

ID kaardi kasutamine autentimiseks



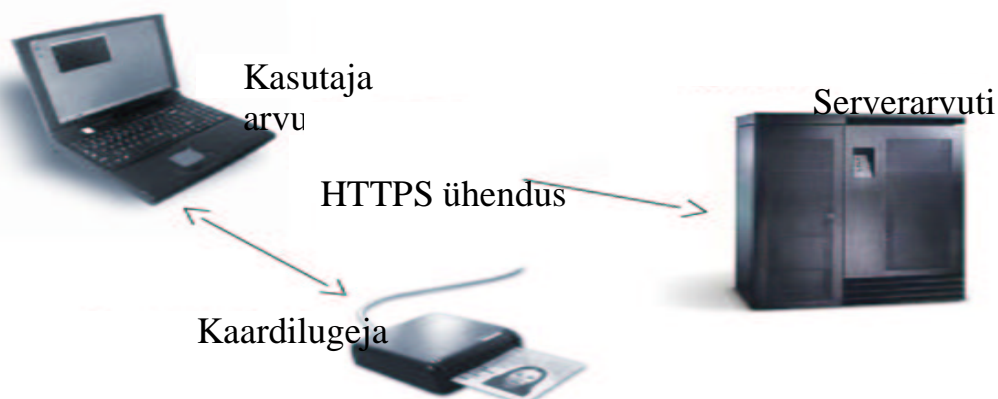
Kõige esimeseks veebiserveriks oli Tim Berners-Lee loodud HTTPd. Seda vabavara veebiserverit saab tõmmata NCSA (the National Center for Supercomputing Applications) saidist ja omal ajal oli ta ka kõige levinum server. Sellest polnud aga küllalt veebimeistrite igapäevaste vajaduste jaoks ja mitmed neist lisasid omatehtud lisamooduleid ja turvapatche sellele programmile. Aastal 1995 koguti loodud lisad parandatud HTTPd versiooni, mida nimetati esialgu lihtsalt parandatud HTTPd -ks "a patchy server", millest hiljem saigi maailma levinuim veebiserver – Apache. Apache algkoodile baseeruvad paljud hiljem loodud spetsiaalproduktid, nagu näiteks IBM HTTP server ja paljud muud. Apache veebiserver sisaldub enamikes Linux -i versioonides.

Üks levinuim krüptograafiateek on OpenSSL, mis baseerub SSLeay teegile, mille autoriteks on Eric A. Young ja Tim Hudson. Peale autorite tööle asumust RSA Corporationis muudeti SSLeay teegi viimane versioon vabavaraks ja selle edasiarenduseks on OpenSSL. Ka OpenSSL on enamikes Linux-i versioonides olemas. Mõni aeg tagasi rakendas USA valitsus OpenSSL teegile ekspordipiiranguid, sest ta sisaldas tugevat krüptograafiat pakkuvaid algoritme, kuid nüüdseks on nende algoritmide patendid aegunud.

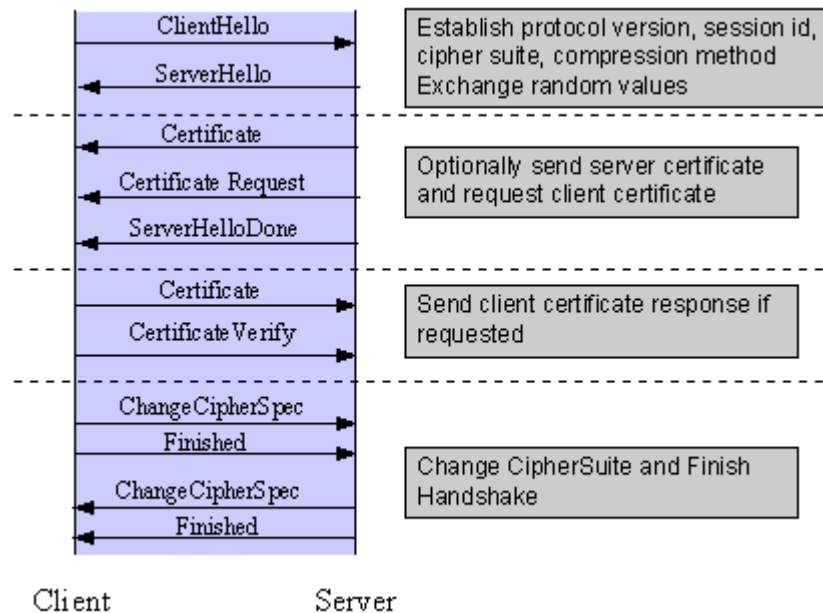


Apache pakub turvalist HTTP/ühendust OpenSSL/teegi abil. OpenSSL võimaluste kasutamiseks tuleb installeerida Apache lisamoodul – *mod_ssl*, mille autoriks on Ralf S. Engelschall. Teine variant on kasutada Apache moodulit *apache_ssl*, mille autoriks on Ben Laurie. Kõik nimetatud tarkvarakomponendid on nagu Linuxgi vabavara ja neid tohib legaalselt kasutada nii isiklikes kui ka kommertssaitides. Nimetatud komponentide installaerimiseks on vaja Linuxit installaerides vaid vajalikud moodulid välja valida.

Üks oluline võimalus ID/kaardi kasutamiseks on autentimisfunktsiooni lisamine oma veebisaidile. Seda kõike saaks teha ka ilma ID/kaardita. Võimalik oleks näiteks väljastada oma saidi kasutajatele ise sertifikaadid ja allkirjastada nad omatehtud CA (certificate authority) sertifikaadiga. Osadel juhtudel on see ka mõistlik, näiteks et lubada juurdepääsu vaid väikesele grupile kasutajatele. ID/kaardiga on aga lihtne lahendada autentimist suure hulga kasutajate puhul, sest riik võtab üle suure hulga administreerimisega seotud ülesandeid.



Selleks on vaja vaid Apache sobivalt konfigureerida. Sertifikaadi või ID-kaardiga autentimise on võimalik vaid HTTPS-ühenduse puhul. HTTPS (ehk HTTP üle SSL) ühendus erineb tavalisest HTTP-ühendusest selles osas, et sõnumi sisu on krüpteeritud vastava sessiooni transpordivõtmega. Sessiooni loomisele eelneb hulk sõnumeid klientprogrammi (veebilehitseja) ja serveri vahel, mille käigus lepitakse kokku kasutatavas protokollis variandis, transpordivõtmes ja paljus muus.



HTTPS variante on mitmeid, aga enamlevinud on SSLv3 ja TLSv1. Netscape oli esimene veebilehitseja, mis toetas HTTPS-ühendust. Enamus levinud veebilehitsejaid tänapäeval toetab HTTPS-i. Konkreetse HTTPS versiooni, kasutatava krüptoalgoritmi, transpordivõtmes ja sertifikaadid lepivad veebilehitseja ja veebiserver üheneduse alguses (nn. ssl handshake) kokku. Sel hetkel saadav server veebilehitsejale oma sertifikaadi, mida kuvatakse kasutajale et ta teaks mis serveriga järgnevas ühendust võetakse. Kasutaja saab vaadelda serveri sertifikaati ja kontrollida kes selle välja andnud on. Kui kasutaja ei usalda seda väljanadjat siis ta katkestab ühenduse. Samas saab ka server nõuda kasutajalt sertifikaati antud veebisaidile juurepääsuks. Server saab kontrollida kasutaja sertifikaadi abil kasutaja kuuluvust mingisse organisatsiooni või firmasse ja vastavalt sellele kas lubada juurepääsu või mitte. Server pakub nüüd kasutajale mitmeid eri krüptoalgoritmi versioone. Kõigile neile on ühine et nad sisaldavad üht asümmeetrilist algoritmi (RSA, DSA jt.) ja üht sümmeetrilist algoritmi (RC4, IDEA jt.). Asümmeetrilise algoritmi abil vahetatakse omavahel genereeritud sümmeetrilise algoritmi transpordivõti, mida järgnevas kasutatakse kogu andmevahetuse krüpteerimiseks.

Apache konfigureerimine

Apache veebiserveri konfiguratsioon sõltub tema versioonist ja Linuxi versioonist. RedHat Linuxi koosseisus olev Apache kasutas kunagi (RedHat 6.x versioonides) kataloogi `/home/httpd` veebilehtede jaoks. Hilisemates RedHat versioonides (7.x ja uuemad) asetsevad veebilehed kataloogis `/var/www/html`. Kui te aga installeerisite Apache uuema versiooni, mille olete tõmmanud internetist, siis asuvad Apache failid tõenäoliselt `/usr/local/apache` kataloogis.

Apache konfiguratsioonifailid asuvad kataloogis `/etc/httpd/conf`. Siin asuvad failid

httpd.conf (peamine konfiguratsioonifail) ja access.conf ning srm.conf. Viimased kaks on reliktid eelmistest versioonidest ning neid ei soovitata enam kasutada.

Apache konfiguratsioonifail sisldab kolmel tasandil direktiive:

Üldised direktiivid

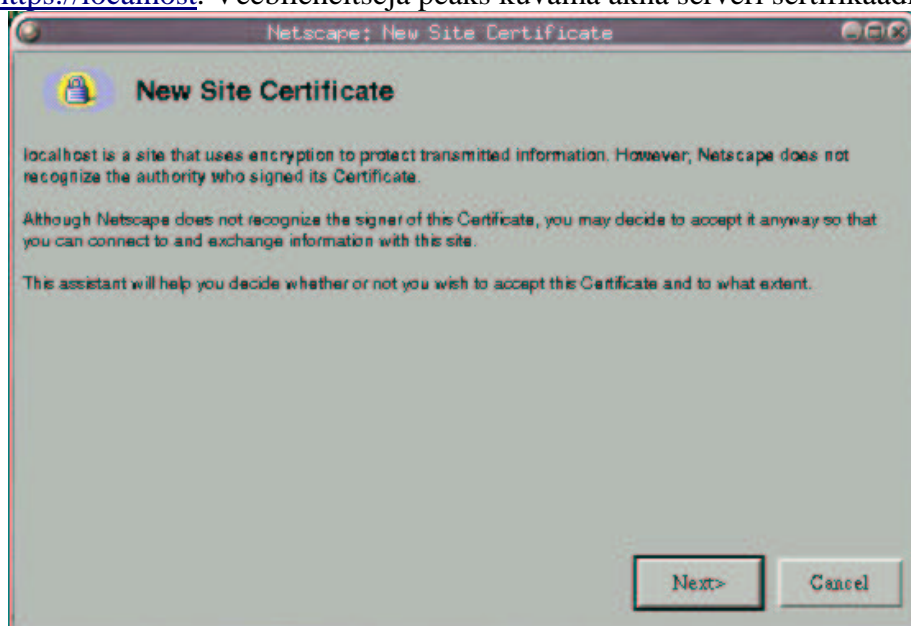
Need direktiivid määravad kogu Apache veebiserveri omadusi. Me ei hakka siin täpsemalt kirjeldama kogu Apache konfigureerimise tehnikat. ID kaardiga autentimise osas on oluline vaid see, et selles osas leiduvad kirjed:

```
<IfDefine HAVE_SSL>
LoadModule ssl_module          modules/libssl.so
</IfDefine>
```

ja

```
<IfDefine HAVE_SSL>
AddModule mod_ssl.c
</IfDefine>
```

Teisiti öeldes kontrollime sellega et teil on installeeritud mod_ssl, mis on vajalik HTTPS ühenduseks. Selle testimiseks startige oma arvutis veebilehitseja ja sisestage URL: <https://localhost>. Veebilehitseja peaks kuvama akna serveri sertifikaadiga.



Virtuaalserveri direktiivid

Apache võimaldab ühe veebiserveri koosseisus kasutada erinevaid konfiguratsioone mis omistatakse erinevatele URL -dele. Kui teie arvutis on võrgukaart, siis peaks teil olema vähemalt kaks erinevat ip aadressi:

- localhost – 127.0.0.1. See võrguühendus on nn. loopback ühendus ja toimib puhtalt ohjurprogrammi sees sama arvuti piires. Selleks pole isegi võrgukaarti vaja ja see on kasulik testimiseks, arendamiseks jne.
- <teie arvuti nimi> - <teie valitud tegelik IP-aadress>. Näiteks 192.168.42.3.

Lisaks sellele tekib teil modemiühendust võttes uus võrguühendus, millele interneti teenuse osutaja omistab dünaamiliselt valitud IP aadressi. Kui te arvuti asub firma intranetis eraldatuna internetist tulemüüriaga, siis võib väljast tema poole pöördumise

IP address erineda teie arvutile konfigureeritust.

Apache konfiguratsioonifailis on tavaliselt defineeritud üks "VirtualHost" HTTPS ühenduse jaoks.

```
<IfDefine HAVE_SSL>
##
## SSL Virtual Host Context
##
# Apache will only listen on port 80 by default. Defining the
virtual server
# (below) won't make it automatically listen on the virtual
server's port.
Listen 443
<VirtualHost _default_:443>
...
```

Selles konfiguratsiooniosas teemegi muudatusi. Siit leiame kirje:

```
# SSL Engine Switch:
# Enable/Disable SSL for this virtual host.
SSLEngine on
```

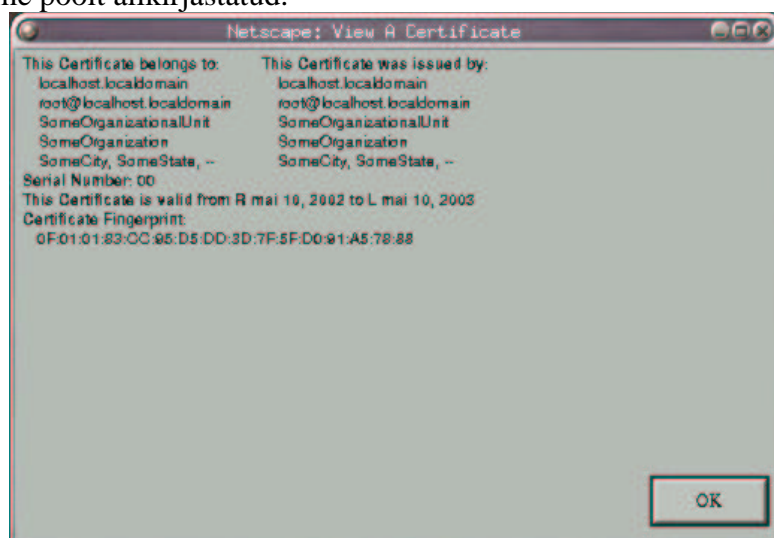
Ima SSL mooduli aktiveerimiseta ei oma kogu järgnev asi mõtet. Edasi veebiserveri enda sertifikaat, salajane võti ja väljanadja sertifikaat ning salajane võti:

SSLCertificateFile /etc/httpd/conf/ssl.crt/server.crt

SSLCertificateKeyFile /etc/httpd/conf/ssl.key/server.key

#SSLCertificateChainFile /etc/httpd/conf/ssl.crt/ca.crt

Direktiiv SSLCertificateFile määrab maili, milles on salvestatud serveri sertifikaat base64 (PEM) kujul ja direktiiv SSLCertificateKeyFile PEM kujul serveri salajase võtme faili. Sellised võtmed on Apache installatioonis igati olemas aga nad sisaldavad sertifikaati mis on väljaantud arvutile nimega "localhost.localdomain" ning on iseenda salajase võtme poolt allkirjastatud.



SSLCertificateChainFile osutab failile, mis sisaldab paljusid tuntud CA sertifikaate. See direktiiv on välja kommenteeritud, sest vaikumisi installeeritud võtmed ei ole ühegi tuntud CA poolt kinnitatud.

Te võite need vaikimisi installeeritud võtmed asendada. Selleks genereerime oma CA ja serveri võtmepaari järgmiste käskudega:

```
# CA.sh -newca
```

Vastavad CA.sh ja CA.pl skriptid on toodud lisatud näidetes. Skript otsib kataloogi demoCA ka kui ta seda ei leia, siis pakub võimalust selline tekitada. Nüüd küsitakse hulk küsimisi ja salvestatakse uus CA sertifikaat failis demoCA/cacert.pem ja CA salajane võti failis demoCA/private/akey.pem.

```
# CA.sh -newreq
```

Küsitakse üldjoones samu asju. Oluline on küsimusele *Common Name* (e.g. *Your name or server's name*) sisestada serveri täielik nimi koos domeeni nimega, näiteks mybox.mydomain.ee. Selle nime leiate failist /etc/hosts. Kohalikus kataloogis tekib fail newreq.pem. Selles asub nii äsja genereeritud salajane võti kui ka sertifikaadi taotlus. Viimase võiks ka otse mingile kommertsiaalsele CA-le saata. Allkirjastame seekord ise:

```
# CA.sh -sign
```

Tekib uus fail newcert.pem, milles asub vaid uus allkirjastatud sertifikaat. Netscape 4.x jaoks poleks kõike seda vaja olnudki. Oleks piisanud ühe ainsa võtmepaari tegemisest. Mozilla ja uuemad Netscape versioonid aga esitavad suuremaid nõudmisi SSL ühenduse turvalisusele. Seepärast teeme eraldi CA võtmepaari ja allkirjastame sellega serveri sertifikaadi. Kuna meie CA pole brauserile tuntud, siis teeme ka väikese skripti selle tuvustamiseks brauserile. See muidugi ei paranda olukorda selles suhtes et ka need võtmed pole mingi tuntud CA poolt kinnitatud aga te võite hiljem oma sertifikaadi vastavale CA-le (Versign, Thawte jt.) allkirjastamiseks saata. See muidugi maksab "natuke".

Tekitame nüüd alamkataloogi /etc/httpd/server ja kopeerime tehtud failid sellesse kataloogi. Siin on lähtunud oletusest et viibisite /root kataloogis võtmeid tehes.

```
# mkdir /etc/httpd/conf/server
# cd /etc/httpd/conf/server
# cp /root/demoCA/cacert.pem .
# cp /root/demoCA/private/akey.pem .
# cp /root/newcert.pem ./server_cert.pem
# cp /root/newreq.pem ./server_key.pem
# openssl x509 -in cacert.pem -out cacert.der -outform DER
```

Editeerime nüüd näiteks vi -ga server_key.pem -i ja eemaldame sealt sertifikaadi taotluse. Järgi jääb vaid osa ----RSA PRIVATE KEY----. Viimase käsuga konverteerisime oma CA sertifikaadi DER kujule, mis on vajalik tema tuvustamiseks brauserile. Selleks teeme veel väikese CGI skripti, *ca_cert.cgi*, mille abil brauser CA sertifikaadi sobiva mime tüübiga veebiserveril kätte saab ja instaleerime ta harilikku CGI kataloogi /var/www/cgi-bin.

```
#!/bin/sh
```

```
CACERT="/etc/httpd/conf/server/cacert.der"
```

```
if [ -r ${CACERT} ] ; then
```

```
    echo "Content-Type: application/x-x509-ca-cert"
```

```
    echo
```

```
    cat ${CACERT}
```

```
else
    echo "Content-Type: text/plain"
    echo
    echo "Error: Can access CA Certificate";
fi
```

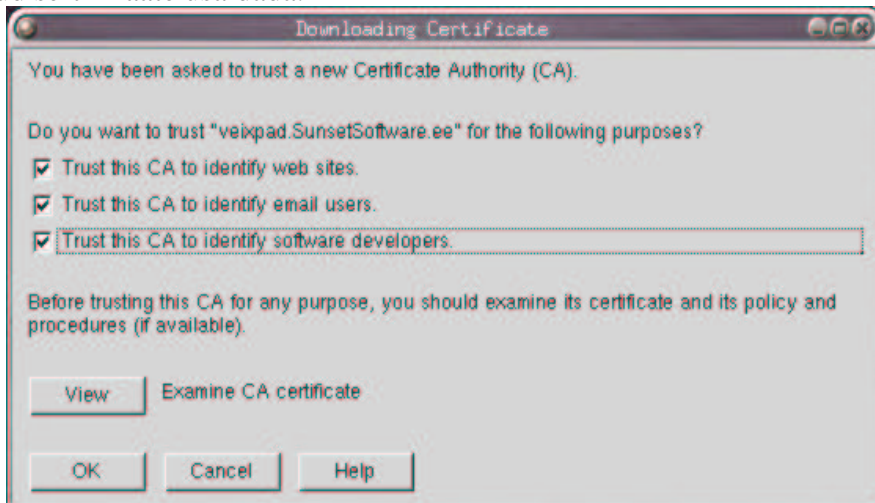
Siis muudame olemasolevad serveri võtmete kirjed selliseks:

```
SSLCertificateFile /etc/httpd/conf/server/server_cert.pem
SSLCertificateKeyFile /etc/httpd/conf/server/server_key.pem
```

Restardime Apache veebiserveri. Kuna me loodud salajane võti oli kaitstud parooliga siis küsib Apache startides salajase võtme parooli:

```
# /etc/rc.d/init.d/httpd stop
# /etc/rc.d/init.d/httpd start
Enter pass phrase: salakala
Ok: Pass Phrase Dialog successful.
```

Tutvustame nüüd brauserile oma uut CA-d. Sellesks suuname brauseri URL-le: http://localhost/cgi-bin/ca_cert.cgi. Brauser küsib nüüd ka me tahame ikka seda sertifikaati CA sertifikaadina usaldada ja sellest tulenevalt ka kõiki tema poolt väljaantud sertifikaate usaldada.



Kinnitame oma usaldust vajutades nupule OK. Kui nüüd suunata veebilehitseja oma arvuti URL-le <https://localhost> siis kuvatakse juba teie valitud arvuti nime.



Edasi tahaksime lisada oma veebisaidile ID-kaardiga autenimist. Selleks tõmbame kõigepealt AS Sertifitseerimiskeskuse veebist ID kaardi juursertifikaadid:

Juursertifikaat: <http://www.sk.ee/certs/JUUR-SK.crt>

ID kaardi sertifikaatide väljaandja sertifikaat: <http://www.sk.ee/certs/ESTEID-SK.crt>

Tühistusnimekiri: <http://www.sk.ee/crls/esteid/esteid.crl>

Need failid on paraku DER kujul ja tuleb seega Apache jaoks PEM kujule konverteerida:

```
# openssl x509 -in JUUR-SK.crt -inform DER -out JUUR-SK.pem  
# openssl x509 -in ESTEID-SK.crt -inform DER -out ESTEID-SK.pem  
# rm *.crt  
# mv JUUR-SK.pem JUUR-SK.crt  
# mv ESTEID-SK.pem ESTEID-SK.crt
```

Tekitame nende sertifikaatide jaoks omaette kataloogi ja installeerime nad sinna. Kuna Apache vajab sertide kasutamiseks failidele viitavaid räsikoodidest viiteid, siis kopeerime olemasolevate sertide kataloogist sobiva Makefile ja tekitame need viited.

```
# mkdir /etc/httpd/conf/esteid_cert  
# mv *.crt /etc/httpd/conf/esteid_cert  
# cd /etc/httpd/conf/esteid_cert  
# cp ../ssl.crt Makefile.crt  
# make -f Makefile.crt
```

Nüüd kirjeldame Apachele (failis httpd.conf) uued kasutaja sertifikaatide kontrolliks sobivad CA sertifikaadid:

```
SSLCACertificatePath /etc/httpd/conf/esteid_cert/
```

Installeerime ka tühistusnimekirja:

```
# openssl crl -in esteid.crl -inform DER -out esteid.pem  
# rm esteid.crl  
# mv esteid.pem esteid.crl  
# mkdir /etc/httpd/conf/esteid_crl  
# mv esteid.crl /etc/httpd/conf/esteid_crl  
# cd /etc/httpd/conf/esteid_crl
```

```
# cp ../ssl.crl/Makefile.crl .
# make -f Makefile.crl
```

Nüüd kirjeldame ka failis httpd.conf tühistusnimekirjade asukohta.

```
SSLCARevocationPath /etc/httpd/conf/esteid_crl/
```

Tühistusnimekirjad muutuvad ja neid tuleks perioodiliselt uuendada. Nimekirja saaks tõmmata automaatselt näiteks *wget* programmiga. Selle jaoks võiks teha väikese shell skripti mis uue tühistusnimekirja tõmbab, PEM formaati teisendab ja vajalikku kohta installeerib ning selle shell skripti programmi *cron* abil iga öö startida.

Apache oskab kontrollida sertifikaate mingite atribuutide abil. Näiteks lugeda sertifikaadist kasutaja CN (nimi) või O (organisatsioon) jne. Selleks pole ID kaartide puhul vajadust, kuna need väärtused on kõigil sertifikaatidel samad. Ainus mis meid hetkel huvitab on kontrollida et kasutaja on korrektse ID kaardi omanik. Selleks sisestame httpd.conf faili veel kirjed:

```
SSLVerifyClient require
SSLVerifyDepth 2
```

Sellega saavutame et veebiserver nõuab kasutaja veebilehitsejalt sertifikaatide esitamist ja eelnevalt oleme juba defineerinud et me aktsepteerime vaid ID kaardi juursertifikaatide poolt kinnitatud sertifikaate ning lisanud ka tühistusnimekirja sertifikaadi heteklise kehtivuse kontrolliks. Oluline on et ID kaartide puhul oleks CA sertifikaadi kontroll lubatud ainult 2 tasemeni (SSLVerifyDepth 2). ID kaartidel olevad sertifikaadid on kinnitatud ESTEID-SK.crt sertifikaadiga ja viimane JUUR-SK.crt sertifikaadiga. Kui lubaksime rohkem tasemeid, siis oleks võimalik mingil kasutajal genereerida ise võtmepaar sisestades seal oma nimeks näiteks "Bill Gates", allkirjastada see sertifikaat oma ID kaardiga ja saata teie veebiserverile. Sellise vea vastu on tundlik näiteks IT veebilehitseja.

Kui me ei taha nõuda ID kaarti kogu veebisaidi jaoks vaid ainult osale sellest, siis defineerime alamkataloogi, millele juurdepääsuks on vajalik ID kaart:

```
<Directory "/home/html/secure">
    SSLVerifyClient require
    SSLVerifyDepth 10
</Directory>
```

Tähelepanu tuleb osutada sellele, et SSL autentimisega kaitstud kataloog ei oleks hariliku HTTP ühendusega kasutatav, s.o. Ta ei tohi olla /var/www/html alamkataloog. Et päris kindel olla võime kas harilku HTTP (port 80) üldse keelata või siis HTTPS ühenduse jaoks uue serveri algkataloogi defineerida. Selleks muudame direktiivi DocumentRoot (ainult HTTPS VirtualHost osas) näiteks sedasi:

```
<VirtualHost _default_:443>
# General setup for the virtual host
DocumentRoot "/var/www/secure"
```

Kui kasutaja on saatnud veebiserverile vajaliku sertifikaadi, mis osutus korrektseks ID kaardi sertifikaadiks ja ei olnud tühistusnimekirjas siis lubab veebiserver juurepääsu soovitud kataloogile. Veebiserver defineerib tekitatud võrguühenduse jaoks järgmised ümbruskonnamuutujad, mida saaks kasutada teie serverprogrammist või CGI-st:

<i>Muutuja</i>	<i>Tüüp</i>	<i>Kirjeldus</i>
HTTPS	flag	Märgib et HTTPS-i kasutatakse

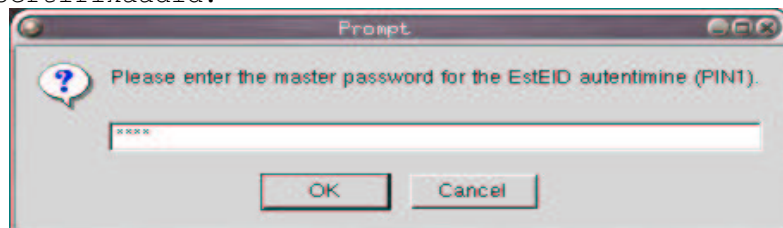
<i>Muutuja</i>	<i>Tüüp</i>	<i>Kirjeldus</i>
SSL_PROTOCOL	string	SSL protoollo versioon: SSLv2, SSLv3, TLSv1
SSL_SESSION_ID	string	SSL sessiooni tunnus hex kujul
SSL_CIPHER	string	Kasutatud krüptoalgoritmi nimi
SSL_CIPHER_EXPORT	string	True, kui tegemist on tugeva krüptoalgoritmiga
SSL_CIPHER_USEKEYSIZE	number	Krüptoalgoritmi võtme pikkus bittides (tegelikult kasutatud)
SSL_CIPHER_ALGKEYSIZE	number	Krüptoalgoritmi võtme pikkus bittides (võimalik)
SSL_VERSION_INTERFACE	string	mod_ssl moduli versioon
SSL_VERSION_LIBRARY	string	OpenSSL teegi versioon
SSL_CLIENT_M_VERSION	string	Kasutaja sertifikaadi versioon
SSL_CLIENT_M_SERIAL	string	Kasutaja sertifikaadi seerianumber
SSL_CLIENT_S_DN	string	Kasutaja täielik nimi sertifikaadist (DN)
SSL_CLIENT_S_DN_x509	string	Kasutaja stringikujuline nimi sertifikaadist (DN)
SSL_CLIENT_I_DN	string	Väljaandja täielik nimi sertifikaadist (DN)
SSL_CLIENT_I_DN_x509	string	Väljaandja stringikujuline nimi sertifikaadist (DN)
SSL_CLIENT_V_START	string	Sertifikaadi kehtivuse alguskuupäev
SSL_CLIENT_V_END	string	Sertifikaadi kehtivuse lõppkuupäev
SSL_CLIENT_A_SIG	string	Sertifikaadi allkirjastamise algoritm
SSL_CLIENT_A_KEY	string	Sertifikaadi avaliku võtme algoritm
SSL_CLIENT_CERT	string	PEM kodeeringus kasutaja sertifikaat
SSL_CLIENT_CERT_CHAIN	string	PEM kodeeringus kasutaja sertifikaadi väljanadjate sertifikaadid
SSL_CLIENT_VERIFY	string	Sertifikaadi kontrolli tulemus: NONE, SUCCESS, GENEROUS or FAILED:reason
SSL_SERVER_M_VERSION	string	Serveri sertifikaadi versioon
SSL_SERVER_M_SERIAL	string	Serveri sertifikaadi seerianumber
SSL_SERVER_S_DN	string	Serveri täielik nimi sertifikaadist (DN)
SSL_SERVER_S_DN_x509	string	Serveri stringikujuline nimi sertifikaadist (DN)

<i>Muutuja</i>	<i>Tüüp</i>	<i>Kirjeldus</i>
SSL_SERVER_I_DN	string	Väljaandja täielik nimi sertifikaadist (DN)
SSL_SERVER_I_DN_x509	string	Väljaandja stringikujuline nimi sertifikaadist (DN)
SSL_SERVER_V_START	string	Sertifikaadi kehtivuse alguskuupäev
SSL_SERVER_V_END	string	Sertifikaadi kehtivuse lõppkuupäev
SSL_SERVER_A_SIG	string	Sertifikaadi allkirjastamise algoritm
SSL_SERVER_A_KEY	string	Sertifikaadi avaliku võtme algoritm
SSL_SERVER_CERT	string	PEM kodeeringus serveri sertifikaat

Nüüd teeme väikese CGI skripti kuvamaks ümbruskonnamuutujaid ja installeerime selle kataloogi /var/www/cgi-bin:

```
#!/bin/bash
echo "Content-Type: text/html"
echo
echo "SSL Keskkonnamuutujad"
echo "<html>"
echo "<head><title>SSL Keskkonnamuutujad</title></head>"
echo "<body>"
echo "<h1>SSL Keskkonnamuutujad</h1>"
echo `env`
echo "</body>"
echo "</html>"
```

Restardime Apache veebiserveri nii nagu eelnevalt kirjeldatud ja stardime veebilehitseja. Veebilehitsejaks sobib Netscape 4.x, Mozilla ja uuemad Netscaped. Installeerime sobiva PKCS#11 ohjurprogrammi valides menüüst Communicator -> Tools -> Security Info -> Cryptographic modules. Kui vaja installeerime ID kaardi PKCS#11 ohjurprogrammi vajutades nupule Add ja sisestades dialoogis mooduli nimeks EstEID ning teegi asukohaks näiteks /usr/local/lib/libesteid.so. Brauser küsib kaardi PIN -e ja loeb kaardilt sertifikaadid.



Suuname nüüd brauseri aadressile https://localhost/cgi-bin/ssl_env.cgi. Brauser näitab serveri sertifikaati ja pakub ID kaardi autentimise sertifikaati serverile saatmiseks. Vajutame nupule OK ja SSL ühendus ongi loodud ning brauseri aknasse kuvatakse SSL-sessiooni keskkonnamuutujad. Neid saaks nüüd kasutada mingi oma programmi turvalahenduse jaoks. Kasutaja on juba Apache poolt autentitud. Keskkonnamuutujast loeme kasutaja täieliku nime (SSL_CLIENT_S_DN) ja sellest otsime üles kasutaja isikukoodi. Viimase abil lubame kasutajal juurepääsu antud isiku

andmetele.

Arvutikasutaja veebisaidist saab tõmmata ettevalmistatud ID kaardi sertifikaadid, Apache konfiguratsioonifaili, näidis CGI ja ID kaardi ohjurprogrammi. Tähelepanu tuleks osutada ka sellele, et teie kasutatud tühistusnimekiri oleks piisavalt värske. Vahel annab brauser väga raskesti mõistetava vea näiteks "The server has rejected your certificate as expired". Kui aga vaadata logisse failis /etc/httpd/logs/error_log, siis selgub et hoopis tühistusnimekiri (CRL) on aegunud. Seda saab alati kontrollida käsuga:

```
# openssl crl -in esteid.crl -text | more
```

Kuvatud teksti alguses on kaks kuupäeva: *Last Update* ja *Next Update*. Kui *Next Update* on juba minevikus siis tuleks kindlasti uus tühistusnimekiri hankida.

AS Sertifitseerimiskeskus pakub ka üksikute sertifikaatide staatuse päringuid OCSP protokollil abil, kuid selle kasutamiseks oleks vaja sobiv CGI või Apache moodul kirjutada, mis OCSP päringuid koostab, saadab ja töötleb.

Lihntne install

Arvutikasutaja saidist <siia tuleb URL> saate tõmmata oma arvutisse arhiivi esteid-apache.tgz. Selle pakime lahti ja installeerime root kasutajana sedasi:

```
$ su - root <küsib passwordi>
# tar xvzf esteid-apache.tgz
# cd samples
# cp -r server /etc/httpd/conf
# cp -r esteid_cert /etc/httpd/conf
# cp -r esteid_crl /etc/httpd/conf
# mv /etc/httpd/conf/httpd.conf /etc/httpd/conf/httpd.conf.orig
# cp httpd.conf /etc/httpd/conf
# cp *.cgi /var/www/cgi-bin
# mkdir /var/www/secure
# cp index.html /var/www/secure
# cp libestid-pkcs11-authonly.so /usr/local/lib
# /etc/rc.d/init.d/http stop
# /etc/rc.d/init.d/http start <küsib serveri võtme passwordi, mis on
"salakala">
```

Siin toodud serveri võtmed on väljastatud arvutile localhost.localdomain ja seega peaks seda olema võimalik testida suvalises arvutis. Arhiivi on lisatud ka uus ID kaardi pkcs11 ohjurprogrammi. Eelmises artiklis kirjeldatu ei sobi brauseris kasutamiseks. Siin toodud ohjurprogrammi on parem (brauseris kasutamiseks) ka selle poolest et ta kasutab vaid ID kaardi autentimissertifikaati ja seega ei hakka brauser teie allkirjastamissertifikaadi jaoks PIN-i küsima. Seega tuleb testimiseks kindlasti installeerida brauserisse siin toodud uus ID kaardi ohjurprogramm. Netscape 4 puhul valige Communicator -> Tools -> Security Info -> Cryptographic Modules. Kui loetelus juba on EstEID moodul, siis eemaldage ta "Delete" nupu abil. Nüüd lisame uue mooduli "Add" nupu abil. Dialoogis sestame nimeks näiteks EstIDAuth ja teegi asukohaks /usr/local/lib/libestid-pkcs11-authonly.so. Bruaser küsib nüüd PIN1 -e. RedHat 7.3 -ga kaasasolev Mozilla 0.9.9 on paraku vigane ja ei kõlba. Testitud on asja aga Mozilla 1.1 ja Netscape 6.2.2 -ga. Nende brauserit puhul tuleb

ohjurprogramm instaleerida Edit -> Preferences -> Privacy & Security -> Certificates
-> Manage Security Devices abil. Nüüd suuname brauseri esmalt URL-le
http://localhost/cgi-bin/ca_cert.cgi ja aktepteerime serveri sertifikaadi. Vajalik on seda
sertifikaati tunnustada vaid veebisaitide CA-na. Siis suuname brauseri URL-le
https://localhost/cgi-bin/ssl_env.cgi. Nüüd peaksite nägema brauseriaknas SSL
sessiooni ümbruskonnamuutujaid ja selahulgas ka oma sertifikaadi infot.

veiks26@hotmail.ee