN=asix-m01/emailAddress=asixm01@edt.org

#keyUsage=digitalSignature,keyEncipherment,nonRepudiation
#keyUsage=decipherOnly
#keyUsage=encipherOnly
#keyUsage=CRLSign
#keyUsage=keyCertSign
#keyUsage=keyAgreement
#keyUsage=nonRepudiation
#keyUsage=dataEncipherment
#keyUsage=keyEncipherment
#keyUsage=digitalSignature

#extendedKeyUsage=serverAuth,clientAuth,emailProtection
#extendedKeyUsage=serverAuth,clientAuth
#extendedKeyUsage=timeStamping
#extendedKeyUsage=codeSigning
#extendedKeyUsage=emailProtection
#extendedKeyUsage=clientAuth
#extendedKeyUsage=serverAuth
```

```
x509v3_config

# ---------------------------------------------------------------------------
# extendedKeyUsage
# ---------------------------------------------------------------------------
extendedKeyUsage=serverAuth,clientAuth,emailProtection
extendedKeyUsage=serverAuth,clientAuth
extendedKeyUsage=timeStamping
extendedKeyUsage=codeSigning
extendedKeyUsage=emailProtection
extendedKeyUsage=clientAuth
#extendedKeyUsage=serverAuth
# ---------------------------------------------
basicConstraints=CA:FALSE
extendedKeyUsage=serverAuth
  X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Extended Key Usage:
                TLS Web Server Authentication
SSL server : Yes
```

```
Netscape SSL server : Yes
CRL signing : Yes
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# ---------------------------------------------
basicConstraints=CA:FALSE
extendedKeyUsage=clientAuth
        X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Extended Key Usage:
                TLS Web Client Authentication
SSL client : Yes
CRL signing : Yes
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# ---------------------------------------------
basicConstraints=CA:FALSE
extendedKeyUsage=serverAuth,clientAuth
        X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication
SSL client : Yes
SSL server : Yes
Netscape SSL server : Yes
CRL signing : Yes
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# -------------------------------------------------
basicConstraints=CA:FALSE
extendedKeyUsage=emailProtection
        X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Extended Key Usage:
                E-mail Protection
S/MIME signing : Yes
S/MIME encryption : Yes
CRL signing : Yes
Any Purpose : Yes
Any Purpose CA : Yes
```

```
OCSP helper : Yes
# --------------------------------------------------
basicConstraints=CA:FALSE
extendedKeyUsage=codeSigning
        X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Extended Key Usage:
                Code Signing
CRL signing : Yes
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# --------------------------------------------------
basicConstraints=CA:FALSE
extendedKeyUsage=timeStamping
        X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Extended Key Usage:
                Time Stamping
CRL signing : Yes
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# --------------------------------------------------
basicConstraints=CA:FALSE
extendedKeyUsage=serverAuth,clientAuth,emailProtection
        X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication, E-mail
Protection
SSL client : Yes
SSL server : Yes
Netscape SSL server : Yes
S/MIME signing : Yes
S/MIME encryption : Yes
CRL signing : Yes
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# --------------------------------------------------
# ---------------------------------------------------------------------------
# keyUsage
# ---------------------------------------------------------------------------
keyUsage=digitalSignature,keyEncipherment,nonRepudiation
keyUsage=decipherOnly
keyUsage=encipherOnly
keyUsage=CRLSign
keyUsage=keyCertSign
keyUsage=keyAgreement
keyUsage=nonRepudiation
keyUsage=dataEncipherment
keyUsage=keyEncipherment
keyUsage=digitalSignature
# ---------------------------------------------
basicConstraints=CA:FALSE
keyUsage=digitalSignature
        X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Key Usage:
                Digital Signature
SSL client : Yes
SSL server : Yes
S/MIME signing : Yes
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# ---------------------------------------------
basicConstraints=CA:FALSE
keyUsage=keyEncipherment
                Exponent: 65537 (0x10001)
        X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Key Usage:
                Key Encipherment
SSL server : Yes
Netscape SSL server : Yes
S/MIME encryption : Yes
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# ---------------------------------------------
basicConstraints=CA:FALSE
keyUsage=dataEncipherment
        X509v3 extensions:
```

```
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Key Usage:
                Data Encipherment
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# ---------------------------------------------
basicConstraints=CA:FALSE
keyUsage=nonRepudiation
        X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Key Usage:
                Non Repudiation
S/MIME signing : Yes
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# ---------------------------------------------
basicConstraints=CA:FALSE
keyUsage=keyAgreement
        X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Key Usage:
                Key Agreement
SSL client : Yes
SSL server : Yes
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# ---------------------------------------------
basicConstraints=CA:FALSE
keyUsage=keyCertSign
        X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Key Usage:
                Certificate Sign
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# ---------------------------------------------
basicConstraints=CA:FALSE
keyUsage=encipherOnly
```

```
        X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Key Usage:
                Encipher Only
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# ---------------------------------------------
basicConstraints=CA:FALSE
keyUsage=decipherOnly
        X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Key Usage:
                Decipher Only
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# ---------------------------------------------
basicConstraints=CA:FALSE
keyUsage=digitalSignature,keyEncipherment,nonRepudiation
        X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Key Usage:
                Digital Signature, Non Repudiation, Key Encipherment
SSL client : Yes
SSL server : Yes
Netscape SSL server : Yes
S/MIME signing : Yes
S/MIME encryption : Yes
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# ---------------------------------------------

# ---------------------------------------------------------------------------
# subjectAltName
# ---------------------------------------------------------------------------
subjectAltName=IP:192.168.1.40
subjectAltName=IP:127.0.0.1
subjectAltName=IP:172.17.0.1
subjectAltName=email:nom1@edt.org
subjectAltName=email:nom2@edt.org
subjectAltName=URI:https://www.edt.org
```

```
subjectAltName=dirName:edtorg_sec
[edtorg_sec]
C=ca
ST=barcelona
L=barcelona
O=edt
OU=inf
CN=asix-m01/emailAddress=asixm01@edt.org
# ---------------------------------------------
basicConstraints=CA:FALSE
extendedKeyUsage=serverAuth,clientAuth,emailProtection
subjectAltName=dirName:edtorg_sec
[edtorg_sec]
C=ca
ST=barcelona
L=barcelona
O=edt
OU=inf
CN=asix-m01/emailAddress=asixm01@edt.org
        X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication, E-mail
Protection
        X509v3 Subject Alternative Name:

DirName:/C=ca/ST=barcelona/L=barcelona/O=edt/OU=inf/CN=asix-m01/emailAddress=
asixm01@edt.org
SSL client : Yes
SSL server : Yes
Netscape SSL server : Yes
S/MIME signing : Yes
S/MIME encryption : Yes
CRL signing : Yes
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# ---------------------------------------------

basicConstraints=CA:FALSE
extendedKeyUsage=serverAuth,clientAuth,emailProtection
subjectAltName=IP:192.168.1.40,IP:127.0.0.1,IP:127.0.0.1,email:nom1@edt.org,email:n
om2@edt.org,URI:https://www.edt.org
        X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication, E-mail
Protection
        X509v3 Subject Alternative Name:
                IP Address:192.168.1.40, IP Address:127.0.0.1, IP Address:127.0.0.1,
email:nom1@edt.org, email:nom2@edt.org, URI:https://www.edt.org
SSL client : Yes
SSL server : Yes
Netscape SSL server : Yes
S/MIME signing : Yes
S/MIME encryption : Yes
CRL signing : Yes
Any Purpose : Yes
Any Purpose CA : Yes
OCSP helper : Yes
# --------------------------------------------
```

Openssl.conf

Atenció: podeu trobar informació més detallada en el manual “extracte Annex A5: OpenSSL Certificats digitals” de la web ASIX-M11. En especial els aparats:

- ❏ “*A5.5 Openssl CA: Gestionar una CA*” que explica com crear els directoris per treballar com a mini CA.
- ❏ “*A5.7 Openssl: Configuració*” que explica i descriu el fitxer de configuració openssl.

La utilitat openssl utilitza el fitxer de configuració global /etc/pki/tls/openssl.cnf. Aquest fitxer està estructurat en directives globals i seccions. A les seccions es defineixen conjunts de directives agrupades sota un nom.

Les seccions són molt útils per definir diferents formats alternatius per fer coses, així per exemple la directiva “default_ca = CA_default ” indica que la definició de les opcions per defecte de la CA es descriuen en una secció anomenada “CA_default”.

Observar del fitxer de configuració les seccions següents:

- [CA_default]   # The default ca section
- [policy_match]
- [policy_anything]
- [req]
  - [req_distinguised_name]
  - [req_attributes]
- [usr_cert]   # These extensions are added when 'ca' signs a request.
- [v3_req]   # Extensions to add to a certificate request
- [v3_ca]   # Extensions for a typical CA

Practicarem els següents aspectes de configuració:

- ❏ [req] definir valors per defecte del DN del req. Per exemple posar valors per defecte del pais, ciutat, empresa, etc. També mirarem com es fa la definició dels tipus de dades permesos.
- ❏ [policy_xxx] Aplicar diferents tipus de policy, match,. anything o una política pròpia.
- ❏ [usr_cert] [v3_req] [v3_ca] personalitzar les seccions ja definides que descriuen com han de ser els certificats finals d’usuari, els de CA i les peticions de request.
- ❏ [my-section] crear seccions pròpies per aplicar en els certificats.

Primerament, però cal crear l’estructura de directoris i la configuració per poder actuar com a CA automatitzadament, tal i com es descriu en l'apartat següent.

Part Prèvia: configuració base openssl CA

Per poder usar l'odre openssl CA i actuar automàticament com una CA cal definir un conjunt de directoris i fitxers propis o bé usar /etc/pki/tls. Per poder fer pràctiques crearem un directori base "test2" des d'on simularem /etc/pki/tls.

Dins del directori base "test2" crear:

- el directori *certs*
- el directori *newcerts*
- el directori *private*
- el directori *crl*
- tenir la clau privada de la ca i posar-la a: ./private/cakey.pem.
- tenir el certificat de la ca amb el nom ./cacert.pem.
- generar un fitxer de text anomenat serial que contingui el valor 01.
- generar un fitxer buit de nom *index.txt*.
- copiar-hi el fitxer de configuració global *openssl.conf* (ull que es diu openssl.cnf). Aquest fitxer caldrà editar-lo.

Editar *openssl.conf* per indicar que el directori actual és el directori base de treball:

```
[ CA_default ]
dir             = .             # Where everything is kept
certs           = $dir/certs            # Where the issued certs are kept
crl_dir         = $dir/crl              # Where the issued crl are kept
database        = $dir/index.txt        # database index file.
#unique_subject = no                    # 'no' allow creation of several certs with same
subject.
new_certs_dir   = $dir/newcerts         # default place for new certs.
certificate     = $dir/cacert.pem       # The CA certificate
serial          = $dir/serial           # The current serial number
crlnumber       = $dir/crlnumber        # the current crl number # must be commented out to
leave a V1 CRL
crl             = $dir/crl.pem          # The current CRL
private_key     = $dir/private/cakey.pem  # The private key
RANDFILE                = $dir/private/.rand  # private random number file
x509_extensions = usr_cert                    # The extensions to add to the cert
```

Configurar req

En el fitxer de configuració openssl.cnf hi ha la secció on es defineix el format i el valor del DN Distinguished Name que identificarà el subject. Aquesta secció s'anomena: *[req_distinguished_name]*.

Exemple amb la configuració per defecte:

```
[ req_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_default             = XX
countryName_min                 = 2
countryName_max                 = 2
stateOrProvinceName             = State or Province Name (full name)
#stateOrProvinceName_default            = Default Province
localityName                    = Locality Name (eg, city)
localityName_default            = Default City
0.organizationName              = Organization Name (eg, company)
0.organizationName_default      = Default Company Ltd
# we can do this but it is not needed normally :-)
#1.organizationName             = Second Organization Name (eg, company)
#1.organizationName_default     = World Wide Web Pty Ltd
organizationalUnitName          = Organizational Unit Name (eg, section)
#organizationalUnitName_default =
commonName                      = Common Name (eg, your name or your server\'s hostname)
commonName_max                  = 64
emailAddress                    = Email Address
emailAddress_max                = 64
```

Podem per exemple personalitzar-lo de manera que alguns dels camps prenguin els valors per defecte que ens interessin:

```
[ req_distinguished_name ]
countryName                     = Country Name
countryName_default             = ca
countryName_min                 = 2
countryName_max                 = 2
stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default             = Barcelona
localityName                    = Locality Name (eg, city)
localityName_default            = Santaco
0.organizationName              = Organization Name (eg, company)
0.organizationName_default      = Escola del Treball
organizationalUnitName          = Organizational Unit Name (eg, section)
organizationalUnitName_default  = informatica
commonName                      = Common Name (eg, your name or your server\'s hostname)
commonName_max                  = 64
emailAddress                    = Email Address
emailAddress_max                = 64
emailAddress_default            = edt@edt.org
```

Verifiquem ara que en fer un request s’utilitzen aquests valors per defecte:

```
$ openssl req -new -config openssl.cnf -key key.pem -out req.pem
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name [ca]:
State or Province Name (full name) [Barcelona]:
Locality Name (eg, city) [Santaco]:
Organization Name (eg, company) [Escola del Treball]:
Organizational Unit Name (eg, section) [informatica]:
Common Name (eg, your name or your server's hostname) []:mycert
Email Address []:
```

```
$ openssl req -noout -subject -in req.pem
subject=C = ca, ST = Barcelona, L = Santaco, O = Escola del Treball, OU = informatica, CN = mycert
```

Configurar policy

Per defecte openssl defineix dues polítiques *policy_match* i *policy_anithing*. La primera exigeix que facin match els camps contryName, stateorProvinceName i organizationname. La segona permet qualsevol valor per a tots els camps. La directiva *policy* defineix quina d'elles és la que està activa.

```
policy          = policy_match
# For the CA policy
[ policy_match ]
countryName             = match
stateOrProvinceName             = match
organizationName                = match
organizationalUnitName  = optional
commonName                      = supplied
emailAddress            = optional

# For the 'anything' policy
# At this point in time, you must list all acceptable 'object'
# types.
[ policy_anything ]
countryName             = optional
stateOrProvinceName             = optional
localityName            = optional
organizationName                = optional
organizationalUnitName  = optional
commonName                      = supplied
emailAddress            = optional
```

Una entitat que fa de CA pot signar certificats de qualsevol tipus a qualsevol? o té unes regles de funcionament. Si som la CA per exemple de l'escola del treball podem tenir una política de signar únicament certificats que formin part de la nostra organització. Si el request no té els primers camps iguals no es signa el certificat.

Exemple amb policy_match on es vol crear un certificat on el subject del request no coincideix amb els camps match amb el del issuer No crea el certificat perquè les localitats Barcelona/barcelona no estan escrites igual:

```
$ openssl x509 -noout -subject -in cacert.pem
subject=C = ca, ST = barcelona, L = barcelona, O = edt, OU = informatica, CN = VeritatAbsoluta, emailAddress = veritat@edt.org

$ openssl req -noout -subject -in req.pem
subject=C = ca, ST = Barcelona, L = Santaco, O = Escola del Treball, OU = informatica
```

```
$ openssl ca -in req.pem -config openssl.cnf -out cert1.pem
```

```
Using configuration from openssl.cnf
Check that the request matches the signature
Signature ok
The stateOrProvinceName field needed to be the same in the
CA certificate (barcelona) and the request (Barcelona)
```

Exemple amb un certificat request totalment diferent de l’organització VeritatAbsoluta:

```
$ openssl req -new -key key.pem -out req2.pem

$ openssl req -noout -subject -in req2.pem
subject=/C=eu/ST=brussels/L=waterloo/O=freedoom/OU=cat/CN=republica

# openssl ca -in req2.pem -config openssl.cnf -out cert1.pem
Using configuration from openssl.cnf
Check that the request matches the signature
Signature ok
The countryName field needed to be the same in the
CA certificate (ca) and the request (eu)
```

PolicyAnithing

```
policy          = policy_anything
```

Repetim l’exemple anterior però ara en el fitxer de configuració openssl.conf s’ha canviat el valor de la directiva policy a policy_anything:

```
$ openssl ca -in req2.pem -config openssl.cnf -out cert1.pem
Using configuration from openssl.cnf
Check that the request matches the signature
Signature ok

# openssl x509 -noout -issuer -subject -in cert1.pem
issuer= /C=ca/ST=barcelona/L=barcelona/O=edt/OU=informatica/CN=VeritatAbsoluta/emailAddress=veritat@edt.org
subject= /C=eu/ST=brussels/L=waterloo/O=freedoom/OU=cat/CN=republica
```

Configurar certs

Per defecte el fitxer *openssl.conf* que tenim inclou dues seccions on es defineixen extensions per als certificats:

- [usr_cert] és la secció on es defineixen les característiques per defcete que tenen els certificats d’usuari.

- [v3_ca] és la secció on es defineixen les característiques que tene per defecte els certificats autosignat, que actuen de CA.

Personalitzar la secció [usr_cert] ampliant les definicions que conté:

```
[ usr_cert ]
basicConstraints=CA:FALSE
nsCertType   = server
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
nsComment                        = "OpenSSL Generated Certificate"
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer
subjectAltName=email:copy
extendedKeyUsage = critical,timeStamping
```

Generar un nou certificat d’usuari i observar que se li apliquen aquestes característiques:

```
$ openssl ca -in req.pem -config openssl.cnf -out cert.pem
$ openssl x509 -noout -text -in cert1.pem
X509v3 extensions:
        X509v3 Basic Constraints:
                CA:FALSE
        Netscape Cert Type:
                SSL Server
        X509v3 Key Usage:
                Digital Signature, Non Repudiation, Key Encipherment
        Netscape Comment:
                OpenSSL Generated Certificate
        X509v3 Subject Key Identifier:
                B0:FA:45:1F:F7:FA:CD:C7:AB:D5:43:C3:03:89:7E:E5:D3:1B:21:FC
        X509v3 Authority Key Identifier:
                keyid:C7:58:F2:1C:98:6E:50:77:C6:7C:4D:2C:AD:39:62:39:7E:DB:72:83
        X509v3 Subject Alternative Name:
                <EMPTY>
        X509v3 Extended Key Usage: critical
                Time Stamping
```

Definir noves seccions

A part de redefinir les seccions que ja incorpora el fitxer de configuració openssl.conf es poden crear noves seccions personalitzades al gust, segons calgui per a prototipus de certificats diferents. per exemple tenim una plantilla (secció) per a certificats d’aules, una per a certificats de profes, una per a certificats d’alumnes, etc.

Exemple de seccions afegides al final del fitxer de configuració openssl.conf. En aquest cas hi ha tres seccions noves [my_client], [my_server] i [my_edt].

```
[ my_client ]
basicConstraints      = CA:FALSE
subjectKeyIdentifier = hash
authorityKeyIdentifier  = keyid,issuer:always
```

```
[ my_server ]
basicConstraints       = CA:FALSE
nsCertType             = server
nsComment              = "OpenSSL Generated Server Certificate"
subjectKeyIdentifier   = hash
authorityKeyIdentifier = keyid,issuer:always
extendedKeyUsage             = serverAuth
keyUsage               = digitalSignature, keyEncipherment

[ my_edt ]
basicConstraints       = CA:FALSE
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid:always,issuer
basicConstraints = critical,CA:true
keyUsage = cRLSign, keyCertSign
```

Per generar un certificat usant una d’aquestes noves seccions podem fer:

```
# openssl ca -in req1.pem -config openssl.cnf -extensions my_server -out cert1.pem
Using configuration from openssl.cnf
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 2 (0x2)
        Validity
        Not Before: Mar 29 19:46:08 2019 GMT
        Not After : Mar 28 19:46:08 2020 GMT
        Subject:
        countryName          = eu
        stateOrProvinceName          = brussels
        localityName         = waterloo
        organizationName             = freedoom
        organizationalUnitName       = cat
        commonName                   = republica
        X509v3 extensions:
        X509v3 Basic Constraints:
        CA:FALSE
        Netscape Cert Type:
        SSL Server
        Netscape Comment:
        OpenSSL Generated Server Certificate
        X509v3 Subject Key Identifier:
        B0:FA:45:1F:F7:FA:CD:C7:AB:D5:43:C3:03:89:7E:E5:D3:1B:21:FC
        X509v3 Authority Key Identifier:
        keyid:C7:58:F2:1C:98:6E:50:77:C6:7C:4D:2C:AD:39:62:39:7E:DB:72:83
DirName:/C=ca/ST=barcelona/L=barcelona/O=edt/OU=informatica/CN=VeritatAbsoluta/emailAddress=veritat@edt.org
        serial:B2:29:33:DB:CB:F3:92:A3
        X509v3 Extended Key Usage:
        TLS Web Server Authentication
        X509v3 Key Usage:
        Digital Signature, Key Encipherment
```

Underconstruction

Documetació RFCs:

- IETF RFC de X509 amb la definició de les extensions: https://tools.ietf.org/html/rfc3280
- https://tools.ietf.org/html/rfc5280#section-4.2.1.12
- Documentació IBM de extensions https://www.ibm.com/support/knowledgecenter/en/SS4T7T_2.4.1/com.ibm.help.seasimplementationguide.doc/SEAS_X509_Exts.html
- Documentació Red Hat de Extensions https://access.redhat.com/documentation/en-US/Red_Hat_Certificate_System/8.0/html/Admin_Guide/Standard_X.509_v3_Certificate_Extensions.html

- https://tools.ietf.org/html/rfc6066
- https://tools.ietf.org/html/rfc4366

Documentació:

> B.3.8. keyUsage
> The Key Usage extension defines the purpose of the key contained in the certificate.
> The Key Usage, Extended Key Usage, and Basic Constraints extensions act together to specify the purposes for which a certificate can be used.
> If this extension is included at all, set the bits as follows:
>
> digitalSignature (0) for SSL client certificates, S/MIME signing certificates, and object-signing certificates.
> nonRepudiation (1) for some S/MIME signing certificates and object-signing certificates.
>
> WARNING
> Use of this bit is controversial. Carefully consider the legal consequences of its use before setting it for any certificate.
> keyEncipherment (2) for SSL server certificates and S/MIME encryption certificates.
> dataEncipherment (3) when the subject's public key is used to encrypt user data instead of key material.
> keyAgreement (4) when the subject's public key is used for key agreement.
> keyCertSign (5) for all CA signing certificates.
> cRLSign (6) for CA signing certificates that are used to sign CRLs.

encipherOnly (7) if the public key is used only for enciphering data. If this bit is set, keyAgreement should also be set.

decipherOnly (8) if the public key is used only for deciphering data. If this bit is set, keyAgreement should also be set.

Table B.31, “Certificate Uses and Corresponding Key Usage Bits” summarizes the guidelines for typical certificate uses.

If the keyUsage extension is present and marked critical, then it is used to enforce the usage of the certificate and key. The extension is used to limit the usage of a key; if the extension is not present or not critical, all types of usage are allowed.

If the keyUsage extension is present, critical or not, it is used to select from multiple certificates for a given operation. For example, it is used to distinguish separate signing and encryption certificates for users who have separate certificates and key pairs for operations.

OID

2.5.29.15

Criticality

This extension may be critical or noncritical. PKIX Part 1 recommends that it should be marked critical if it is used.

Table B.31. Certificate Uses and Corresponding Key Usage Bits

Purpose of Certificate	Required Key Usage Bit
CA Signing	keyCertSign cRLSign
SSL Client	digitalSignature
SSL Server	keyEncipherment
S/MIME Signing	digitalSignature
S/MIME Encryption	keyEncipherment
Certificate Signing	keyCertSign
Object Signing	digitalSignature

B.3.6. extKeyUsage

The Extended Key Usage extension indicates the purposes for which the certified public key may be used. These purposes may be in addition to or in place of the basic purposes indicated in the Key Usage extension.

The Extended Key Usage extension must include OCSP Signing in an OCSP responder's certificate unless the CA signing key that signed the certificates validated by the responder is also the OCSP signing key. The OCSP responder's certificate must be issued directly by the CA that signs certificates the responder will validate.

The Key Usage, Extended Key Usage, and Basic Constraints extensions act together to define the purposes for which the certificate is intended to be used. Applications can use these extensions to disallow the use of a certificate in inappropriate contexts.

Table B.29, “PKIX Extended Key Usage Extension Uses” lists the uses defined by PKIX for this extension, and Table B.30, “Private Extended Key Usage Extension Uses” lists uses privately defined by Netscape.

OID

2.5.29.37

Criticality

If this extension is marked critical, the certificate must be used for one of the indicated purposes only. If it is not marked critical, it is treated as an advisory field that may be used to identify keys but does not restrict the use of the certificate to the indicated purposes.

Table B.29. PKIX Extended Key Usage Extension Uses

Use	OID
Server authentication	1.3.6.1.5.5.7.3.1
Client authentication	1.3.6.1.5.5.7.3.2
Code signing	1.3.6.1.5.5.7.3.3
Email	1.3.6.1.5.5.7.3.4
Timestamping	1.3.6.1.5.5.7.3.8
OCSP Signing	1.3.6.1.5.5.7.3.9[a]

[a] OCSP Signing is not defined in PKIX Part 1, but in RFC 2560, X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP.

Table B.30. Private Extended Key Usage Extension Uses

Use	OID
Certificate trust list signing	1.3.6.1.4.1.311.10.3.1
Microsoft Server Gated Crypto (SGC)	1.3.6.1.4.1.311.10.3.3
Microsoft Encrypted File System	1.3.6.1.4.1.311.10.3.4
Netscape SGC	2.16.840.1.113730.4.1

Table B.8. Key Usage Extension Default Configuration Parameters

Parameter Description

critical Select true to mark this extension critical; select false to mark the extension noncritical.

digitalSignature Specifies whether to allow signing SSL client certificates and S/MIME signing certificates. Select true to set.

nonRepudiation Specifies whether to use for S/MIME signing certificates. Select true to set.

WARNING

Using this bit is controversial. Carefully consider the legal consequences of its use before setting it for any certificate.

keyEncipherment Specifies whether the public key in the subject is used to encipher private or secret keys. This is set for SSL server certificates and S/MIME encryption certificates. Select true to set.

dataEncipherment Specifies whether to set the extension when the subject's public key is used to encipher user data as opposed to key material. Select true to set.

keyAgreement Specifies whether to set the extension whenever the subject's public key is used for key agreement. Select true to set.

keyCertsign Specifies whether the public key is used to verify the signature of other certificates. This setting is used for CA certificates. Select true to set the option.

cRLSign Specifies whether to set the extension for CA signing certificates that sign CRLs. Select true to set.

encipherOnly Specifies whether to set the extension if the public key is only for encrypting data while performing key agreement. If this bit is set, keyAgreement should also be set. Select true to set.

decipherOnly Specifies whether to set the extension if the public key is only for decrypting data while performing key agreement. If this bit is set, keyAgreement should also be set. Select true to set.

OpenVPN Samples

openssl.conf

```
# Heavily borrowed from EasyRSA 3, for use with OpenSSL 1.0.*

####################################################################
[ ca ]
default_ca    = CA_default        # The default ca section

####################################################################
[ CA_default ]

dir           = sample-ca         # Where everything is kept
certs         = $dir              # Where the issued certs are kept
crl_dir       = $dir              # Where the issued crl are kept
database    = $dir/index.txt    # database index file.
new_certs_dir    = $dir                   # default place for new certs.

certificate    = $dir/ca.crt        # The CA certificate
serial        = $dir/serial       # The current serial number
crl           = $dir/crl.pem      # The current CRL
private_key    = $dir/ca.key      # The private key
RANDFILE    = $dir/.rand          # private random number file

x509_extensions    = basic_exts           # The extentions to add to the cert

# This allows a V2 CRL. Ancient browsers don't like it, but anything Easy-RSA
# is designed for will. In return, we get the Issuer attached to CRLs.
crl_extensions    = crl_ext

default_days    = 3650                    # how long to certify for
default_crl_days= 30                      # how long before next CRL
default_md    = sha256            # use public key default MD
preserve    = no                  # keep passed DN ordering

# A few difference way of specifying how similar the request should look
# For type CA, the listed attributes must be the same, and the optional
# and supplied fields are just that :-)
policy        = policy_anything

# For the 'anything' policy, which defines allowed DN fields
[ policy_anything ]
countryName            = optional
stateOrProvinceName    = optional
localityName           = optional
organizationName    = optional
organizationalUnitName    = optional
commonName             = supplied
name                   = optional
emailAddress           = optional

####################################################################
# Easy-RSA request handling
# We key off $DN_MODE to determine how to format the DN
[ req ]
default_bits           = 2048
default_keyfile        = privkey.pem
default_md             = sha256
distinguished_name    = cn_only
x509_extensions        = easyrsa_ca    # The extentions to add to the self signed cert

# A placeholder to handle the $EXTRA_EXTS feature:
#%EXTRA_EXTS%    # Do NOT remove or change this line as $EXTRA_EXTS support requires it
```

```
########################################################################
# Easy-RSA DN (Subject) handling

# Easy-RSA DN for cn_only support:
[ cn_only ]
commonName          = Common Name (eg: your user, host, or server name)
commonName_max              = 64
commonName_default    = changeme

# Easy-RSA DN for org support:
[ org ]
countryName                     = Country Name (2 letter code)
countryName_default             = KG
countryName_min                 = 2
countryName_max                                 = 2

stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default    = NA

localityName                    = Locality Name (eg, city)
localityName_default            = BISHKEK

0.organizationName              = Organization Name (eg, company)
0.organizationName_default    = OpenVPN-TEST

organizationalUnitName          = Organizational Unit Name (eg, section)
organizationalUnitName_default    =

commonName                      = Common Name (eg: your user, host, or server name)
commonName_max                                  = 64
commonName_default              =

emailAddress                    = Email Address
emailAddress_default            = me@myhost.mydomain
emailAddress_max                = 64

########################################################################

[ basic_exts ]
basicConstraints    = CA:FALSE
subjectKeyIdentifier    = hash
authorityKeyIdentifier    = keyid,issuer:always

# The Easy-RSA CA extensions
[ easyrsa_ca ]

# PKIX recommendations:

subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid:always,issuer:always

# This could be marked critical, but it's nice to support reading by any
# broken clients who attempt to do so.
basicConstraints = CA:true

# Limit key usage to CA tasks. If you really want to use the generated pair as
# a self-signed cert, comment this out.
keyUsage = cRLSign, keyCertSign

# CRL extensions.
[ crl_ext ]

# Only issuerAltName and authorityKeyIdentifier make any sense in a CRL.

# issuerAltName=issuer:copy
authorityKeyIdentifier=keyid:always,issuer:always
```

```
# Server extensions.
[ server ]

basicConstraints      = CA:FALSE
nsCertType            = server
nsComment             = "OpenSSL Generated Server Certificate"
subjectKeyIdentifier   = hash
authorityKeyIdentifier = keyid,issuer:always
extendedKeyUsage                = serverAuth
keyUsage              = digitalSignature, keyEncipherment
```

gen-sample-keys.sh

```
#!/bin/sh
#
# Run this script to set up a test CA, and test key-certificate pair for a
# server, and various clients.
#
# Copyright (C) 2014 Steffan Karger <steffan@karger.me>
set -eu

command -v openssl >/dev/null 2>&1 || { echo >&2 "Unable to find openssl. Please make sure openssl is installed and in your path.";
exit 1; }

if [ ! -f openssl.cnf ]
then
        echo "Please run this script from the sample directory"
        exit 1
fi

# Create required directories and files
mkdir -p sample-ca
rm -f sample-ca/index.txt
touch sample-ca/index.txt
echo "01" > sample-ca/serial

# Generate CA key and cert
openssl req -new -newkey rsa:4096 -days 3650 -nodes -x509 \
        -extensions easyrsa_ca -keyout sample-ca/ca.key -out sample-ca/ca.crt \
        -subj "/C=KG/ST=NA/L=BISHKEK/O=OpenVPN-TEST/emailAddress=me@myhost.mydomain" \
        -config openssl.cnf

# Create server key and cert
openssl req -new -nodes -config openssl.cnf -extensions server \
        -keyout sample-ca/server.key -out sample-ca/server.csr \
        -subj "/C=KG/ST=NA/O=OpenVPN-TEST/CN=Test-Server/emailAddress=me@myhost.mydomain"
openssl ca -batch -config openssl.cnf -extensions server \
        -out sample-ca/server.crt -in sample-ca/server.csr

# Create client key and cert
openssl req -new -nodes -config openssl.cnf \
        -keyout sample-ca/client.key -out sample-ca/client.csr \
        -subj "/C=KG/ST=NA/O=OpenVPN-TEST/CN=Test-Client/emailAddress=me@myhost.mydomain"
openssl ca -batch -config openssl.cnf \
        -out sample-ca/client.crt -in sample-ca/client.csr

# Create password protected key file
openssl rsa -aes256 -passout pass:password \
        -in sample-ca/client.key -out sample-ca/client-pass.key

# Create pkcs#12 client bundle
```

```
openssl pkcs12 -export -nodes -password pass:password \
        -out sample-ca/client.p12 -inkey sample-ca/client.key \
        -in sample-ca/client.crt -certfile sample-ca/ca.crt

# Create a client cert, revoke it, generate CRL
openssl req -new -nodes -config openssl.cnf \
        -keyout sample-ca/client-revoked.key -out sample-ca/client-revoked.csr \
        -subj "/C=KG/ST=NA/O=OpenVPN-TEST/CN=client-revoked/emailAddress=me@myhost.mydomain"
openssl ca -batch -config openssl.cnf \
        -out sample-ca/client-revoked.crt -in sample-ca/client-revoked.csr
openssl ca -config openssl.cnf -revoke sample-ca/client-revoked.crt
openssl ca -config openssl.cnf -gencrl -out sample-ca/ca.crl

# Create EC server and client cert (signed by 'regular' RSA CA)
openssl ecparam -out sample-ca/secp256k1.pem -name secp256k1

openssl req -new -newkey ec:sample-ca/secp256k1.pem -nodes -config openssl.cnf \
        -extensions server \
        -keyout sample-ca/server-ec.key -out sample-ca/server-ec.csr \
        -subj "/C=KG/ST=NA/O=OpenVPN-TEST/CN=Test-Server-EC/emailAddress=me@myhost.mydomain"
openssl ca -batch -config openssl.cnf -extensions server \
        -out sample-ca/server-ec.crt -in sample-ca/server-ec.csr

openssl req -new -newkey ec:sample-ca/secp256k1.pem -nodes -config openssl.cnf \
        -keyout sample-ca/client-ec.key -out sample-ca/client-ec.csr \
        -subj "/C=KG/ST=NA/O=OpenVPN-TEST/CN=Test-Client-EC/emailAddress=me@myhost.mydomain"
openssl ca -batch -config openssl.cnf \
        -out sample-ca/client-ec.crt -in sample-ca/client-ec.csr

# Generate DH parameters
openssl dhparam -out dh2048.pem 2048

# Copy keys and certs to working directory
cp sample-ca/*.key .
cp sample-ca/*.crt .
cp sample-ca/*.p12 .
cp sample-ca/*.crl .
```

Format dels directoris de certificats

Quan en les confiuracions es defineixen directoris de certificats, aquests directoris no són simples directoris amb fitxers de certificats sinó que han de tenir un format especial.

Així per exemple en la configuració de /etc/openldap/ldap.conf:

```
 cat /etc/openldap/ldap.conf
#SIZELIMIT    12
#TIMELIMIT    15
#DEREF        never
TLS_CACERTDIR /etc/openldap/certs
SASL_NOCANON    on
URI ldap://ldap/
BASE dc=escoladeltreball,dc=org
```

Encara que en aquest directori hi hagi el fitxer cacert.pem amb el certificat de la ca, no es podrà carregar perquè no carrega fitxers amb el format estàndard:

```
ldapsearch -x -d1 -LLL -ZZ -b 'dc=edt,dc=org' -h ldap.edt.org dn
...
TLS: certdb config: configDir='/etc/openldap/certs' tokenDescription='ldap(0)' certPrefix='' keyPrefix='' flags=readOnly
TLS: cannot open certdb '/etc/openldap/certs', error -8018:Unknown PKCS #11 error.
TLS: skipping 'cacert.pem' - filename does not have expected format (certificate hash with numeric suffix)
TLS: certificate [E=veritat@edt.org,CN=VeritatAbsoluta,OU=informatica,O=edt,L=barcelona,ST=barcelona,C=ca] is not valid - error
-8172:Peer's certificate issuer has been marked as not trusted by the user..
TLS: error: connect - force handshake failure: errno 0 - moznss error -8172
TLS: can't connect: TLS error -8172:Peer's certificate issuer has been marked as not trusted by the user..
ldap_err2string
ldap_start_tls: Connect error (-11)
        additional info: TLS error -8172:Peer's certificate issuer has been marked as not trusted by the user.
ldap_free_connection 1 1
```

Cal que es transformi el directori de manera que cada certificat estigui en un fitxer que té per nom el seu hash. En el directori */etc/pki/tls* hi ha utilitats per realitzar aquesta transformació: