Lab Journal (temporal file)

Group 16: David Rozas & Kunal Masse

Part I: Certificate Authority

19th February 2008

- Group certificate request generated and sent:
  - $ openssl req -in grnn_req.pem -verify -text -noout
- We decided to start with part II (see activities performed on 19th February in Part II)

20th February 2008

- Certificate properly signed received and checked with commands in the lab description.
- Creation of directories accomplished
- We copy gr16_private.key and we rename it to cakey.pem
- To create caconf.cnf, we copy and rename stud_openssl.cnf. Some changes will be applied over this configuration file
- Creation of configuration file caconf.cnf for the CA: based on http://www.eclectica.ca/howto/ssl-cert-howto.php && http://www.openssl.org/docs/apps/req.html
  - In the tutorial at the beginning they explained how to be create own root CA, but we have a signed one, so this is not necessary.
  - We have followed these steps:
    - create directories (file structure differs from tutorial, newcerts in our is inside private)
    - create a database for the certificates we will sign:

```
# echo '01' >serial
# touch index.txt
```

    - Create caconf.cnf, first step:

```
#---Begin---
#
# OpenSSL configuration file.
#
# drozas: Establish working directory.
dir             = .


[ req ]
```

```
default_bits                = 2048
#drozas: changed default keyfile from private.key to cakey.pem
default_keyfile             = cakey.pem
distinguished_name          = req_distinguished_name
string_mask                 = utf8only
#drozas: default_message_diggest necessary?
[ req_distinguished_name ]
countryName                 = Country Name (2 letter code)
countryName_default         = NO
countryName_min             = 2
countryName_max             = 2
#drozas change 0 to 1 because of the value on serial
1.organizationName          = Organization Name (eg, company)
1.organizationName_default  = NORGES TEKNISK-NATURVITENSKAPELIGE UNIVERSITET NTNU
organizationalUnitName      = Department
organizationalUnitName_default  = Telematics
#drozas: change comment, and default value added
commonName                  = Common Name (name of the server)
commonName_default          = ttm4135.item.ntnu.no
commonName_max              = 64

#drozas added to avoid put more parameters on the command line in certificate creation, we are going to force the use of server_cert
#distinguished_name     = req_distinguished_name ###duplicated!!
req_extensions      = server_cert

#drozas: necessary?. To protect of unauthorized use of our CA?
#[ v3_ca ]
#basicConstraints       = CA:TRUE
#subjectKeyIdentifier   = hash
#authorityKeyIdentifier = keyid:always,issuer:always

#Section with options to create certificates, probably not necessary because we are only going to certificate for apache
#[ v3_req ]
#basicConstraints       = CA:FALSE
#subjectKeyIdentifier   = hash

#Options to sign apache certificates
[ server_cert ]
```

basicConstraints = critical, CA:FALSE
subjectKeyIdentifier = hash
keyUsage = digitalSignature, keyEncipherment
nsCertType = server
#----End----

- Generation of the apache certificate request,
  - openssl req -new -nodes -keyout newcerts/server_key.pem -out newcerts/server_cert.pem -config caconf.cnf
  - The Common Name we have chosen is ttm4135.item.ntnu.no because this is the name of the host were we are going to run apache
- Checking that is correct:
  - openssl req -in newcerts/server_cert.pem -text -verify -noout

verify OK

Certificate Request:

Data:

Version: 0 (0x0)

Subject: C=NO, O=NORGES TEKNISK-NATURVITENSKAPELIGE UNIVERSITET NTNU, OU=Telematics, CN=ttm4135.item.ntnu.no

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

RSA Public Key: (2048 bit)

Modulus (2048 bit):

00:c7:ce:c1:77:88:d5:72:9a:6e:dc:42:b5:3d:88:

3a:62:61:8a:08:61:b4:90:98:e3:93:74:e7:c6:73:

a8:06:37:4d:e0:98:71:31:47:cf:ca:78:26:51:a7:

f0:e8:e1:81:4e:d5:d5:d8:70:f8:27:de:8f:61:40:

8f:89:e9:9a:28:0c:bb:aa:8a:1b:dc:c7:7e:94:29:

fb:b4:db:58:ad:7a:c7:c0:a6:8a:97:e4:68:81:a0:

8a:e9:33:04:4e:cd:5f:4f:b6:d9:bb:00:a4:60:29:

9c:8c:31:0e:cc:01:58:ae:02:5e:73:b7:bc:16:91:

f3:18:5b:28:ae:ae:47:7d:7f:ea:88:5c:4e:e7:5c:

c4:09:3f:45:6c:4b:4f:23:3c:1f:27:3f:4c:14:3b:

c3:26:21:55:4a:11:a2:81:05:47:63:4e:66:06:c5:

5c:bf:1f:7a:37:d0:44:51:05:d2:aa:d1:ae:d6:7e:

99:98:fa:de:e3:61:94:f1:6a:8e:44:bb:60:c0:d9:

3f:62:5a:99:e1:79:6f:03:26:0b:34:4b:68:4e:9a:

65:46:a5:6a:65:6d:04:72:ab:dd:63:8a:82:d0:6c:

6b:43:b0:b8:66:08:19:fd:b1:25:bc:8d:62:9f:12:

65:54:a3:2c:f6:70:79:b8:e7:2a:bb:82:ab:49:cb:

0d:07

Exponent: 65537 (0x10001)

Attributes:

a0:00

Signature Algorithm: sha1WithRSAEncryption

a5:42:09:9b:4b:6f:35:25:d0:33:90:73:7e:d0:56:d4:e8:6f:

60:c5:08:e2:91:71:4d:e9:75:a8:db:03:e4:0f:e9:68:b5:7d:

fd:f6:d8:74:3d:60:7a:60:d0:f8:23:a4:0b:3e:0d:b8:a9:aa:

3f:28:a4:fe:05:3e:ec:55:6b:58:29:14:f4:8e:36:f1:00:25:

56:28:61:56:45:10:53:12:56:77:96:ea:b5:4d:82:c3:5e:aa:

03:0c:9a:b6:d6:42:7d:d6:62:34:7c:7f:86:ef:0f:b9:79:23:

ae:bf:ea:db:6a:10:62:78:88:bc:a3:41:4c:f7:29:e1:14:90:

dc:df:c4:2b:7d:29:b0:ee:7d:8e:3f:1f:6a:dc:6e:1e:92:85:

9f:f7:2c:71:2e:57:81:6b:bd:d8:ea:31:1e:4b:e5:e6:a5:52:

e4:49:ff:1d:67:4a:7b:94:ee:20:86:1b:3a:e9:62:ab:bb:26:

0b:7d:fd:07:81:59:7c:40:dd:29:c4:1c:dc:94:11:e1:77:c1:

40:8b:f4:82:55:f1:a4:59:f9:ee:ea:4e:d8:22:be:9f:1d:32:

ef:a3:f7:da:f5:8b:ad:70:ee:2e:21:65:c0:bd:e7:ce:28:dc:

24:d1:c5:8d:7a:ab:e3:31:72:76:73:b8:19:56:95:32:a7:7b:

e1:96:ce:3b

- Adding information necessary to caconf.cnf in order to be able to sign:

#drozas: added to allow signing certificate

[ca]

default_ca = CA_default

[CA_default]

serial = $dir/serial

#drozas: change index.txt to index

database = $dir/index

new_certs_dir = $dir/newcerts

certificate = $dir/cacert.pem

#drozas: deleted private from path (we are going to execute from there)

private_key = $dir/cakey.pem

default_days = 365

default_md = md5

preserve = no

email_in_dn = no

nameopt = default_ca

certopt = default_ca

policy = policy_match

[policy_match]

countryName = match

organizationName = match

organizationalUnitName = optional

commonName = supplied

emailAddress = optional

- Signing certificate:
  - openssl ca -out newcerts/server_cert_signed.pem -config caconf.cnf -infiles newcerts/server_cert.pem
  - NOTE: We made a mistake about the names of the files which are not very

undestandable : server_cert.pem is in fact the request , and server_cert_signed.pem should be server_cert.pem (we will rename later).

- Checking server_cert_signed.pem:
  - openssl x509 -in newcerts/server_cert_signed.pem -noout -text -purpose | more

Certificate:

Data:

Version: 1 (0x0)

Serial Number: 1 (0x1)

Signature Algorithm: md5WithRSAEncryption

Issuer: C=NO, O=NORGES TEKNISK-NATURVITENSKAPELIGE UNIVERSITET NTNU, OU=Telematics, CN=Gr16 CA

Validity

Not Before: Feb 20 19:52:30 2008 GMT

Not After : Feb 19 19:52:30 2009 GMT

Subject: C=NO, O=NORGES TEKNISK-NATURVITENSKAPELIGE UNIVERSITET NTNU, OU=Telematics, CN=ttm4135.item.ntnu.no

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

RSA Public Key: (2048 bit)

Modulus (2048 bit):

00:c7:ce:c1:77:88:d5:72:9a:6e:dc:42:b5:3d:88:

3a:62:61:8a:08:61:b4:90:98:e3:93:74:e7:c6:73:

a8:06:37:4d:e0:98:71:31:47:cf:ca:78:26:51:a7:

f0:e8:e1:81:4e:d5:d5:d8:70:f8:27:de:8f:61:40:

8f:89:e9:9a:28:0c:bb:aa:8a:1b:dc:c7:7e:94:29:

fb:b4:db:58:ad:7a:c7:c0:a6:8a:97:e4:68:81:a0:

8a:e9:33:04:4e:cd:5f:4f:b6:d9:bb:00:a4:60:29:

9c:8c:31:0e:cc:01:58:ae:02:5e:73:b7:bc:16:91:

f3:18:5b:28:ae:ae:47:7d:7f:ea:88:5c:4e:e7:5c:

c4:09:3f:45:6c:4b:4f:23:3c:1f:27:3f:4c:14:3b:

c3:26:21:55:4a:11:a2:81:05:47:63:4e:66:06:c5:

5c:bf:1f:7a:37:d0:44:51:05:d2:aa:d1:ae:d6:7e:

99:98:fa:de:e3:61:94:f1:6a:8e:44:bb:60:c0:d9:

3f:62:5a:99:e1:79:6f:03:26:0b:34:4b:68:4e:9a:

65:46:a5:6a:65:6d:04:72:ab:dd:63:8a:82:d0:6c:

6b:43:b0:b8:66:08:19:fd:b1:25:bc:8d:62:9f:12:

65:54:a3:2c:f6:70:79:b8:e7:2a:bb:82:ab:49:cb:

0d:07

Exponent: 65537 (0x10001)

Signature Algorithm: md5WithRSAEncryption

9a:8c:b4:50:33:a8:b6:5b:b9:5b:ba:45:60:0a:38:37:6b:97:

02:eb:ef:c1:05:fe:5a:0e:45:c3:eb:3f:d7:0f:8f:68:c5:ad:

01:95:35:14:d1:98:18:dc:4f:e2:97:8a:93:c2:77:09:89:96:

12:2d:a0:c2:bb:00:cd:2f:f3:25:03:62:9c:c0:1f:0e:de:92:

7b:36:00:b2:c5:3b:f3:03:b9:dd:1b:33:e8:f0:de:7c:9f:95:

e6:a1:ca:b1:c2:2b:db:3a:17:b3:5a:92:ff:e9:2f:66:f0:79:

73:4b:a7:06:7a:83:74:7a:cd:4c:72:5a:d1:d4:b4:93:96:52:

43:18:bb:48:14:fe:00:0f:79:78:c0:ac:5f:09:c9:93:44:58:

c9:97:1a:87:b4:ed:35:bf:ab:89:40:27:41:fb:56:83:31:e1:

32:19:6a:3b:d1:ef:2e:2d:a6:43:ee:b7:90:33:92:3f:1a:a1:

ab:b4:5a:91:2d:4a:7f:df:7e:a4:a8:b5:b5:1b:2a:38:41:cf:

53:8d:ed:c3:91:89:74:30:ec:eb:fa:76:f5:1e:ff:32:02:d0:

a2:e3:f8:f7:b1:bf:88:1b:51:61:07:87:63:b1:c6:cd:02:b6:

d0:54:d7:a3:0a:04:7c:bd:ae:81:5a:ec:9a:95:6b:c6:ba:a3:

85:e4:dd:74

Certificate purposes:

SSL client : Yes

SSL client CA : No

SSL server : Yes

SSL server CA : No

Netscape SSL server : Yes

Netscape SSL server CA : No

S/MIME signing : Yes

S/MIME signing CA : No

S/MIME encryption : Yes

S/MIME encryption CA : No

CRL signing : Yes

CRL signing CA : No

Any Purpose : Yes

Any Purpose CA : Yes

OCSP helper : Yes

OCSP helper CA : No

- Renaming to more understandable names:
  - server_cert.pem = server_cert.req
  - server_cert_signed.pem = server_cert.pem
- We check again with the following commands, and everything seems be correct:
  - openssl x509 -in newcerts/server_cert.pem -noout -text -purpose | more
  - openssl req -in newcerts/server_cert.req -text -verify -noout
- private key and certificate of the server has been copied to apache path

Part II

19th February 2008
(the date is previous to the finalization of Part I because we were waiting to receive the group certificate signed)
[THIS PART WILL BE PROBABLY REPEATED FROM THE BEGINNING IN THE NEXT SESSION IN ORDER TO ANALYZE THE PROBLEMS MORE CAREFULLY]
What is the name of the person signing this Apache release? --> Jim Jagielski
[http://httpd.apache.org/dev/verification.html]
- get the signature

- get the public key of the releaser from a database with many public keys:

Attempt1:
gpg --keyserver pgpkeys.mit.edu --recv-key DE885DD3
gpg: requesting key DE885DD3 from hkp server pgpkeys.mit.edu
gpg: key DE885DD3: duplicated user ID detected - merged
gpg: key DE885DD3: public key "Sander Striker <striker@apache.org>" imported
gpg: no ultimately trusted keys found
gpg: Total number processed: 1
gpg: imported: 1

Why does this public key have a duplicated user?

There is a duplicated user...but anyway this was not the public key necessary (we have to download the public key of 08C975E5

Attempt 2:
gpg --keyserver pgpkeys.mit.edu --recv-key 08C975E5

gpg: requesting key 08C975E5 from hkp server pgpkeys.mit.edu
gpg: key 08C975E5: duplicated user ID detected - merged
gpg: key 08C975E5: public key "Jim Jagielski <jim@apache.org>" imported
gpg: no ultimately trusted keys found
gpg: Total number processed: 1
gpg: imported: 1

It is DUPLICATED AS WELL!

But, with this one we can verify the signature (with a public key we have also to verify)

gpg: requesting key 08C975E5 from hkp server pgpkeys.mit.edu
gpg: key 08C975E5: duplicated user ID detected - merged
gpg: key 08C975E5: public key "Jim Jagielski <jim@apache.org>" imported
gpg: no ultimately trusted keys found
gpg: Total number processed: 1
gpg: imported: 1

Ask about the process, this is what we understand:

About Apache:
We have the file and the signature of this file (specific), that has been created through the SIGNATURE

(unique for the user) for this file.

So we can compare that with the hash function case in this way:
source file = source file
process of signing the file with the USER SIGNATURE = apply the hash function to the source file
signature of the file (.asc) = to the fingerprint

Finally, the problem we have is to verify the public key we have downloaded in order to verify that the signature of the file is real, is to verify that this public key is really the public key.
We can achieve that trusting that the public key from the website is real.

24th February 2008

- TODO: Re-check fingerprints (ask in the next lab session)
- Our port will be: 8100 + 10 * (16-1) = 8250
- So our port range will be: 8250 8259
- Compiling and installing:
  - ./configure --prefix=/home/gr16/apache/ --enable-ssl enable-moduel=so
  - make
  - make install
  - The process seems to be accomplished successfully
- Configure httpd
  - Change Listen 80 to Listen 8250 in httpd.conf
  - Run the sever: bin/apachectl start
  - Verifying: http://ttm4135.item.ntnu.no:8250/
  - It works!
  - Changing htdocs/index.html message to Group 16 server for TTM4135 exercises
  - Verifying: http://ttm4135.item.ntnu.no:8250/

- First attempt shows the old message, probably because of the browser cache.
- After cleaning browser cache, the page showed is the one we have just modified. OK
- Ask about: recall the group's range of communication ports . We do not understand what do we have to do

Virtual hosts

- Based on: http://httpd.apache.org/docs/2.2/vhosts/name-based.html
- Created and added group.conf
- Test creating vhost with the same information than previous:
  - #To use name-based virtual hosting, you must designate the IP address
  - #(and possibly port) on the server that will be accepting requests for
  - #the hosts. This is configured using the NameVirtualHost directive.
  - NameVirtualHost *:8250
  - <VirtualHost *:8250>
  - ServerAdmin davidro@stud.ntnu.no
  - DocumentRoot /home/gr16/apache/htdocs
  - ServerName ttm4135.item.ntnu.no
  - <Directory /home/gr16/apache/htdocs>
  - Options +Indexes
  - Allow from all
  - </Directory>
  - </VirtualHost>
- Question. Who are the options finally taken (because we have the same configuration)

- Example: We are going to create another directory, and change in the vHost configuration:
  - DocumentRoot /home/gr16/apache/vhost1
  - ServerName ttm4135.item.ntnu.no
  - <Directory /home/gr16/apache/vhost1>
- It took the changes in VirtualHost directive...so it seems in case of duplicated info, it takes the last one
- Adding vhost 2:
  - Adding listening in another port: 8251 in httpd.conf
  - Adding Virtual host directives in group.conf

Configure ssl

- vhost1 will be http server, and vhost2 will be htpps server (we already did this part)
- So now, we have two different servers running over the same machine executing one only instance of apache (this is more elegant than execute two instances of apache)
- Reading: http://httpd.apache.org/docs/2.2/ssl/ and http://www.modssl.org/docs/2.8/ssl_overview.html
- Adding paths to certificates in group.cfn (in vhost2 section):
  - SSLCertificateFile = /home/gr16/ca/private/newcerts/server_cert.pem
  - SSLCertificateKeyFile = /home/gr16/ca/private/newcerts/sever_key.pem
- Creating Certificate Chain: /home/gr16/ca/private/newcerts/cert_chain.pem by concatenating all the certificates from group to root
  - SSLCertificateChainFile = /home/gr16/ca/private/newcerts/cert_chain.pem
- But we are not still authenticating

Client Authentication

- Creating new file system (htdocs/secure, htdocs/secure/restristect), deleting the one we use for our tests (vhost1,vhost2) and making necessary changes in group.cnf
- Creating html test files, restarting apache and cheking...Changes work properly
- Reading: http://httpd.apache.org/docs/2.2/ssl/ssl_howto.html and http://httpd.apache.org/docs/2.2/mod/mod_ssl.html#sslrequire
- We should use SSLRequire directives

Configuring /gr16/apache/htdocs/secure

- For .../secure, it would be enough adding:
- SSLRequireSSL Directive

Description: Deny access when SSL is not used for the HTTP request
Syntax: `SSLRequireSSL`
Context: directory, .htaccess
Override: AuthConfig
Status: Extension
Module: mod_ssl

This directive forbids access unless HTTP over SSL (i.e. HTTPS) is enabled for the current connection. This is very handy inside the SSL-enabled virtual host or directories for defending against configuration errors that expose stuff that should be protected. When this directive is present all requests are denied which are not using SSL.

Example

```
SSLRequireSSL
```

- We have also to start the SSL service adding SSLEngine on in the Virtual host Context
- So, for /secure, the following changes are applied:
- ...
- #Enable/Disable SSL for this virtual host.

- SSLEngine on
-
- <Directory /home/gr16/apache/htdocs/secure>
- SSLRequireSSL
- Options +Indexes
- Allow from all
- </Directory>
- It seems to work properly: if we try to connect with http returns and error, and when we try to connect with https we have to accept the certificate and show the index.html of vhost2 properly

Configuring /gr16/apache/htdocs/secure/admin

- In this case, we will have to use SSLRequire directives. We have started with this one because the regular expression to use seems to be easier to find.
- Reading again **http://httpd.apache.org/docs/2.2/mod/mod_ssl.html#sslrequire**
- Reading again httpd-ssl.conf
- To get access to SSL environment variables, we add SSLOptions +StdEnvVars
- If we try to connect, returns error -12227...it should work because I have certificate signed by NTNU CA (the one that I used for register in the group).
- WHY?. Ask in the lab
- Anyway, we are going to think in the condition for /require, and we will ask to the teachers in the lab the next day
- WHAT IS THE PROBLEM?
  - Error in the conditions
  - Error in how to use certificates to test it

- Current group.cnf version:
- #drozas- adding parameters about certificates
- #Path to server certificate
- SSLCertificateFile /home/gr16/ca/private/newcerts/server_cert.pem
- #Path to server private key
- SSLCertificateKeyFile /home/gr16/ca/private/newcerts/server_key.pem
- #Path to server certificate chain
- SSLCertificateChainFile /home/gr16/ca/private/newcerts/cert_chain.pem
-
-
- #To use name-based virtual hosting, you must designate the IP address
- #(and possibly port) on the server that will be accepting requests for
- #the hosts. This is configured using the NameVirtualHost directive.
-
- #Virtual host 1 - HTTP Server
- NameVirtualHost *:8250
- <VirtualHost *:8250>
- ServerAdmin davidro@stud.ntnu.no
- DocumentRoot /home/gr16/apache/htdocs
- ServerName ttm4135.item.ntnu.no
- <Directory /home/gr16/apache/htdocs>
- Options +Indexes

- Allow from all
- </Directory>
- </VirtualHost>
-
-
- #Virtual host 2 - HTTPS Server
- NameVirtualHost *:8251
- <VirtualHost *:8251>
- ServerAdmin davidro@stud.ntnu.no
- DocumentRoot /home/gr16/apache/htdocs/secure
- ServerName ttm4135.item.ntnu.no
-
- #drozas: Enable/Disable SSL for this virtual host.
- SSLEngine on
- SSLVerifyClient none
-
- #This directory is only accessible using HTTPS, but everybody can access
- <Directory /home/gr16/apache/htdocs/secure>
- SSLRequireSSL
- Options +Indexes
- Allow from all
- </Directory>

```
-
- #This directory is only accesible by all clients with an NTNU-certificate
- <Directory /home/gr16/apache/htdocs/secure/admin>
- SSLVerifyClient require
-
- #drozas: get access to ssl enviroment vars
- SSLOptions +StdEnvVars
- SSLCACertificateFile /home/gr16/ca/private/newcerts/server_cert.pem
-
- SSLRequireSSL
- SSLRequire %{SSL_CLIENT_S_DN_O} eq %{SSL_SERVER_S_DN_O} \
- and %{SSL_CLIENT_I_DN_O} eq %{SSL_SERVER_S_DN_O} \
-
- Options +Indexes
- Allow from all
- </Directory>
-
- <Directory /home/gr16/apache/htdocs/secure/restricted>
- SSLVerifyClient require
- #At least,
- SSLVerifyDepth 3
- #drozas: get access to ssl enviroment vars
```

- SSLOptions +StdEnvVars
- SSLCACertificateFile /home/gr16/ca/private/newcerts/server_cert.pem
-
- SSLRequireSSL
- SSLRequire %{SSL_CLIENT_S_DN_O} eq %{SSL_SERVER_S_DN_O} \
- and %{SSL_CLIENT_I_DN_O} eq %{SSL_SERVER_S_DN_O} \
- and %{SSL_CLIENT_S_DN_OU} eq %{SSL_SERVER_S_DN_OU} \
- and %{SSL_CLIENT_S_DN_CN} in {%{SSL_SERVER_S_DN_CN}, "tm4135.item.ntnu.no"}
-
-
- Options +Indexes
- Allow from all
- </Directory>
-
-
- </VirtualHost>

26th February 2008

- Correcting problems in the group.conf:
  - The SSLCACertificateFile has to be the ntnu: /home/gr16/ca/private/newcerts/ntnuca.pem
  - We have to take into account the SSLVerifyDepth:
  - This directive sets how deeply mod_ssl should verify before deciding that the clients don't have

a valid certificate. Notice that this directive can be used both in per-server and per-directory context. In per-server context it applies to the client authentication process used in the standard SSL handshake when a connection is established. In per-directory context it forces a SSL renegotation with the reconfigured client verification depth after the HTTP request was read but before the HTTP response is sent.

The depth actually is the maximum number of intermediate certificate issuers, i.e. the number of CA certificates which are max allowed to be followed while verifying the client certificate. A depth of 0 means that self-signed client certificates are accepted only, the default depth of 1 means the client certificate can be self-signed or has to be signed by a CA which is directly known to the server (i.e. the CA's certificate is under `SSLCACertificatePath`), etc.

- So it will be 2 for /restricted and 3 for /admin
- We have also to check the conditions:
  - For admin:
    - #drozas: condition is that the client must have a certificate of NTNU
    - SSLRequire %{SSL_CLIENT_S_DN_O} eq %{SSL_SERVER_S_DN_O} \
    - and %{SSL_CLIENT_I_DN_O} eq %{SSL_SERVER_S_DN_O}
  - For restricted
    - #drozas: condition is that the client must have a certificate from ntnu,
    - #belonging to the telematics department and being part of the staff
    - #(has been issued by staff ca) or being one of the members (our cn is unique, because is our e-mail addresses)
    - SSLRequire %{SSL_CLIENT_S_DN_O} eq %{SSL_SERVER_S_DN_O} \
    - and %{SSL_CLIENT_I_DN_O} eq %{SSL_SERVER_S_DN_O} \
    - and %{SSL_CLIENT_S_DN_OU} eq %{SSL_SERVER_S_DN_OU} \
    - and (%{SSL_CLIENT_I_DN_CN} eq "Staff CA" or (%{SSL_CLIENT_S_DN_CN} in {"davidro@stud.ntnu.no", "masse@stud.ntnu.no"}))
- Now it seems to work.

- The following tests have been made:
  - Test to /secure
    - If we try to make a not safe connection, it returns a error: OK
    - If we try to make a safe connection, it works: OK
  - Test to secure/admin
    - If we try to connect without NTNU certificate, it returns an error: OK
    - If we try to connect with an NTNU certificate, it works: OK
  - Test to secure/restricted
    - If we try to connect without certificate, it returns an error: OK
    - If we try to connect with an student certificate with another e-mail (we make that by changing temporarily the condition), it returns an error: OK
    - If we try to connect with davidro@stud.ntnu.no, it works: OK
    - TODO: ask the teachers if they can access with a staff certificates
- If teachers can access, we can conclude that the part 2 is finished
- With these changes, the final state of group.conf is:

#drozas- adding parameters about certificates

#Path to server certificate

SSLCertificateFile /home/gr16/ca/private/newcerts/server_cert.pem

#Path to server private key

SSLCertificateKeyFile /home/gr16/ca/private/newcerts/server_key.pem

#Path to server certificate chain

SSLCertificateChainFile /home/gr16/ca/private/newcerts/cert_chain.pem

```
#To use name-based virtual hosting, you must designate the IP address

#(and possibly port) on the server that will be accepting requests for

#the hosts. This is configured using the NameVirtualHost directive.


#Virtual host 1 - HTTP Server

NameVirtualHost *:8250

<VirtualHost *:8250>

ServerAdmin davidro@stud.ntnu.no

DocumentRoot /home/gr16/apache/htdocs

ServerName ttm4135.item.ntnu.no

<Directory /home/gr16/apache/htdocs>

Options +Indexes

Allow from all

</Directory>

</VirtualHost>


#Virtual host 2 - HTTPS Server

NameVirtualHost *:8251

<VirtualHost *:8251>

ServerAdmin davidro@stud.ntnu.no

DocumentRoot /home/gr16/apache/htdocs/secure
```

```
ServerName ttm4135.item.ntnu.no

#drozas: Enable/Disable SSL for this virtual host.

SSLEngine on

SSLVerifyClient none


#This directory is only accessible using HTTPS, but everybody can access

<Directory /home/gr16/apache/htdocs/secure>

SSLRequireSSL

Options +Indexes

Allow from all

</Directory>


#This directory is only accesible by all clients with an NTNU-certificate

<Directory /home/gr16/apache/htdocs/secure/admin>

SSLVerifyClient require


#drozas: get access to ssl enviroment vars

SSLOptions +StdEnvVars

#this should be ntnu

SSLCACertificateFile /home/gr16/ca/private/newcerts/ntnuca.pem

SSLVerifyDepth 3

SSLRequireSSL
```

#drozas: condition is that the client must have a certificate of NTNU

SSLRequire %{SSL_CLIENT_S_DN_O} eq %{SSL_SERVER_S_DN_O} \

and %{SSL_CLIENT_I_DN_O} eq %{SSL_SERVER_S_DN_O}

Options +Indexes

Allow from all

</Directory>

<Directory /home/gr16/apache/htdocs/secure/restricted>

SSLVerifyClient require

SSLVerifyDepth 2

#drozas: get access to ssl enviroment vars

SSLOptions +StdEnvVars

SSLCACertificateFile /home/gr16/ca/private/newcerts/ntnuca.pem

SSLRequireSSL

#drozas: condition is that the client must have a certificate from ntnu,

#belonging to the telematics department and being part of the staff

#(has been issued by staff ca) or being one of the members (our cn is unique, because is our e-mail addresses)

SSLRequire %{SSL_CLIENT_S_DN_O} eq %{SSL_SERVER_S_DN_O} \

and %{SSL_CLIENT_I_DN_O} eq %{SSL_SERVER_S_DN_O} \

and %{SSL_CLIENT_S_DN_OU} eq %{SSL_SERVER_S_DN_OU} \

and (%{SSL_CLIENT_I_DN_CN} eq "Staff CA" or (%{SSL_CLIENT_S_DN_CN} in {"davidro@stud.ntnu.no", "masse@stud.ntnu.no"}))

Options +Indexes

Allow from all

</Directory>

</VirtualHost>

- Error reported by teachers: the certificate chain is not valid
- We made a mistake, we copy the server certificate in the chain instead of the group 16 certificate.
  - Solved, pending of teacher's answer

Part III Writing a PHP application

4th March 2008

4.1-2 Install PHP and test

- Install commands:

  ./configure --with-apxs2=/home/gr16/apache/bin/apxs --with-config-file-path=/home/gr16/apache/conf/php.ini --disable-cgi --disable-cli --prefix=/home/gr16/php with-mysqli=/usr/bin/mysql_config

  make

  make install

- Output: looks right

- Copying configuration file:

  cp php.ini-dist ../../../apache/conf/php.ini

- Configuring apache to execute files ending with php

  AddType application/x-httpd-php .php

- Test works properly (it has been moved temporarily to /restricted, because only we and the staff have access): https://ttm4135.item.ntnu.no:8251/restricted/test.php

  4.3Mysql database

- First version: username = PK, password in clear, but...
  - Future improvements:
    - Password not clear (ex.:, using MD5)
    - Index as a primary key ?
- Testing DB, and testing some commands:
  - SHOW DATABASES;

- USE grdb16;
- SHOW TABLES; (empty)

- Script to create a simple test db ~/scripts_sql/create_test.sql:

  --drozas: create a very simple table

  CREATE TABLE IF NOT EXISTS users_test(

  user CHAR(15) PRIMARY KEY,

  pass CHAR(15) NOT NULL);

- Script with insert for testing: mysql> source /home/gr16/sql_scripts/insert_test.sql
- Checking: SELECT * FROM users_test;

- mysql> SELECT * FROM users_test;
- +------+--------+
- | user | pass |
- +------+--------+
- | u01 | pass01 |
- | u02 | pass02 |
- +------+--------+
- 2 rows in set (0.01 sec)

4.4 Constructing the application

6th March 2008

- TODO: Explain a little bit how we have made the application
- TESTS:
  - signup.php
    - Try to get access with a student certificate which not belongs to our group: not allowed (ok)
    - Duplicated entries, try to delete a value which does not exists, insert only name or only password, insert empty strings,...: not allowed (ok)
    - Inserting encryption in password: ok
  - getpdf.php
    - Try to access without session: not allowed (ok)
  - login.php
    - Cookies and sessions: ok
    - No user, no pass, ...: not allowed(ok)
    - Tests deleting cookies one by one and all at the time: ok
    - Testing encryption in passwords: ok
  - logout.php
    - Redirection: ok
    - Not possible to access to pdf once we logout trying to go directly to getpdf.php: ok
    - After login out it is still possible to create a new session by going back through the browser (but the user has already validated, so it is not supposed a possible attack, it is just an interface side effect)
  - TODO

- Research about possible SQL injection attacks, Cross-site attacks, etc.