SSL/TLS tanúsítványkezelés

Fájlok titkosítása szimmetrikus kulcs segítségével

A titkosításhoz és visszafejtéshez használható szimmetrikus titkosító algoritmusok

```
$ openssl enc -ciphers

Supported ciphers:
-aes-128-cbc               -aes-128-cfb               -aes-128-cfb1
-aes-128-cfb8              -aes-128-ctr               -aes-128-ecb
-aes-128-ofb               -aes-192-cbc               -aes-192-cfb
-aes-192-cfb1              -aes-192-cfb8              -aes-192-ctr
-aes-192-ecb               -aes-192-ofb               -aes-256-cbc
-aes-256-cfb               -aes-256-cfb1              -aes-256-cfb8
-aes-256-ctr               -aes-256-ecb               -aes-256-ofb
-aes128                    -aes128-wrap               -aes128-wrap-pad
-aes192                    -aes192-wrap               -aes192-wrap-pad
-aes256                    -aes256-wrap               -aes256-wrap-pad
-aria-128-cbc              -aria-128-cfb              -aria-128-cfb1
-aria-128-cfb8             -aria-128-ctr              -aria-128-ecb
-aria-128-ofb              -aria-192-cbc              -aria-192-cfb
-aria-192-cfb1             -aria-192-cfb8             -aria-192-ctr
-aria-192-ecb              -aria-192-ofb              -aria-256-cbc
-aria-256-cfb              -aria-256-cfb1             -aria-256-cfb8
-aria-256-ctr              -aria-256-ecb              -aria-256-ofb
-aria128                   -aria192                   -aria256
-bf                        -bf-cbc                    -bf-cfb
-bf-ecb                    -bf-ofb                    -blowfish
-camellia-128-cbc          -camellia-128-cfb          -camellia-128-cfb1
-camellia-128-cfb8         -camellia-128-ctr          -camellia-128-ecb
-camellia-128-ofb          -camellia-192-cbc          -camellia-192-cfb
-camellia-192-cfb1         -camellia-192-cfb8         -camellia-192-ctr
-camellia-192-ecb          -camellia-192-ofb          -camellia-256-cbc
-camellia-256-cfb          -camellia-256-cfb1         -camellia-256-cfb8
-camellia-256-ctr          -camellia-256-ecb          -camellia-256-ofb
-camellia128               -camellia192               -camellia256
-cast                      -cast-cbc                  -cast5-cbc
-cast5-cfb                 -cast5-ecb                 -cast5-ofb
-chacha20                  -des                       -des-cbc
-des-cfb                   -des-cfb1                  -des-cfb8
-des-ecb                   -des-ede                   -des-ede-cbc
-des-ede-cfb               -des-ede-ecb               -des-ede-ofb
-des-ede3                  -des-ede3-cbc              -des-ede3-cfb
-des-ede3-cfb1             -des-ede3-cfb8             -des-ede3-ecb
-des-ede3-ofb              -des-ofb                   -des3
-des3-wrap                 -desx                      -desx-cbc
```

```
-id-aes128-wrap            -id-aes128-wrap-pad        -id-aes192-wrap
-id-aes192-wrap-pad        -id-aes256-wrap            -id-aes256-wrap-pad
-id-smime-alg-CMS3DESwrap  -idea                      -idea-cbc
-idea-cfb                  -idea-ecb                  -idea-ofb
-rc2                       -rc2-128                   -rc2-40
-rc2-40-cbc                -rc2-64                    -rc2-64-cbc
-rc2-cbc                   -rc2-cfb                   -rc2-ecb
-rc2-ofb                   -rc4                       -rc4-40
-rc5-cbc                   -rc5-cfb                   -rc5-ecb
-rc5-ofb                   -seed                      -seed-cbc
-seed-cfb                  -seed-ecb                  -seed-ofb
```

Titkosítási példa (üres titkosítási algoritmussal)

```
$ echo "Üzenet" > uzenet.txt
$ openssl enc -in uzenet.txt -out titkositott_uzenet.enc -k 12345678
$ cat titkositott_uzenet.enc
Üzenet
```

Titkosítási példa (aes-256-ecb)

```
$ openssl enc -aes-256-ecb -in uzenet.txt -out titkositott_uzenet.enc -k
12345678
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.

$ cat titkositott_uzenet.enc
��-TX]����



$ openssl enc -aes-256-ecb -pbkdf2 -in uzenet.txt -out
titkositott_uzenet.enc -k 12345678

$ cat titkositott_uzenet.enc
Salted__��l]�����s���)<ū��
```

Üzenet visszafejtése

```
$ openssl enc -d -aes-256-ecb -pbkdf2 -in titkositott_uzenet.enc -k 12345678
Üzenet
```

Tanúsítvány kezelés

RSA alapú CA létrehozása (két lépésben):

Kulcs létrehozása:

```
openssl genpkey -algorithm RSA -out ca-rsa-key.pem -pkeyopt
rsa_keygen_bits:2048
```

Önaláírt tanúsítvány létrehozása:

```
$ openssl req -key ca-rsa-key.pem -new -x509 -days 365000 -subj
'/O=EXAMPLE/OU=Teszt RSA CA' -out ca-rsa-cert.pem
```

DSA kulcs alapú CA létrehozása:

Paraméterek létrehozása:

```
$ openssl genpkey -genparam -algorithm DSA -out dsa-params.pem -pkeyopt
dsa_paramgen_bits:2048
```

Kulcs létrehozása:

```
$ openssl genpkey -paramfile dsa-params.pem -out ca-dsa-key.pem
```

A kulcs létrehozható egyetlen utasítással:

```
openssl genpkey -paramfile <(openssl genpkey -genparam -algorithm DSA -
pkeyopt dsa_paramgen_bits:2048) -out ca-dsa-key.pem
```

Önaláírt tanúsítvány létrehozása:

```
openssl req -key ca-dsa-key.pem -new -x509 -days 365000 -subj
'/O=EXAMPLE/OU=Teszt DSA CA' -out ca-dsa-cert.pem
```

EC kulcs alapú CA létrehozása

Paraméterek létrehozása:

```
$ openssl genpkey -genparam -algorithm DSA -out dsa-params.pem -pkeyopt
dsa_paramgen_bits:2048
```

Kulcs létrehozása:

```
openssl genpkey -algorithm EC -out ca-ec-key.pem -pkeyopt
ec_paramgen_curve:prime256v1
```

Önaláírt tanúsítvány létrehozása:

```
openssl req -key ca-ec-key.pem -new -x509 -days 365000 -subj
'/O=EXAMPLE/OU=Teszt EC CA' -out ca-ec-cert.pem
```

Ha egy utasításban szeretnénk létrehozni, akkor a következő utasítások használhatók:

RSA:

```
openssl req -newkey rsa:2048 -noenc -keyout ca-rsa-key.pem -new -x509 -days
365000 -subj '/O=EXAMPLE/OU=Teszt RSA CA' -out ca-rsa-cert.pem
```

DSA:

```
openssl req -newkey dsa:<(openssl genpkey -genparam -algorithm DSA -pkeyopt
dsa_paramgen_bits:2048) -noenc -keyout ca-dsa-key.pem -new -x509 -days
365000 -subj '/O=EXAMPLE/OU=Teszt DSA CA' -out ca-dsa-cert.pem
```

EC:

```
openssl req -newkey ec:<(openssl ecparam -name prime256v1) -noenc -keyout
ca-ec-key.pem -new -x509 -days 365000 -subj '/O=EXAMPLE/OU=Teszt EC CA' -out
ca-ec-cert.pem
```

Szerver tanúsítvány kérelem

RSA szerver tanúsítvány kérelem létrehozása:

```
openssl req -newkey rsa:2048 -noenc -keyout server-rsa-key.pem -new -subj
'/CN=server1' -out server-rsa-req.pem
```

EC szerver tanúsítvány kérelem létrehozása:

```
openssl req -newkey ec:<(openssl ecparam -name prime256v1) -noenc -keyout
server-ec-key.pem -new -subj '/CN=server1' -out server-ec-req.pem
```

Szerver tanúsítvány

Szerver tanúsítvány kérelem aláírása RSA alapú tanúsítványkiadóval:

```
openssl x509 -req -in server-rsa-req.pem -CA ca-rsa-cert.pem -CAkey ca-rsa-
key.pem -CAcreateserial -out server-rsa-cert.pem -days 36500 -extfile
<(printf "subjectAltName = DNS:example.hu, IP:127.0.0.1\nkeyUsage =
digitalSignature, keyEncipherment\nextendedKeyUsage = serverAuth")
```

Tanúsítvány ellenőrzése:

```
$ openssl x509 -in server-rsa-cert.pem -noout -startdate -enddate -issuer -
subject -ext subjectAltName,keyUsage,extendedKeyUsage
```

```
notBefore=Jun 27 09:21:53 2025 GMT
notAfter=Jun  3 09:21:53 2125 GMT
issuer=O=EXAMPLE, OU=Teszt RSA CA
subject=CN=server1
X509v3 Subject Alternative Name:
    DNS:example.hu, IP Address:127.0.0.1
X509v3 Key Usage:
    Digital Signature, Key Encipherment
X509v3 Extended Key Usage:
    TLS Web Server Authentication
```

Szerver tanúsítvány kérelem aláírása EC alapú tanúsítványkiadóval:

```
openssl x509 -req -in server-rsa-req.pem -CA ca-ec-cert.pem -CAkey ca-ec-
key.pem -CAcreateserial -out server-ec-cert.pem -days 36500 -extfile
<(printf "subjectAltName = DNS:example.hu, IP:127.0.0.1\nkeyUsage =
digitalSignature, keyEncipherment\nextendedKeyUsage = serverAuth")
```

Tanúsítvány ellenőrzése:

```
openssl x509 -in server-ec-cert.pem -noout -startdate -enddate -issuer -
subject -ext subjectAltName,keyUsage,extendedKeyUsage
notBefore=Jun 27 09:20:19 2025 GMT
notAfter=Jun  3 09:20:19 2125 GMT
issuer=O=EXAMPLE, OU=Teszt EC CA
subject=CN=server1
X509v3 Subject Alternative Name:
    DNS:example.hu, IP Address:127.0.0.1
X509v3 Key Usage:
    Digital Signature, Key Encipherment
X509v3 Extended Key Usage:
    TLS Web Server Authentication
```

Kliens tanúsítvány

Kliens tanúsítvány kérelem készítése

RSA:

```
openssl req -newkey rsa:2048 -noenc -keyout client-rsa-key.pem -new -subj
'/CN=Teszt Elek/mail=teszt.elek@example.com' -out client-rsa-req.pem
```

EC:

```
openssl req -newkey ec:<(openssl ecparam -name prime256v1) -noenc -keyout
client-ec-key.pem -new -subj '/CN=Teszt Elek/mail=teszt.elek@example.com' -
out client-ec-req.pem
```

Kliens tanúsítvány kérelem aláírása

RSA:

```
$ openssl x509 -req -in client-rsa-req.pem -CA ca-rsa-cert.pem -CAkey ca-
rsa-key.pem -CAcreateserial -out client-rsa-cert.pem -days 36500 -extfile
<(printf "keyUsage = digitalSignature, keyEncipherment\nextendedKeyUsage =
clientAuth")

Certificate request self-signature ok
subject=CN=Teszt Elek, mail=teszt.elek@example.com
```

Tanúsítványok és kulcsok összetartozásának vizsgálata

RSA:

```
$ openssl rsa -in client-rsa-key.pem -noout -modulus | md5sum
d966c3260370c2067d74155b0b3112ac  -

$ openssl x509 -in client-rsa-cert.pem -noout -modulus | md5sum
d966c3260370c2067d74155b0b3112ac  -
```

A két ellenőrző összeg megegyezik, a kulcs és a tanúsítvány összetartozik.

EC:

```
$ openssl ec -in ca-ec-key.pem -pubout | md5sum
read EC key
writing EC key
d26480c8f24e47268004cc3f5a6958bb  -

$ openssl x509 -in ca-ec-cert.pem -noout -pubkey | md5sum
d26480c8f24e47268004cc3f5a6958bb  -
```

A két ellenőrző összeg megegyezik, a kulcs és a tanúsítvány összetartozik.

Kliens tanúsítvány becsomagolása

A PKCS12 formátum (kiterjesztés: .pfx, .p12) titkos kulcs és tanúsítvány tárolására használható. A tároló export/import műveleteihez jelszót kell megadni.

Becsomagolás:

```
$ openssl pkcs12 -export -in  client-rsa-cert.pem -inkey client-rsa-key.pem
-CAfile ca-rsa-cert.pem -out client-rsa.p12 -passout pass:12345678
```

Kicsomagolás:

Csak a privát kulcs:

```
$ openssl pkcs12 -in client-rsa.p12 -nocerts -noenc -passin pass:12345678
```

Csak a tanúsítványok:

```
$ openssl pkcs12 -in client-rsa.p12 -nokeys -noenc -passin pass:12345678
```

További információk:

A PKCS7 formátum (kiterjesztés: .p7b vagy .p7c) tanúsítványokat tartalmaznak (DER vagy PEM formában). Titkos kulcsot nem tartalmazhatnak! Általában tanúsítványláncok (root vagy intermediate) tárolására használjuk. A .p7b DER formátumú, Windows környezetben, a .p7c PEM formátumú, inkább UNIX környezetben használjuk.

Becsomagolás:

```
$ openssl crl2pkcs7 -nocrl -certfile ca-rsa-cert.pem -certfile ca-dsa-
cert.pem -certfile ca-ec-cert.pem -out ca-chain.p7b -outform DER

$ openssl crl2pkcs7 -nocrl -certfile ca-rsa-cert.pem -certfile ca-dsa-
cert.pem -certfile ca-ec-cert.pem -out ca-chain.p7c -outform PEM
```

Kicsomagolás:

```
$ openssl pkcs7 -inform DER -in ca-chain.p7b -print_certs


___
subject=O=EXAMPLE, OU=Teszt RSA CA
issuer=O=EXAMPLE, OU=Teszt RSA CA
-----BEGIN CERTIFICATE-----
MIIDLzCCAhegAwIBAgIUWGRMI4xJNXGystYJY1MiuLY5qHAwDQYJKoZIhvcNAQEL
BQAwJjENMAsGA1UECgwEQVVESTEVMBMGA1UECwwMVGVzenQgUlNBIENBMCAXDTI1
MDYyNzA5MDAwMVoYDzMwMjQxMDI4MDkwMDAxWjAmMQ0wCwYDVQQKDARBVURJMRUw
EwYDVQQLDAxUZXN6dCBSU0EgQ0EwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEK
___

$ openssl pkcs7 -inform PEM -in ca-chain.p7c -print_certs


___
subject=O=EXAMPLE, OU=Teszt RSA CA
issuer=O=EXAMPLE, OU=Teszt RSA CA
-----BEGIN CERTIFICATE-----
MIIDLzCCAhegAwIBAgIUWGRMI4xJNXGystYJY1MiuLY5qHAwDQYJKoZIhvcNAQEL
BQAwJjENMAsGA1UECgwEQVVESTEVMBMGA1UECwwMVGVzenQgUlNBIENBMCAXDTI1
MDYyNzA5MDAwMVoYDzMwMjQxMDI4MDkwMDAxWjAmMQ0wCwYDVQQKDARBVURJMRUw
EwYDVQQLDAxUZXN6dCBSU0EgQ0EwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEK
___
```

Tanúsítvány konverzió

PEM - DER

```
$ openssl x509 -inform PEM -in client-rsa-cert.pem -outform DER -out client-rsa-cert.cer
```

DER – PEM

```
$ openssl x509 -inform DER -in client-rsa-cert.cer -outform PEM -out client-rsa-cert.crt
```

Kliens tanúsítvány megkövetelése Apache szerver esetén

A szerver oldalon telepíteni kell a httpd és a mod_ssl csomagokat. Az engedélyezett kliensek tanúsítványkiadójának CA tanúsítványát hozzá kell adni a /etc/pki/tls/certs/client-ca-chain.crt állományhoz. Amennyiben több tanúsítványkiadót szeretnénk engedélyezni, akkor mindegyik tanúsítványt be kell másolni a fájlba.

Módosítani kell a /etc/httpd/conf.d/ssl.conf állományt az alábbiak szerint:

```
# diff /etc/httpd/conf.d/ssl.conf.orig /etc/httpd/conf.d/ssl.conf
124c124
< #SSLCACertificateFile /etc/pki/tls/certs/ca-bundle.crt
---
> SSLCACertificateFile /etc/pki/tls/certs/client-ca-chain.crt
131,132c131,132
< #SSLVerifyClient require
< #SSLVerifyDepth  10
---
> SSLVerifyClient require
> SSLVerifyDepth  10
```

Az apache szerver újraindítása után csak kliens tanúsítvánnyal lehet hozzáférni az oldalakhoz.

Tesztelés:

```
$ openssl s_client -connect 192.168.110.11:443
Connecting to 192.168.110.11
CONNECTED(00000003)
---
Acceptable client certificate CA names
O=EXAMPLE, OU=Teszt RSA CA
O=EXAMPLE, OU=Teszt EC CA
---
C042A0470F7F0000:error:0A00045C:SSL routines:ssl3_read_bytes:tlsv13 alert certificate required:ssl/record/rec_layer_s3.c:909:SSL alert number 116
```

Megadjuk a kliens tanúsítványt:

```
$ openssl s_client -connect 192.168.110.11:443 -cert ~/client-rsa-cert.pem -
key ~/client-rsa-key.pem
Connecting to 192.168.110.11
CONNECTED(00000003)
---
Acceptable client certificate CA names
O=EXAMPLE, OU=Teszt RSA CA
O=EXAMPLE, OU=Teszt EC CA
---
    Verify return code: 19 (self-signed certificate in certificate chain)
```

Megadjuk a kliens tanúsítványát és elfogadjuk a szerver root-ca állományát:

```
$ openssl s_client -connect 192.168.110.11:443 -cert ~/client-rsa-cert.pem -
key ~/client-rsa-key.pem -CAfile ~/root-ca.pem

Connecting to 192.168.110.11
CONNECTED(00000003)
---
Acceptable client certificate CA names
O=EXAMPLE, OU=Teszt RSA CA
O=EXAMPLE, OU=Teszt EC CA
---
    Verify return code: 0 (ok)
```