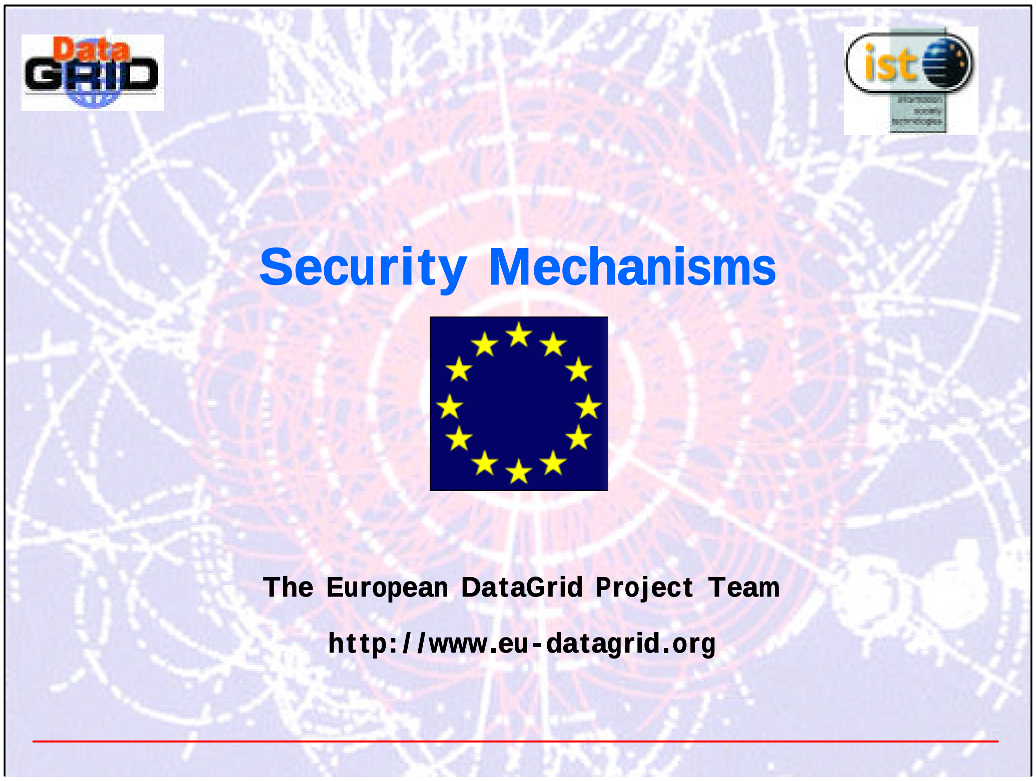
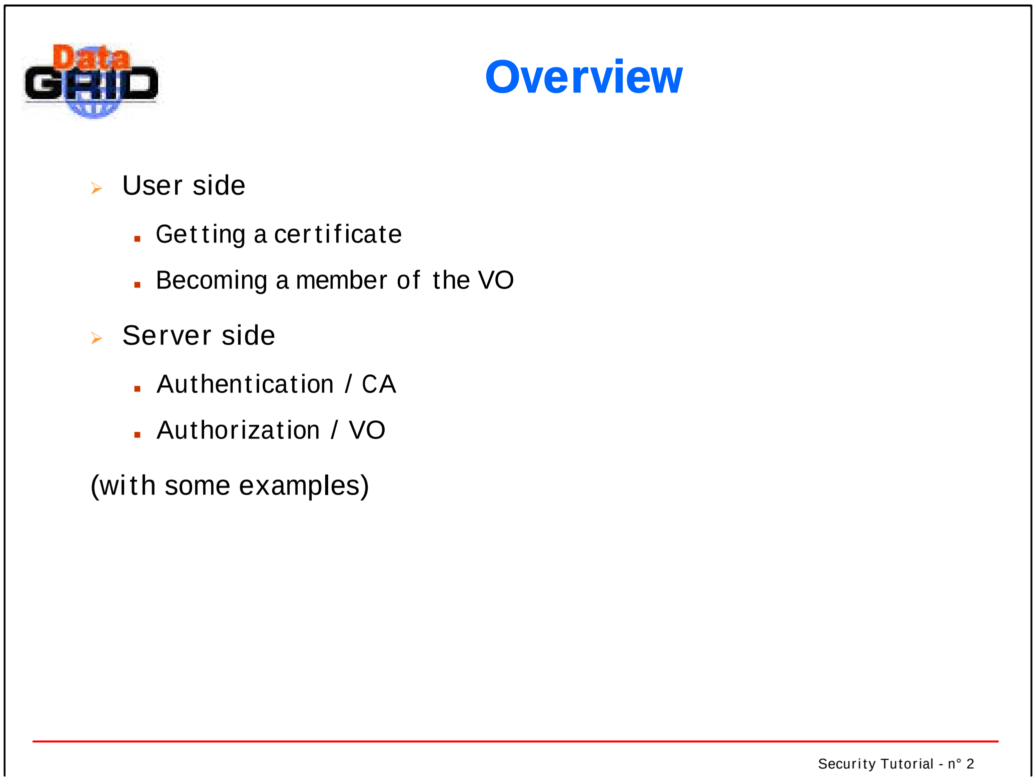


Security Mechanisms

The European DataGrid Project Team

http://www.eu-datagrid.org



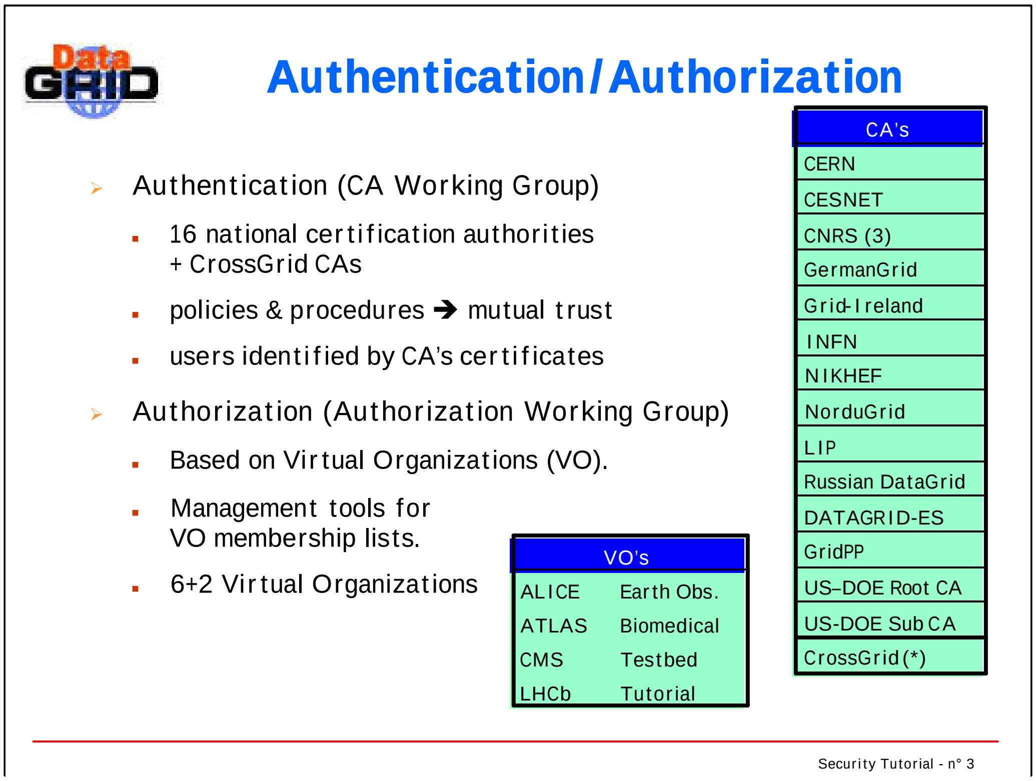
Overview

- User side
  - Getting a certificate
  - Becoming a member of the VO
- Server side
  - Authentication / CA
  - Authorization / VO

(with some examples)

Security Tutorial - n° 2

In this tutorial you will learn how the EDG Security mechanisms are being used.



Authentication/Authorization

- Authentication (CA Working Group)
  - 16 national certification authorities + CrossGrid CAs
  - policies & procedures ➔ mutual trust
  - users identified by CA's certificates
- Authorization (Authorization Working Group)
  - Based on Virtual Organizations (VO).
  - Management tools for VO membership lists.
  - 6+2 Virtual Organizations

CA's
CERN
CESNET
CNRS (3)
GermanGrid
Grid-Ireland
INFN
NIKHEF
NorduGrid
LIP
Russian DataGrid
DATAGRID-ES
GridPP
US–DOE Root CA
US-DOE Sub CA
CrossGrid (*)

VO's	
ALICE	Earth Obs.
ATLAS	Biomedical
CMS	Testbed
LHCb	Tutorial

The project supports CAs for members of the project and for related projects.

For a machine to participate as a Testbed resource **all** the CAs must be enabled.

•All CA certificates can be installed without compromising local site security.

•Site managers has full control over local resources.

CAs: there are 3 CAs at CNRS:

- CNRS
- CNRS – Projects
- CNRS – Datagrid-fr (catchall, which is for the ones, who have no CA)

The six main VOs are for WP8 (LHC experiments), WP9 (Earth Observation) and WP10 (Biomedical)

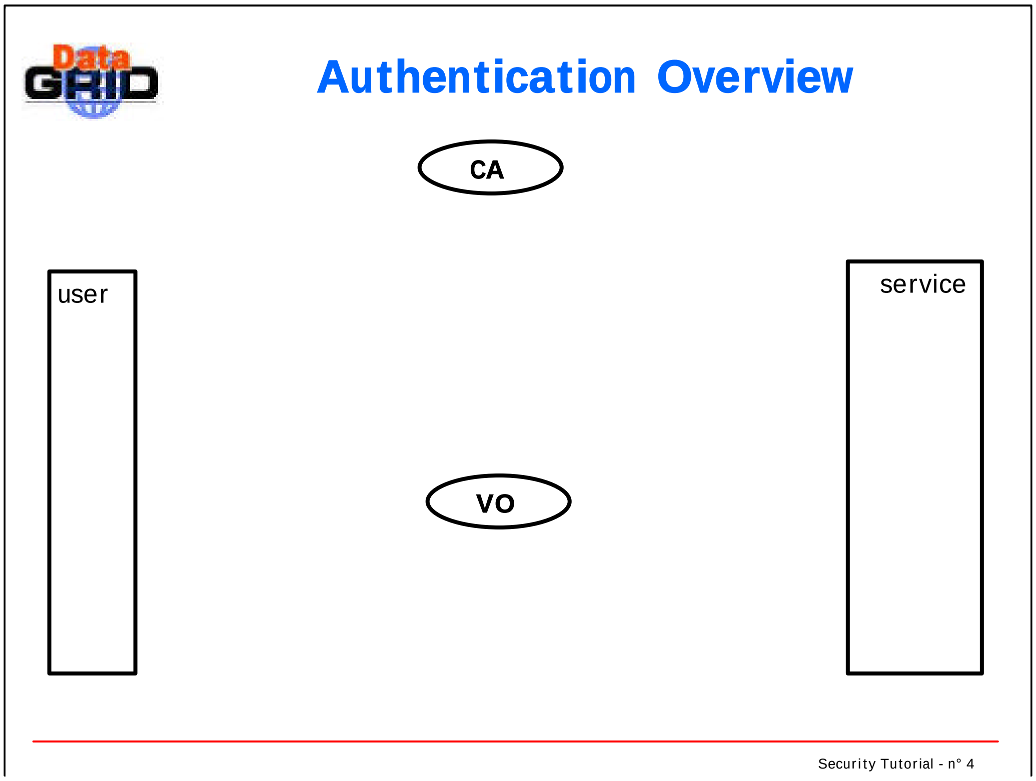
The Testbed VO is for testing, development and demo purposes.

The Tutorial VO is for this audience sitting in the room ☺

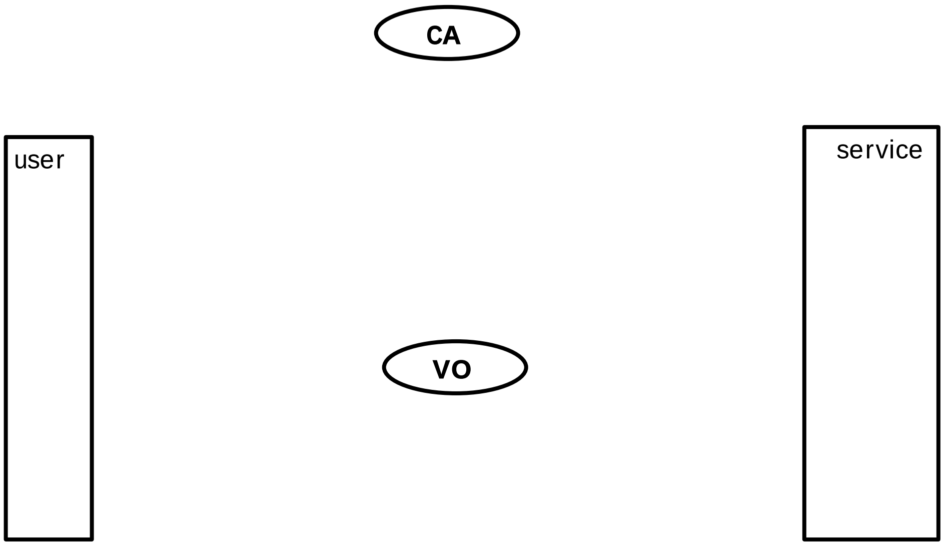
Real life example: traveling abroad

CA = gov. authority in your country issuing passport

VO = gov. authority in the destination country issuing visa

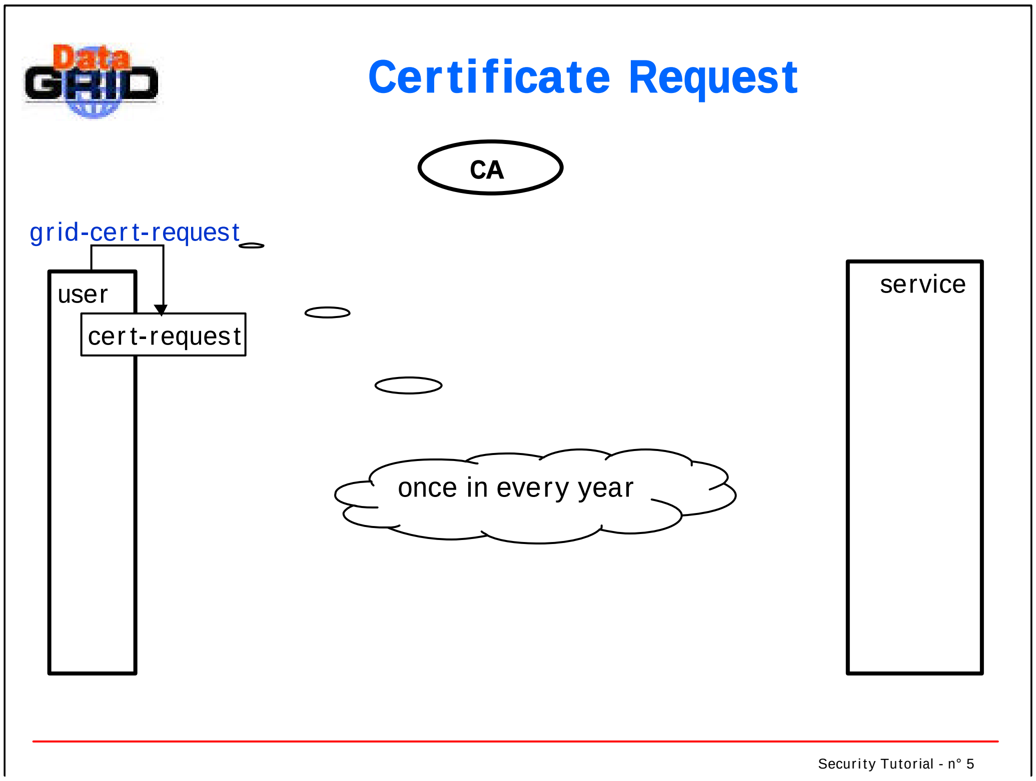


Authentication Overview



The authentication steps shown in the following slides are:

- user requests a certificate
- user sends the requests to the CA, which sends back the signed certificate
- conversion to browser digestable PKCS12 format
- registration in the EDG LDAP-VO
- generats a proxy certificate (24 hours lifetime)
- host/service requests a certificate
- host/service sends the requests to the CA, which sends back the signed certificate
- retrival of the trusted CA certificates and their CRLs
- generating a gridmap file from the LDAP database for authorization and mapping
- user contacts a service: they exchange their certificates to authenticate each other; the service bases its authorization decision on the gridmap file (... currently)



Certificate Request

Security Tutorial - n° 5

1. user requests a certificate

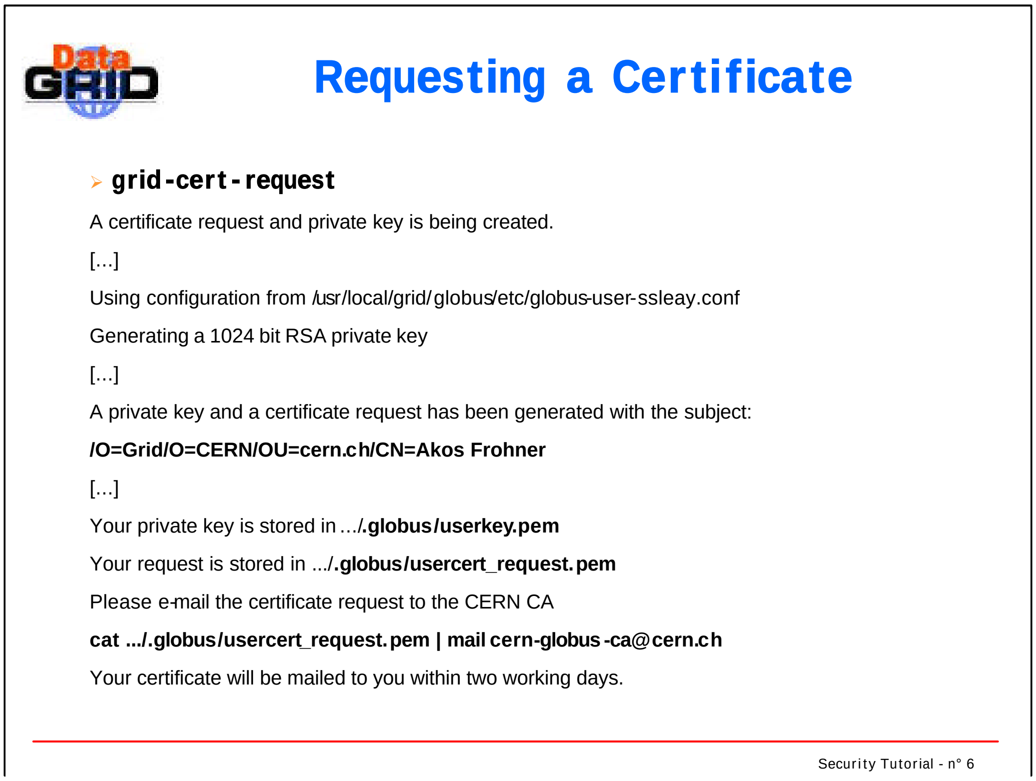
A certificate is a set of information (e.g. user’s name) + identification key (public key) + signature from the CA.

In a certificate request there is the set of informaiton + identification key.

Real life example:

A passport is a set of information (e.g. traveller’s name) + identification key (picture) + unique identifiers of the authority (passport material, stamps, etc.).

In a passport request (filled form) there is a set of information + picture.



Requesting a Certificate

➢ **grid-cert-request**

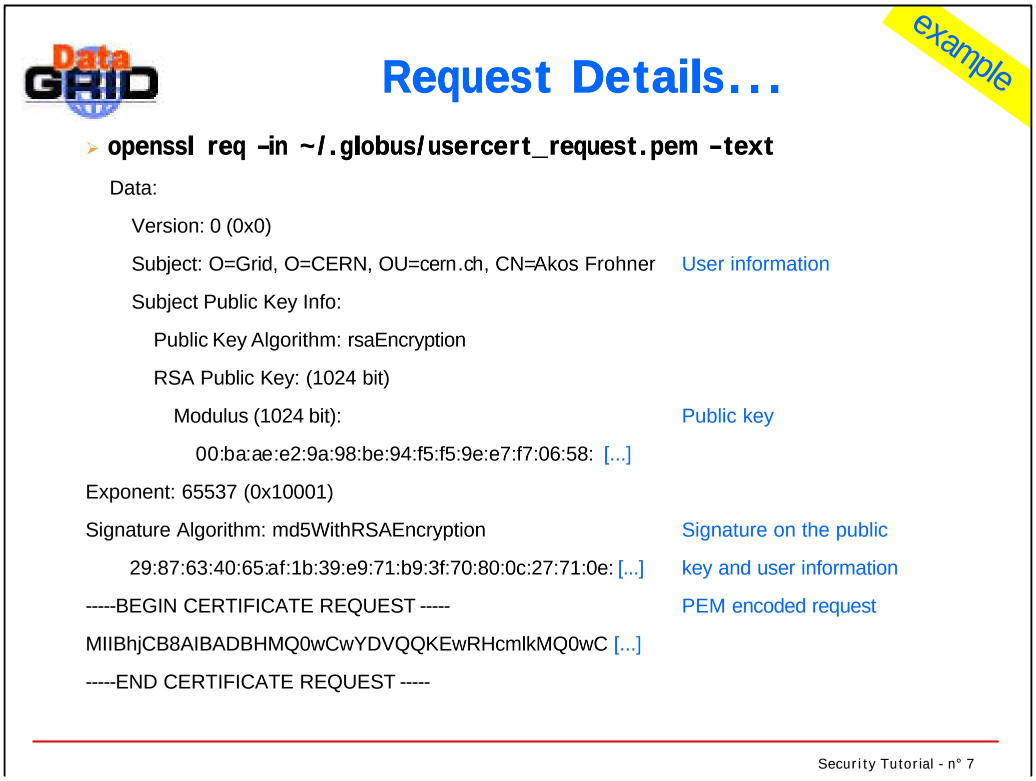
```
A certificate request and private key is being created.
[...]
Using configuration from /usr/local/grid/globus/etc/globus-user-ssleay.conf
Generating a 1024 bit RSA private key
[...]
A private key and a certificate request has been generated with the subject:
/O=Grid/O=CERN/OU=cern.ch/CN=Akos Frohner
[...]
Your private key is stored in .../.globus/userkey.pem
Your request is stored in .../.globus/usercert_request.pem
Please e-mail the certificate request to the CERN CA
cat .../.globus/usercert_request.pem | mail cern-globus-ca@cern.ch
Your certificate will be mailed to you within two working days.
```

Security Tutorial - n° 6

The subject or distinguished name is configured by the site administrator:

/O=Grid	grid/virtual organisations
/O=CERN	organisation CERN
/OU=cern.ch	organisation unit, the Internet domain name
/CN=Akos Frohner	the user's name from the site's user database (e.g. /etc/passwd)

The user has to provide a pass phrase, which will be used to encrypt his/her private key.



Request Details...

example

- **openssl req -in ~/.globus/usercert_request.pem -text**

```
  Data:
    Version: 0 (0x0)
    Subject: O=Grid, O=CERN, OU=cern.ch, CN=Akos Frohner     User information
    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
      RSA Public Key: (1024 bit)
        Modulus (1024 bit):                                  Public key
          00:ba:ae:e2:9a:98:be:94:f5:f5:9e:e7:f7:06:58: [...]
Exponent: 65537 (0x10001)
Signature Algorithm: md5WithRSAEncryption                    Signature on the public
    29:87:63:40:65:af:1b:39:e9:71:b9:3f:70:80:0c:27:71:0e: [...]   key and user information
-----BEGIN CERTIFICATE REQUEST-----                          PEM encoded request
MIIBhjCB8AIBADBHMQ0wCwYDVQQKEwRHcmlkMQ0wC [...]
-----END CERTIFICATE REQUEST-----
```

The certificate request has two important fields:

- user information (Info)
- and public key (Kpub)

These fields are signed by the private key (Kpriv) and this signature is placed in the request

signature = encrypt(Kpriv, hash(Info, Kpub))

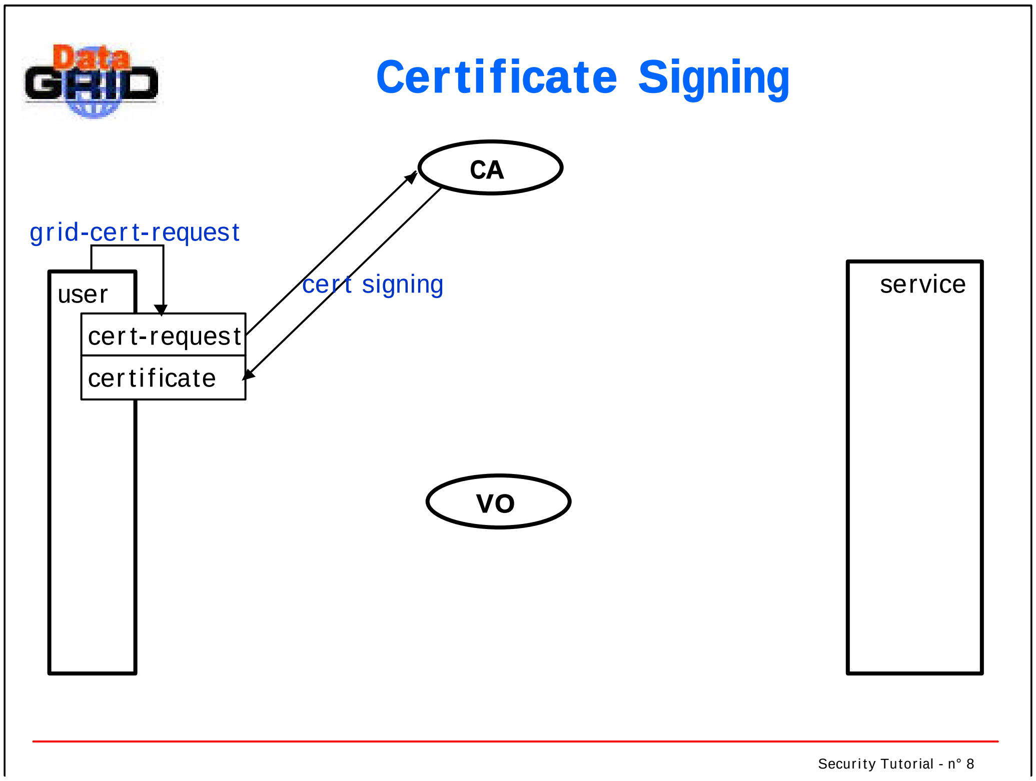
The Certificate Authority will check the validity of the request by creating the same hash and comparing it with the decrypted signature:

decrypt(Kpub, signature) = hash(Info, Kpub)

The first part of the output is the textual representation of the certificate request. This textual representation is only for informational purposes. It will not be processed by any program.

The second part is the PEM encoded version of the request. PEM encoding is necessary to send binary data over ASCII channels, like emails.

The certificate request itself is in ASN.1 format, which is a platform independent binary representation of arbitary data structures. It is basically the binary counterpart of XML (there are even tools to convert between ASN.1 and XML).



Certificate Signing

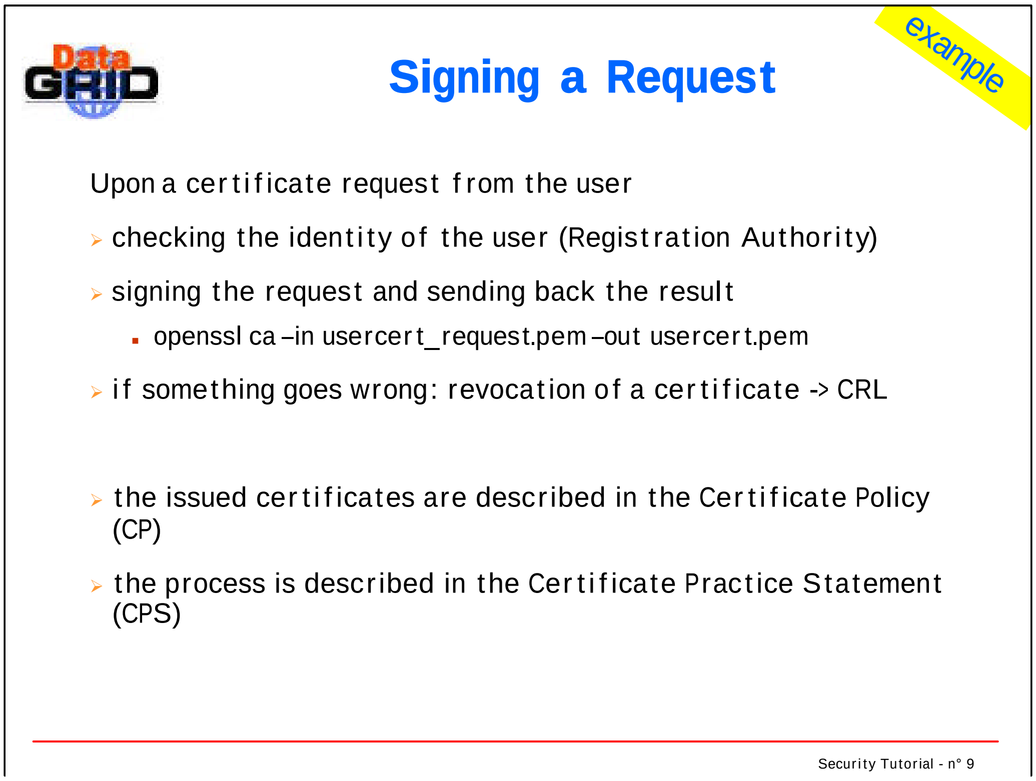
1. user requests a certificate
2. user sends the requests to the CA, which sends back the signed certificate

The CA calculates the hash of the certificate request and encodes it with its private key = signature.

The CA identifies the user by a registration authority (RA).

Real life example:

The authority identifies the traveller and if there is no problem, then issues a passport.



example

Signing a Request

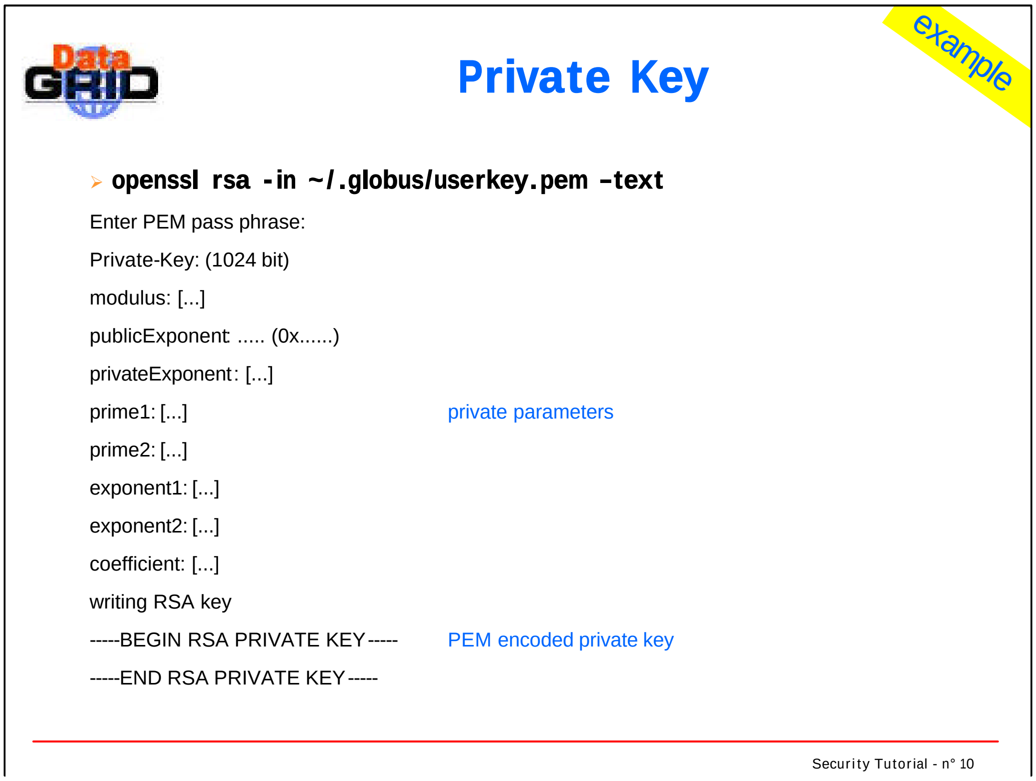
Upon a certificate request from the user

- checking the identity of the user (Registration Authority)
- signing the request and sending back the result
  - openssl ca –in usercert_request.pem –out usercert.pem
- if something goes wrong: revocation of a certificate -> CRL

- the issued certificates are described in the Certificate Policy (CP)
- the process is described in the Certificate Practice Statement (CPS)

The „openssl ca ...” line is a over-simplified version of the real command, however it usually differs only in the configuration parameters.

The distribution of the revocation list is still under discussion. We will need a reliable messaging protocol to the subscribed services from every CA in a VO.



example

Private Key

- **openssl rsa -in ~/.globus/userkey.pem -text**

```
Enter PEM pass phrase:
Private-Key: (1024 bit)
modulus: [...]
publicExponent ..... (0x......)
privateExponent: [...]
prime1: [...]                          private parameters
prime2: [...]
exponent1: [...]
exponent2: [...]
coefficient: [...]
writing RSA key
-----BEGIN RSA PRIVATE KEY-----        PEM encoded private key
-----END RSA PRIVATE KEY-----
```

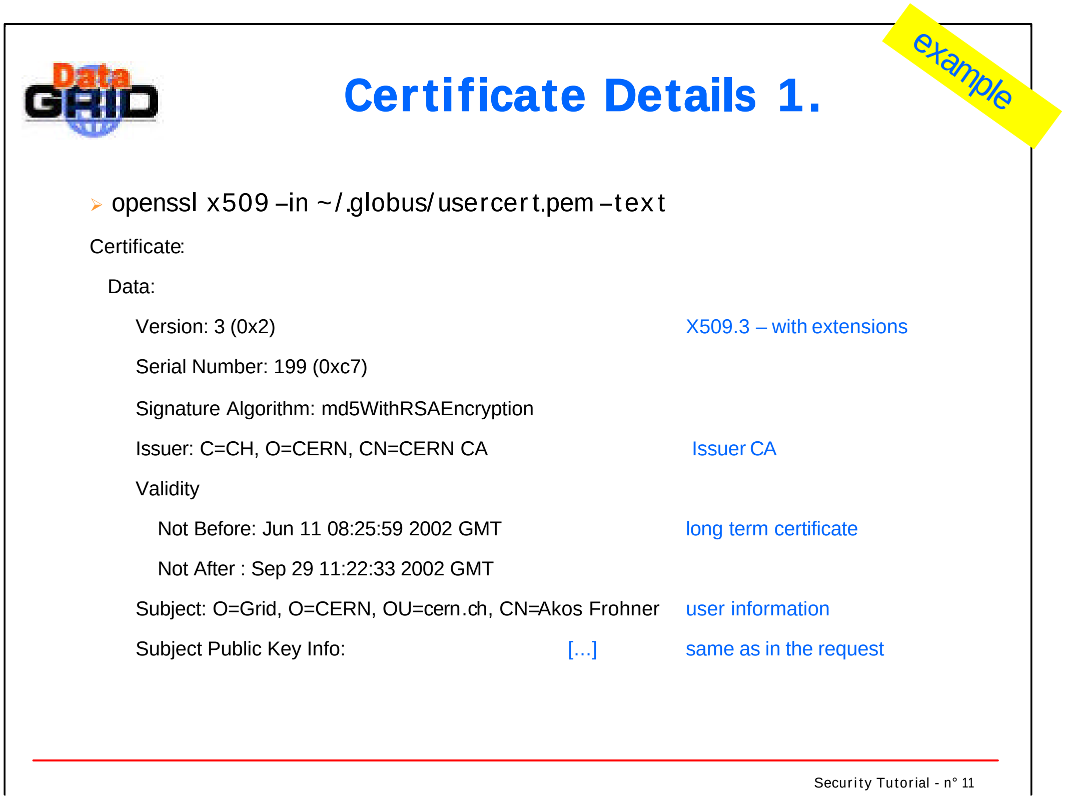
Security Tutorial - n° 10

One may print the private key as well. The actual values are omitted here, since they are private.

Actually it is a **bad** idea to print this key, since it would be visible on the screen and could be stored in some screen buffer...

The private key also stored in an ASN.1 format, but it is encrypted with a symmetric cypher algorithm (like DES or AES) on the disk. Once the user wants to use it, it has to be decrypted.

The key for the symmetric en/decryption is called the „pass phrase”. The user shall handle this phrase as a password, although it will never be sent over the network, even in encrypted format.



Certificate Details 1.

example

- openssl x509 –in ~/.globus/usercert.pem –text

```
Certificate:
  Data:
     Version: 3 (0x2)                                          X509.3 – with extensions
     Serial Number: 199 (0xc7)
     Signature Algorithm: md5WithRSAEncryption
     Issuer: C=CH, O=CERN, CN=CERN CA                          Issuer CA
     Validity
        Not Before: Jun 11 08:25:59 2002 GMT                   long term certificate
        Not After : Sep 29 11:22:33 2002 GMT
     Subject: O=Grid, O=CERN, OU=cern.ch, CN=Akos Frohner       user information
     Subject Public Key Info:                       [...]       same as in the request
```

Security Tutorial - n° 11

When the CA issues a certificate, it is basically giving a signature on some data, proving that this package really descibes one person.

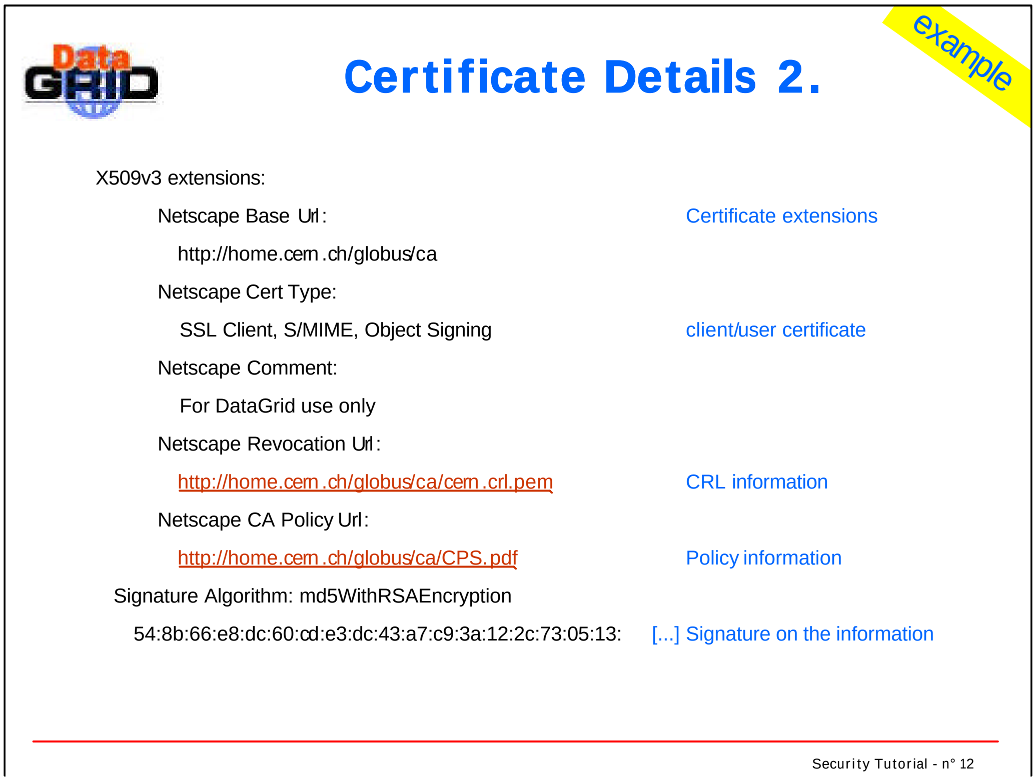
- user information (DN – same as in the reqest)
- user's public key
- validity period of the information
- various extended informations

The CA takes the data fields, calculates the hash and encrypts it with its private key:

signature = encrypt(CA-Kpriv, hash(Info, Kpub, ...))

A third party (e.g. service) can check the validity of a certificate (ie the user information and the public key belongs together) by calculating the hash of the informational fields and decrypting the signature:

hash(Info, Kpub, ...) = decrypt(CA-Kpub, signature)



Certificate Details 2.

example

```
X509v3 extensions:
        Netscape Base Url:
          http://home.cern.ch/globus/ca                         Certificate extensions
        Netscape Cert Type:
          SSL Client, S/MIME, Object Signing                    client/user certificate
        Netscape Comment:
          For DataGrid use only
        Netscape Revocation Url:
          http://home.cern.ch/globus/ca/cern.crl.pem            CRL information
        Netscape CA Policy Url:
          http://home.cern.ch/globus/ca/CPS.pdf                 Policy information
   Signature Algorithm: md5WithRSAEncryption
      54:8b:66:e8:dc:60:cd:e3:dc:43:a7:c9:3a:12:2c:73:05:13:  [...] Signature on the information
```

Extensions in the certificate:

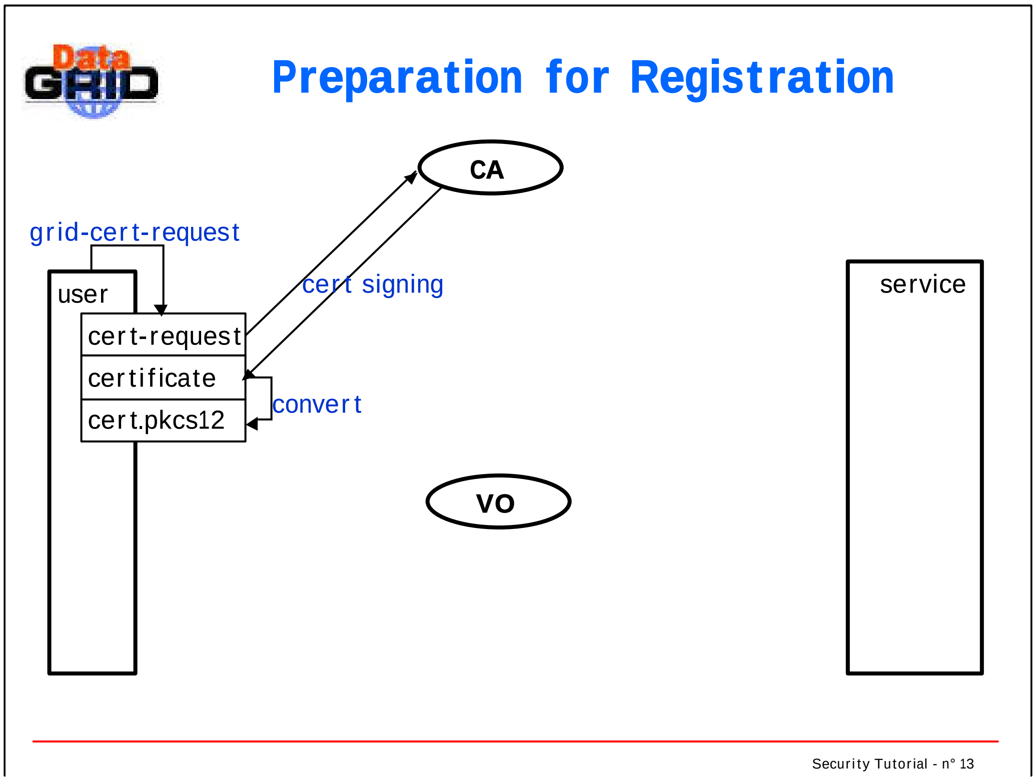
- the base URL for the CA – other information on the CA is here

- the type of the certificate: it can only be user as a client certificate, it is not valid to sign any other certificate and it cannot be used in a server

- revocation URL: checking the validity of this certificate shall include a check, that this certificate is not revoced. Revocation list of the issuer CA is published under this URL, so a service shall download this piece of information from here.

- establishing a trust relationship with a CA requires some information about how it registrates/issues certificates. The registration process and the format of the issued certificates are described in this policy.

The rest of the output is the ASN.1 format certificate with PEM encoding.



Preparation for Registration

1. user requests a certificate
2. user sends the requests to the CA, which sends back the signed certificate
3. conversion to browser digestable PKCS12 format

Registration/Authorization

User registration in an EDG Virtual Organisation

- convert your certificate:
  - **openssl pkcs12 -export -in ~/.globus/usercert.pem -inkey ~/.globus/userkey.pem -out user.p12 -name 'Joe Smith'**
- import your certificate in your browser
- sign the usage guidelines: https://marianne.in2p3.fr/cgi-bin/datagrid/register/account.pl
- ask an account from your VO administrator by email

-> You are registered in the VO-LDAP server and have a user account.

Netscape

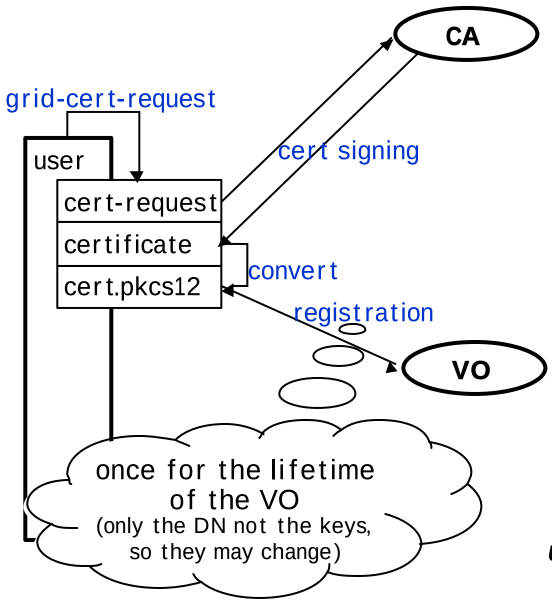
To import a certificate into Netscape, first start Netscape. Then select "Communicator->Tools->Security Info" from the top menus and then click on the "Certificates/yours" section. This will open a window where you will find an "Import a Certificate" button. You will be asked a new password to protect your Netscape certificates DB (keep track of it as you will need it to import other certificates). You will be able to choose your .p12 file from a standard file selection window.

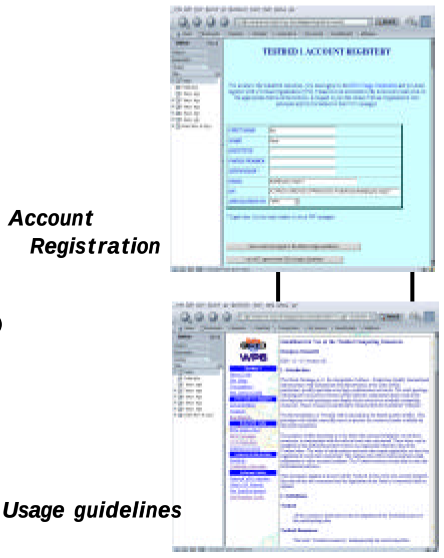
Internet Explorer

Go to the Tools menu and select Internet Options. Choose the Content tab and click on Certificates. In the new window click on Import to get the Import wizard, then follow the instructions. Select the file for import and give the password (it should go into the "Personal" certificate store).

You should also select high security otherwise IE remembers your pass phrase for you.

Registration





Security Tutorial - n° 15

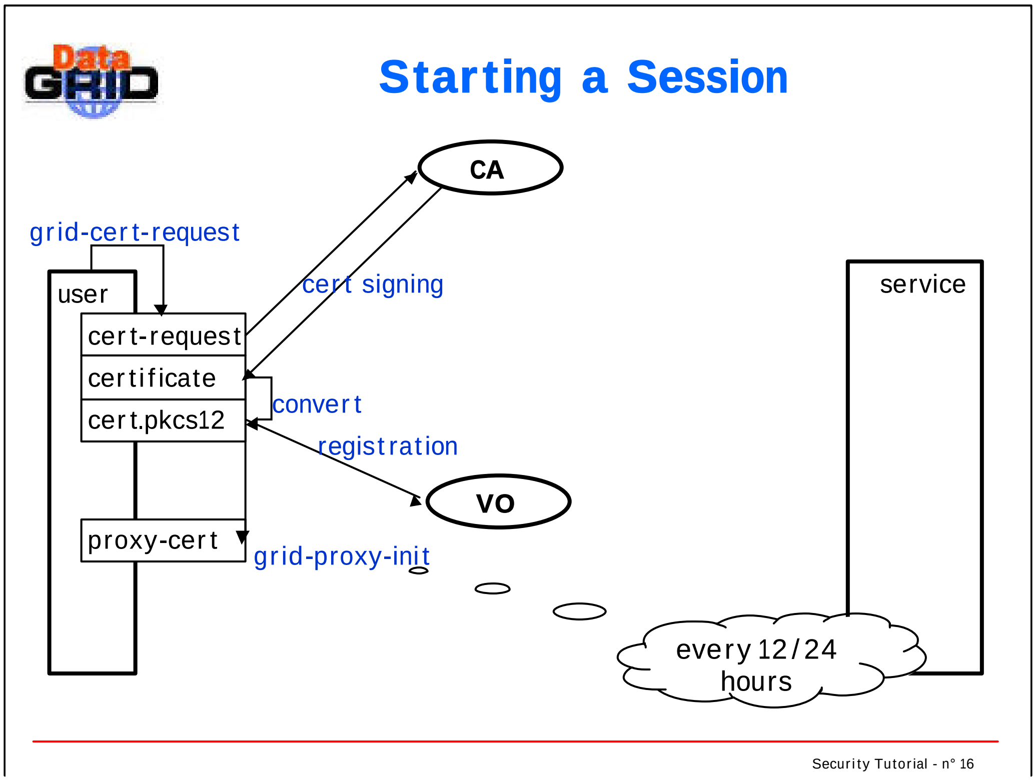
1. user requests a certificate
2. user sends the requests to the CA, which sends back the signed certificate
3. conversion to browser digestable PKCS12 format
4. registration in the EDG LDAP-VO

Real life example:

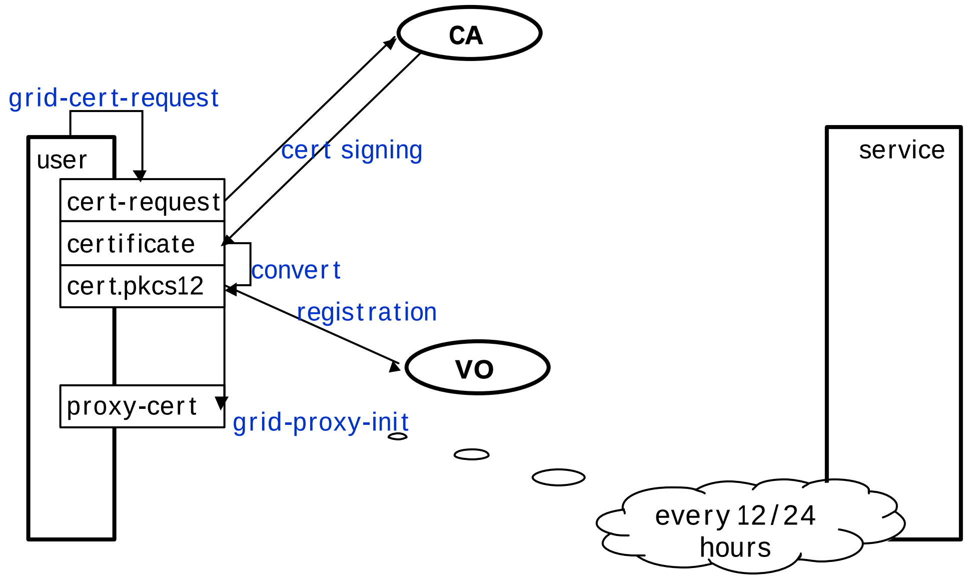
Before travelling to a foreign country you may need to apply for a visa. For this you need to fill out some forms and send them to the destination country's appropriate authority.

If everything is OK with you the authority will give you a visa, which you can show up at the border.

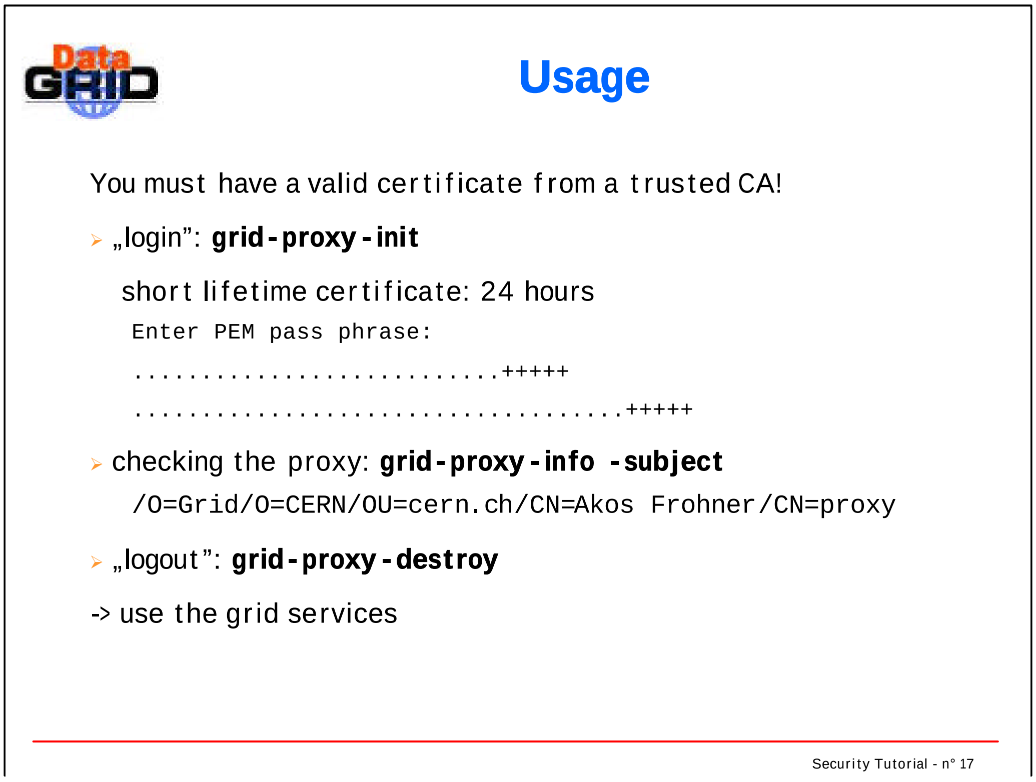
In the current EDG setup this visa is not sent to you, but the list of „good" users is „sent" to the sites, so they would know your status.



Starting a Session



1. user requests a certificate
2. user sends the requests to the CA, which sends back the signed certificate
3. conversion to browser digestable PKCS12 format
4. registration in the EDG LDAP-VO
5. generates a proxy certificate (24 hours lifetime)



Usage

You must have a valid certificate from a trusted CA!

- „login”: **grid-proxy-init**

  short lifetime certificate: 24 hours

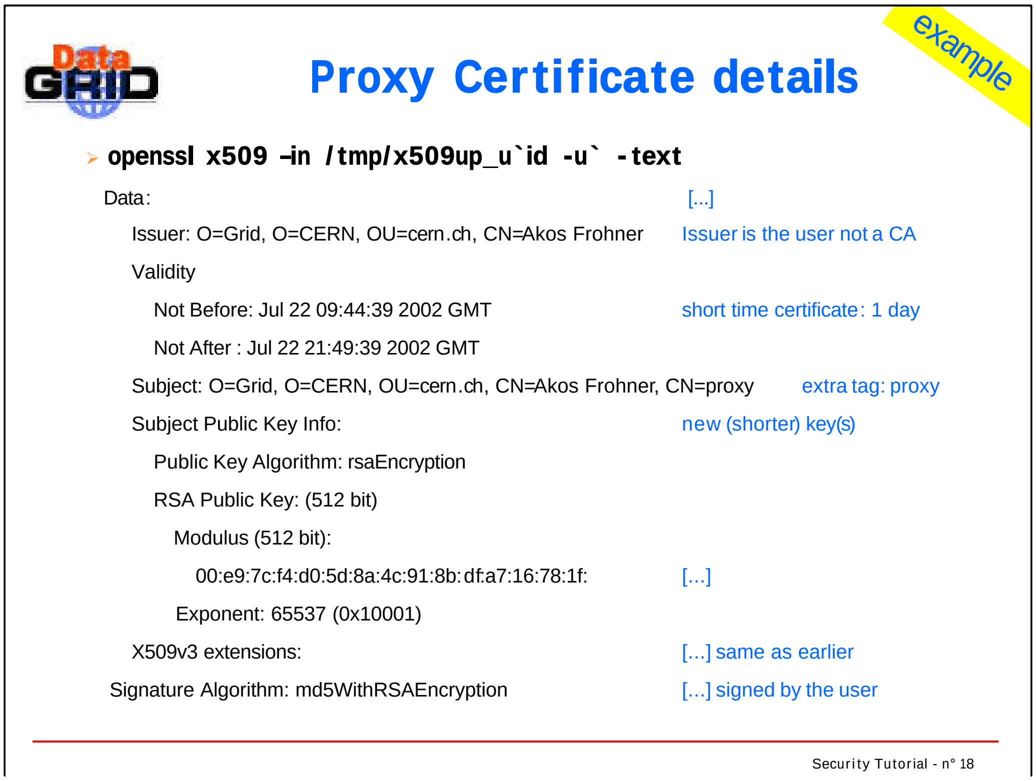
```
Enter PEM pass phrase:
...........................+++++
....................................+++++
```

- checking the proxy: **grid-proxy-info -subject**

```
/O=Grid/O=CERN/OU=cern.ch/CN=Akos Frohner/CN=proxy
```

- „logout”: **grid-proxy-destroy**

-> use the grid services



Proxy Certificate details

example

- **openssl x509 –in /tmp/x509up_u`id -u` -text**

```
Data:                                                              [...]
    Issuer: O=Grid, O=CERN, OU=cern.ch, CN=Akos Frohner          Issuer is the user not a CA
    Validity
      Not Before: Jul 22 09:44:39 2002 GMT                         short time certificate: 1 day
      Not After : Jul 22 21:49:39 2002 GMT
    Subject: O=Grid, O=CERN, OU=cern.ch, CN=Akos Frohner, CN=proxy      extra tag: proxy
    Subject Public Key Info:                                       new (shorter) key(s)
      Public Key Algorithm: rsaEncryption
      RSA Public Key: (512 bit)
        Modulus (512 bit):
          00:e9:7c:f4:d0:5d:8a:4c:91:8b:df:a7:16:78:1f:            [...]
        Exponent: 65537 (0x10001)
    X509v3 extensions:                                             [...] same as earlier
 Signature Algorithm: md5WithRSAEncryption                         [...] signed by the user
```

Security Tutorial - n° 18

The certificate created by grid-proxy-init is a temporary one only: it is valid only for one day.

This certificate has some specialities: the subject name is almost the same one as the real cert's subject name, but there is an extra field: „CN=proxy”.

grid-proxy-init basically creates a new certificate request with a short, temporary key:

Proxy-Kpub, Proxy-Info, (Proxy-Kpriv)

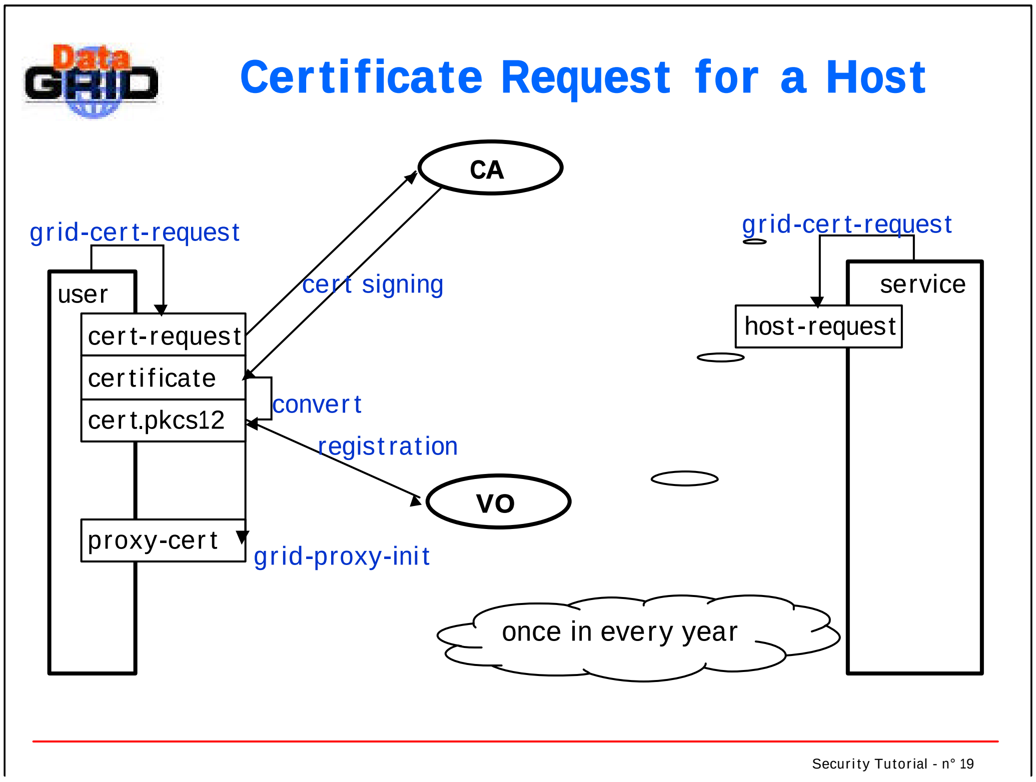
this information is used to create the new certificate:

signature = encrypt(Kpub, hash(Proxy-Info, Proxy-Kpub))

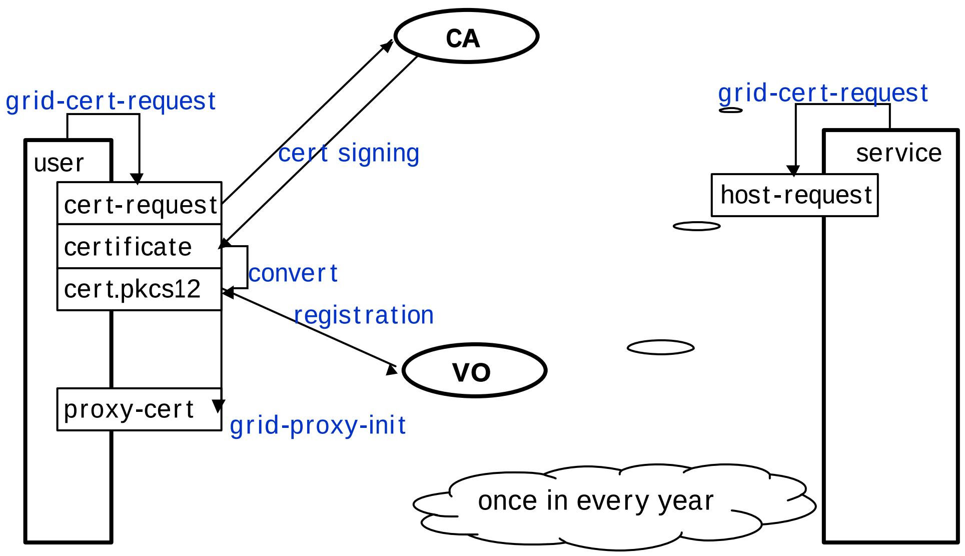
Advantage: you don't deal with revocation list (as the CA)

Drawback: it is valid only for one day

(remark: the proxy certificate is signed by a normal client certificate, thus an x509.3 compilant software will refuse it. It will be accepted only by GSI libraries/softwares.)



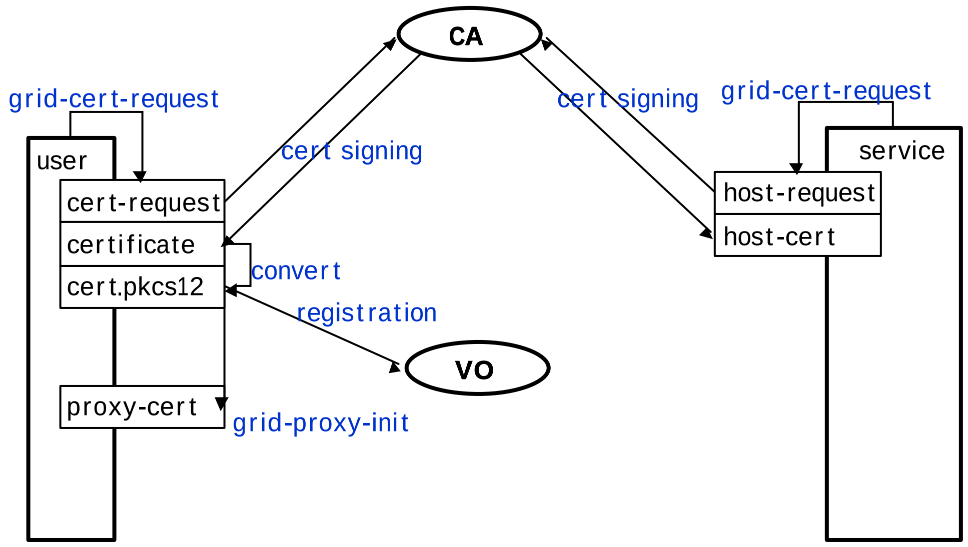
Certificate Request for a Host



Security Tutorial - n° 19

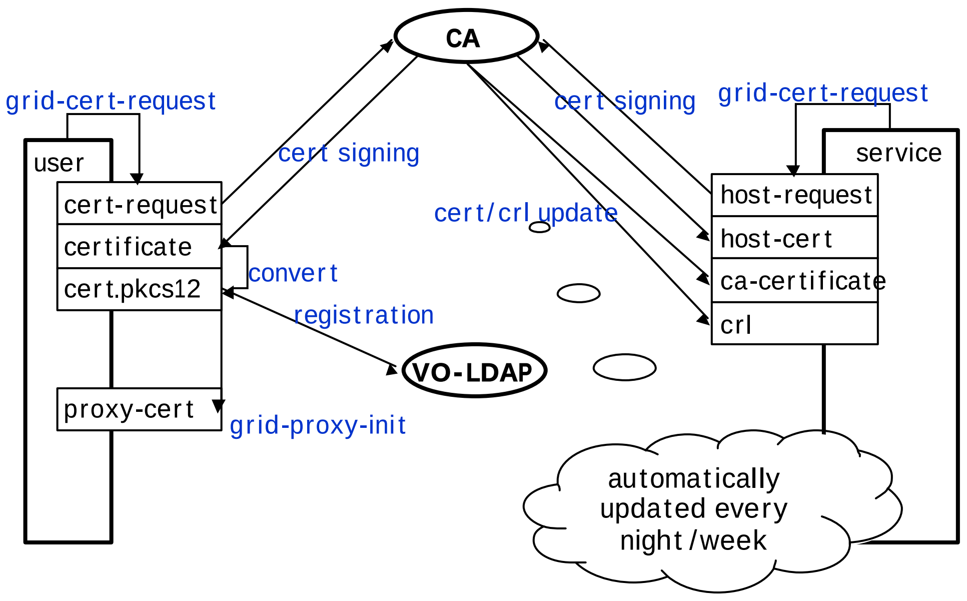
1. user requests a certificate
2. user sends the requests to the CA, which sends back the signed certificate
3. conversion to browser digestable PKCS12 format
4. registration in the EDG LDAP-VO
5. generats a proxy certificate (24 hours lifetime)
6. host/service requests a certificate

Signing the Certificate



1. user requests a certificate
2. user sends the requests to the CA, which sends back the signed certificate
3. conversion to browser digestable PKCS12 format
4. registration in the EDG LDAP-VO
5. generats a proxy certificate (24 hours lifetime)
6. host/service requests a certificate
7. host/service sends the requests to the CA, which sends back the signed certificate

Configuration on the Server



Security Tutorial - n° 21

1. user requests a certificate
2. user sends the requests to the CA, which sends back the signed certificate
3. conversion to browser digestable PKCS12 format
4. registration in the EDG LDAP-VO
5. generats a proxy certificate (24 hours lifetime)
6. host/service requests a certificate
7. host/service sends the requests to the CA, which sends back the signed certificate
8. retrival of the trusted CA certificates and their CRLs

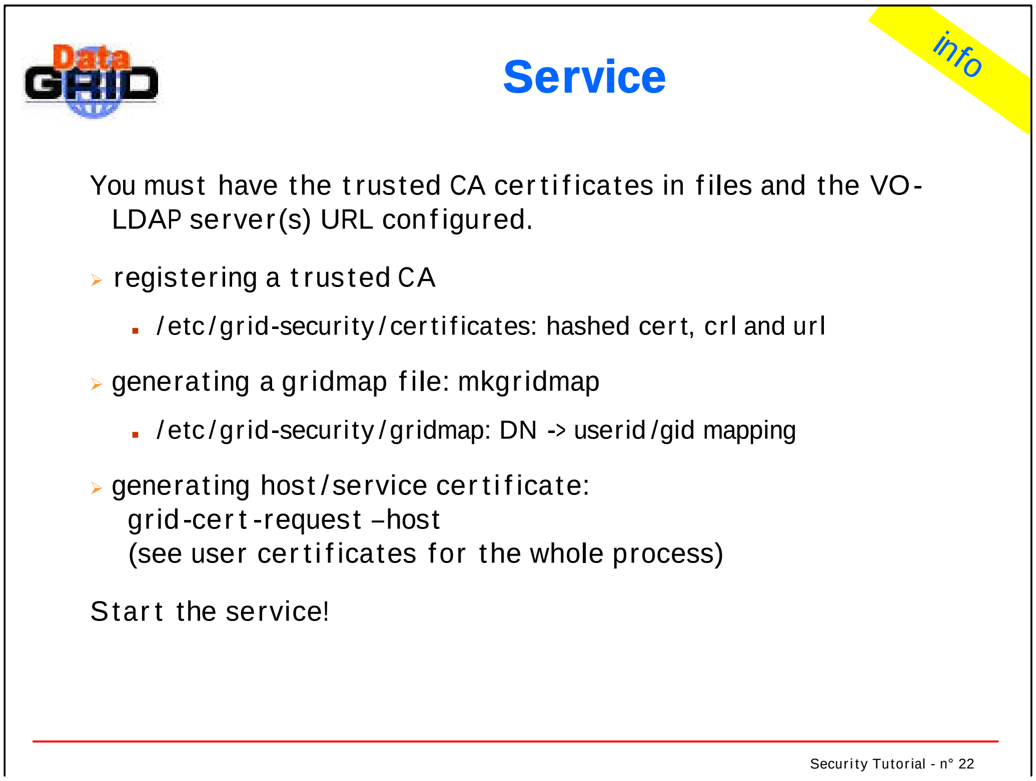
Real life example:

The officers at the border has to know all the valid passport types, which would be shown to him He learns it by looking at example passports from the various countries.

The ca-certificate plays this role in this setup: the service can validate a user certificate by using the ca-certificate. So the list of valid CA certificates is similar to the set of passports accepted at a border.

A passport can be stolen, so a country time-to-time issues a black list of the invalid numbers.

The Certificate Revocation List (CRL) is this list for the CA: the list of invalid certificate numers signed by this CA.



Service

info

You must have the trusted CA certificates in files and the VO-LDAP server(s) URL configured.

- registering a trusted CA
  - /etc/grid-security/certificates: hashed cert, crl and url
- generating a gridmap file: mkgridmap
  - /etc/grid-security/gridmap: DN -> userid/gid mapping
- generating host/service certificate:
  grid-cert-request –host
  (see user certificates for the whole process)

Start the service!

certificates:

hash: openssl x509 –hash –noout –in cacert.pem

cacert: <hash>.0

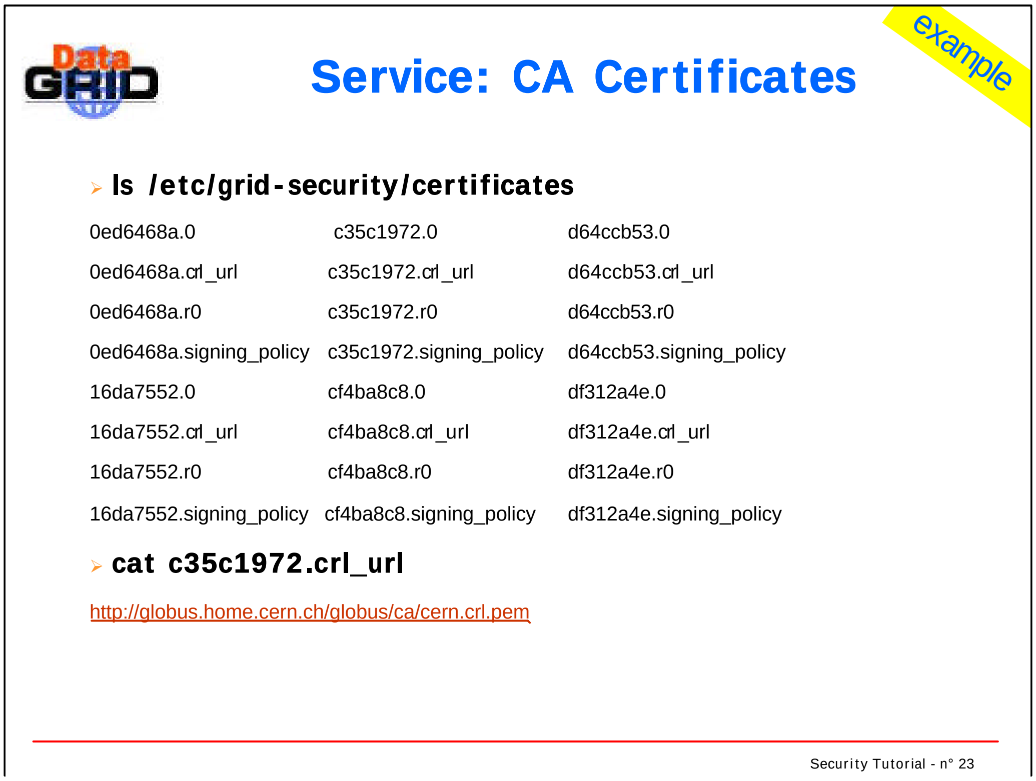
crl: <hash>.crl

url: <hash>.crl_url

used by edg-update-crl

mkgridmap assumes that the VO LDAP servers and their local groups are configured

host certificate: by default it doesn't have a password on the private key, so it must be better protected!



Service: CA Certificates

example

- **ls /etc/grid-security/certificates**

```
0ed6468a.0                 c35c1972.0                 d64ccb53.0
0ed6468a.crl_url           c35c1972.crl_url           d64ccb53.crl_url
0ed6468a.r0                c35c1972.r0                d64ccb53.r0
0ed6468a.signing_policy    c35c1972.signing_policy    d64ccb53.signing_policy
16da7552.0                 cf4ba8c8.0                 df312a4e.0
16da7552.crl_url           cf4ba8c8.crl_url           df312a4e.crl_url
16da7552.r0                cf4ba8c8.r0                df312a4e.r0
16da7552.signing_policy    cf4ba8c8.signing_policy    df312a4e.signing_policy
```

- **cat c35c1972.crl_url**

```
http://globus.home.cern.ch/globus/ca/cern.crl.pem
```

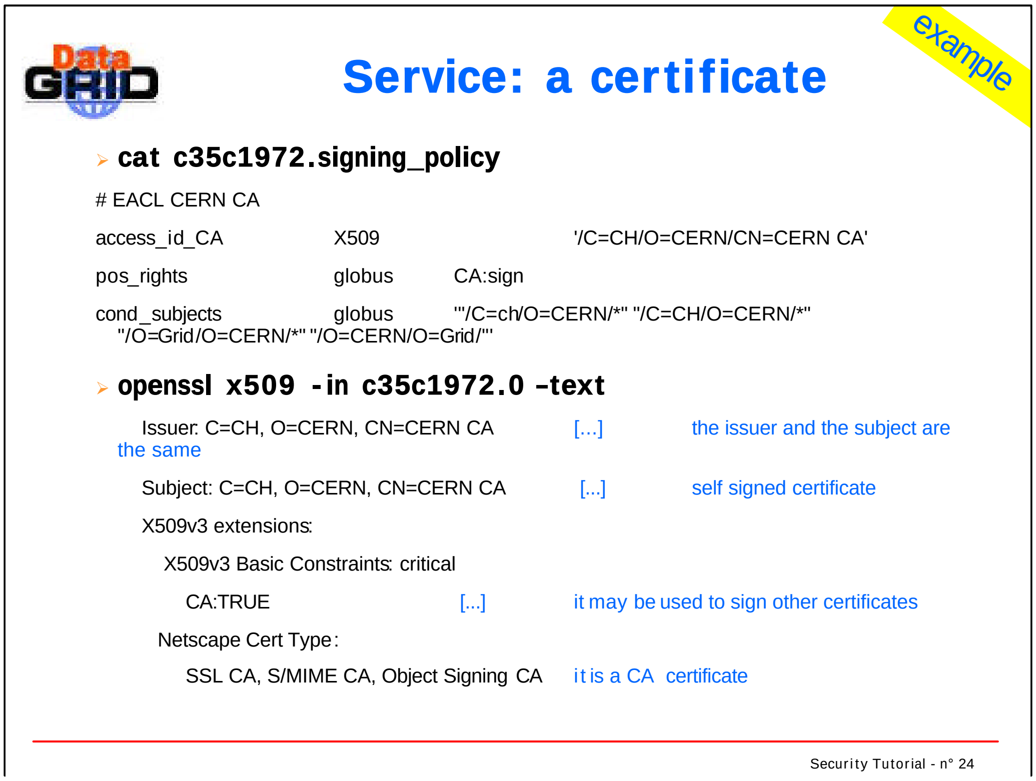
Security Tutorial - n° 23

The /etc/grid-security/certificate directory contains information on the known CAs:

- CA certificate	.0
- CA crl	.r0
- CA crl update URL	.crl_url
- signing policy	.signing_policy

The update URL is the source location of the corresponding CRL file (.r0). A utility (edg-crl-update) will download the updated CRLs from every CA regularly. This update algorithm uses the .crl_url files to locate the sources, although most of the CA certificates also contain this information.

In our example we show the CERN CA.



Service: a certificate

- **cat c35c1972.signing_policy**

```
# EACL CERN CA
access_id_CA          X509                  '/C=CH/O=CERN/CN=CERN CA'
pos_rights            globus       CA:sign
cond_subjects         globus       '"/C=ch/O=CERN/*" "/C=CH/O=CERN/*"
  "/O=Grid/O=CERN/*" "/O=CERN/O=Grid/"'
```

- **openssl x509 -in c35c1972.0 -text**

```
    Issuer: C=CH, O=CERN, CN=CERN CA          [...]      the issuer and the subject are
  the same
    Subject: C=CH, O=CERN, CN=CERN CA          [...]      self signed certificate
    X509v3 extensions:
      X509v3 Basic Constraints: critical
        CA:TRUE                       [...]     it may be used to sign other certificates
      Netscape Cert Type:
        SSL CA, S/MIME CA, Object Signing CA   it is a CA  certificate
```

Security Tutorial - n° 24

The signing policy describes the

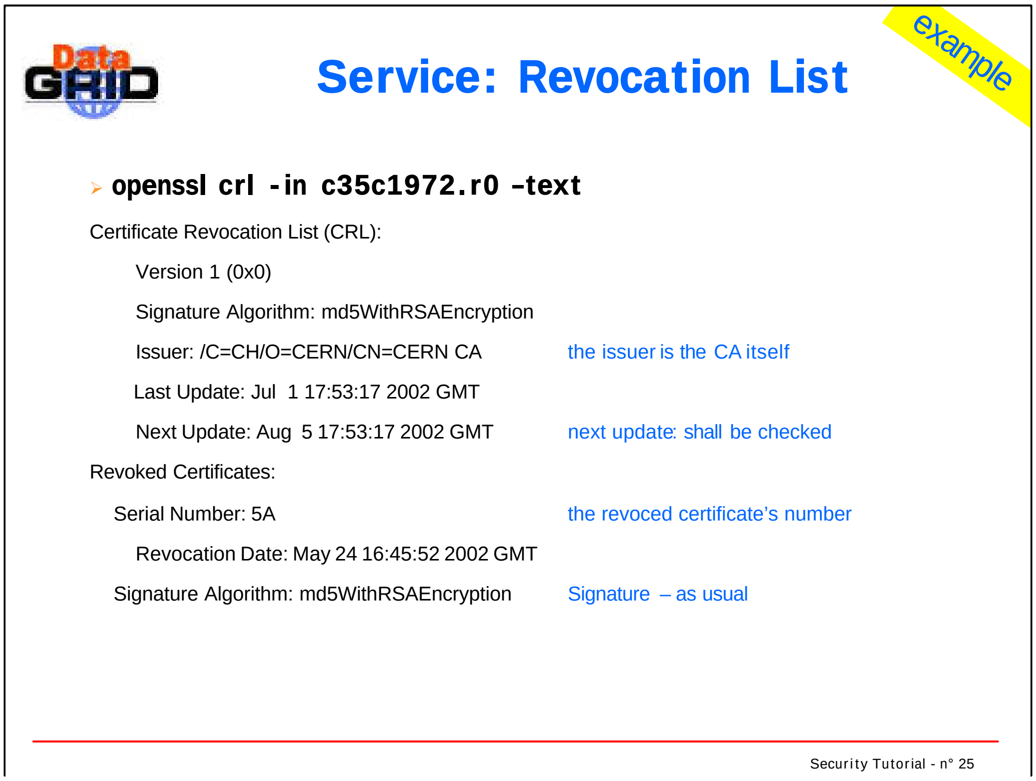
- DN of the CA
- and a pattern for the valid DNs signed by this CA

A service shall not accept any certificate, which doesn’t match this pattern.

(remark: the latest x509.3 RFC provides Basic Constraints to embed this information into the CA certificate, but the globus framework was created before this „standard”)

The CA certificate is a self-signed certificate. One may decide to get a signature from another CA (e.g. a commercial CA, like VeriSing), but for grid purposes the trust relationship can be established with self-signed certificates as well.

New attributes: „CA: TRUE” basic constraint and the CA bits in the certificate type field. These attributes will enable this certificate to be used for signing any other certificate. It will be accepted in any x509.3 compilant software, not only in grid applications.



example

Service: Revocation List

➢ **openssl crl -in c35c1972.r0 –text**

```
Certificate Revocation List (CRL):
        Version 1 (0x0)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: /C=CH/O=CERN/CN=CERN CA          the issuer is the CA itself
        Last Update: Jul  1 17:53:17 2002 GMT
        Next Update: Aug  5 17:53:17 2002 GMT    next update: shall be checked
Revoked Certificates:
    Serial Number: 5A                            the revoced certificate's number
        Revocation Date: May 24 16:45:52 2002 GMT
    Signature Algorithm: md5WithRSAEncryption    Signature – as usual
```

Security Tutorial - n° 25

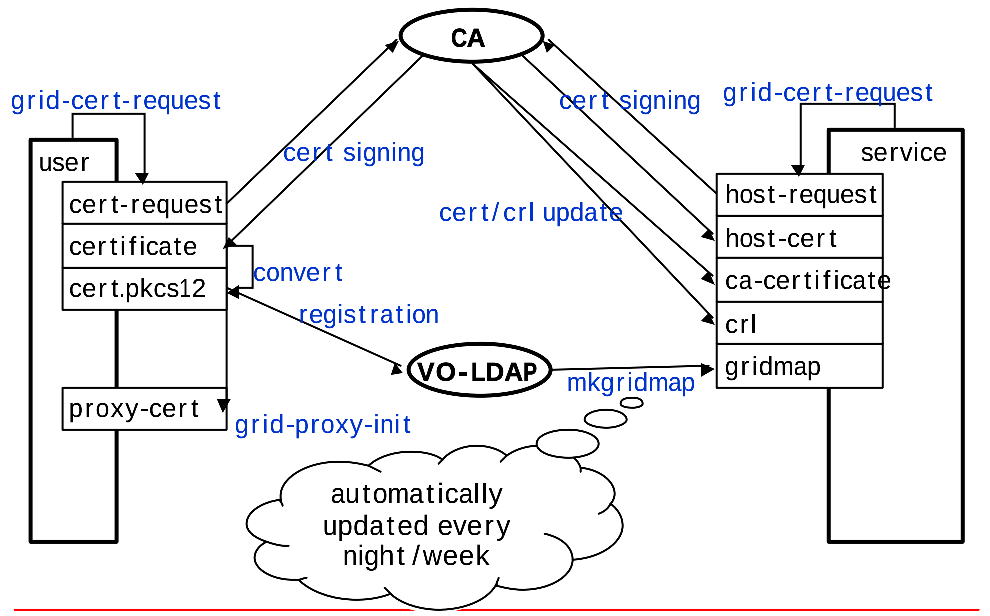
Revocation list: list of revoced certificate numbers

A CRL is similar to a certificate: it has an information field (list of serial numbers) and a signature.

The CRL has a validity period, so the next version shall be downloaded by the service regularly.

The time of the next update is usually marked in the CRL itself.

Authorization Information

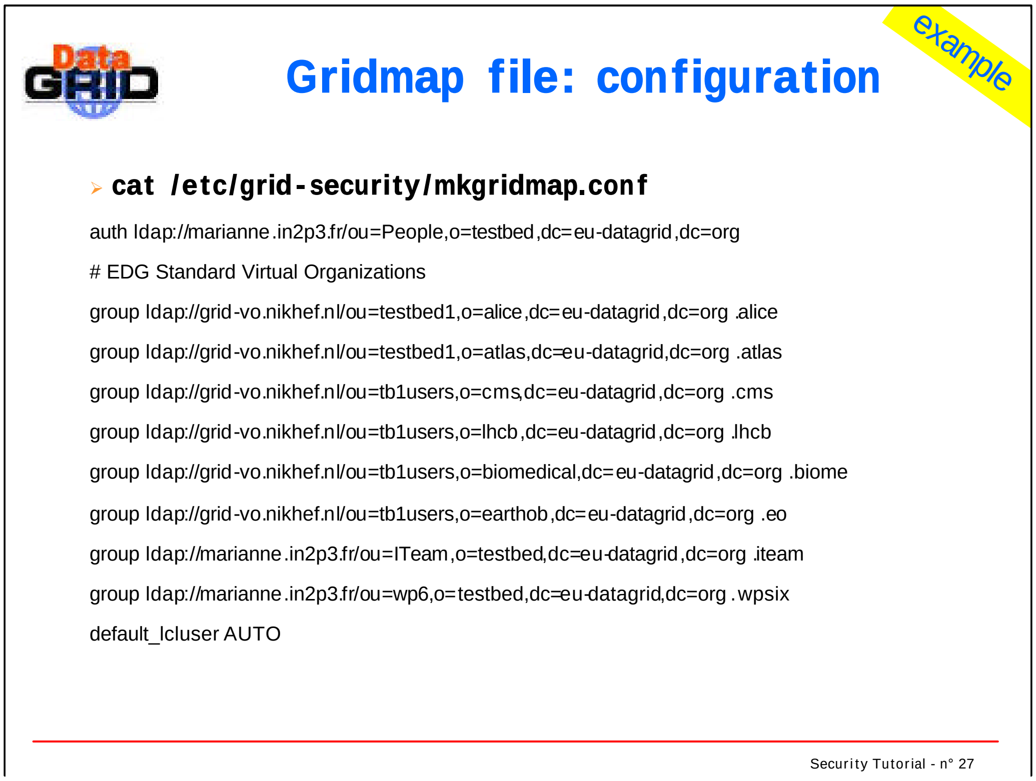


1. user requests a certificate
2. user sends the requests to the CA, which sends back the signed certificate
3. conversion to browser digestable PKCS12 format
4. registration in the EDG LDAP-VO
5. generats a proxy certificate (24 hours lifetime)
6. host/service requests a certificate
7. host/service sends the requests to the CA, which sends back the signed certificate
8. retrival of the trusted CA certificates and their CRLs
9. generating a gridmap file from the LDAP database for authorization and mapping

Real life example:

Normally a visa is bind (glued into) the traveller's passport, so an officer at the border can check it without contacting the country's central authority, which issued it.

In the current EDG setup the services has to download the list of valid passports, which are acceptable for the VO. The list of valid certificates is stored in the gridmap-file. This will change in the future to mimic the behaviour of our real life example.



Gridmap file: configuration

example

➢ **cat /etc/grid-security/mkgridmap.conf**

```
auth ldap://marianne.in2p3.fr/ou=People,o=testbed,dc=eu-datagrid,dc=org
# EDG Standard Virtual Organizations
group ldap://grid-vo.nikhef.nl/ou=testbed1,o=alice,dc=eu-datagrid,dc=org .alice
group ldap://grid-vo.nikhef.nl/ou=testbed1,o=atlas,dc=eu-datagrid,dc=org .atlas
group ldap://grid-vo.nikhef.nl/ou=tb1users,o=cms,dc=eu-datagrid,dc=org .cms
group ldap://grid-vo.nikhef.nl/ou=tb1users,o=lhcb,dc=eu-datagrid,dc=org .lhcb
group ldap://grid-vo.nikhef.nl/ou=tb1users,o=biomedical,dc=eu-datagrid,dc=org .biome
group ldap://grid-vo.nikhef.nl/ou=tb1users,o=earthob,dc=eu-datagrid,dc=org .eo
group ldap://marianne.in2p3.fr/ou=ITeam,o=testbed,dc=eu-datagrid,dc=org .iteam
group ldap://marianne.in2p3.fr/ou=wp6,o=testbed,dc=eu-datagrid,dc=org .wpsix
default_lcluser AUTO
```

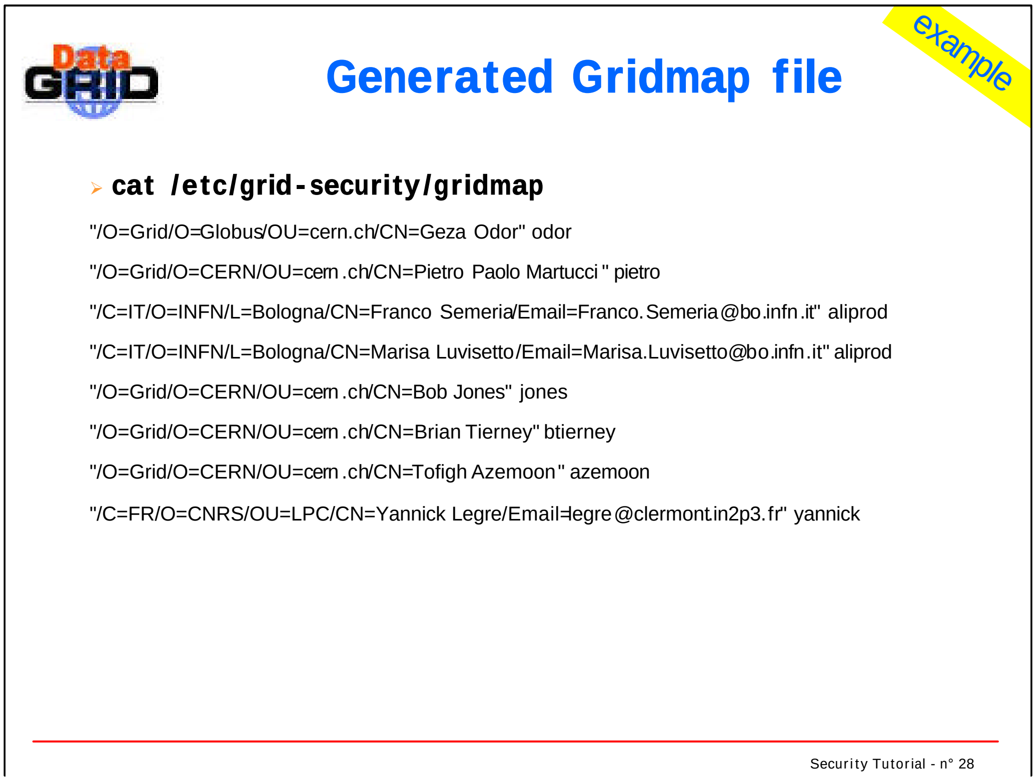
Security Tutorial - n° 27

configuration for mkgridmap

auth: listing for DNs, who are signed the EDG usage guidelines, thus they can be a member of any EDG VO

(this is a precondition to be a user anywhere)

group: the VOs are represented by local groups. If a user is a member of a VO, then this configuration file will tell to the mkgridmap script the group to be assigned to the user.



Generated Gridmap file

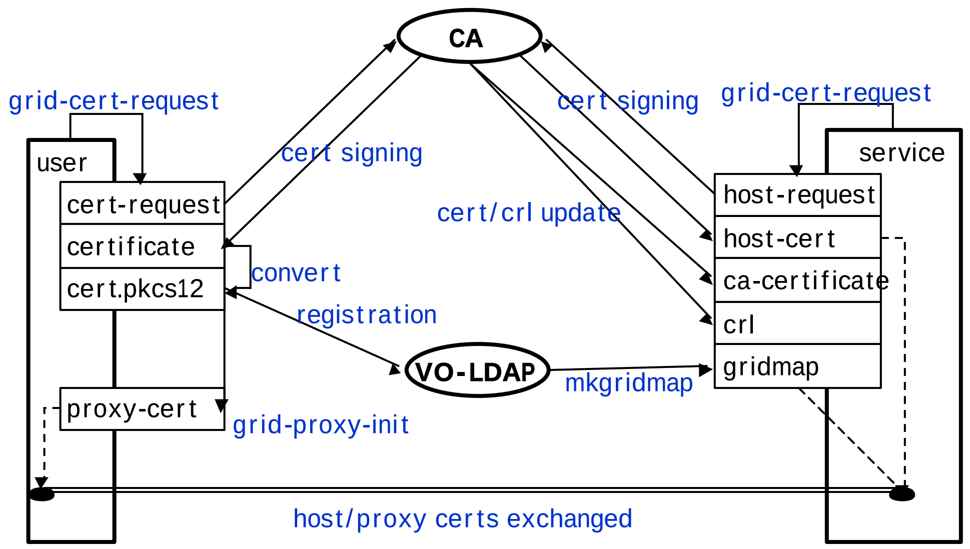
example

- **cat /etc/grid-security/gridmap**

```
"/O=Grid/O=Globus/OU=cern.ch/CN=Geza Odor" odor
"/O=Grid/O=CERN/OU=cern.ch/CN=Pietro Paolo Martucci" pietro
"/C=IT/O=INFN/L=Bologna/CN=Franco Semeria/Email=Franco.Semeria@bo.infn.it" aliprod
"/C=IT/O=INFN/L=Bologna/CN=Marisa Luvisetto/Email=Marisa.Luvisetto@bo.infn.it" aliprod
"/O=Grid/O=CERN/OU=cern.ch/CN=Bob Jones" jones
"/O=Grid/O=CERN/OU=cern.ch/CN=Brian Tierney" btierney
"/O=Grid/O=CERN/OU=cern.ch/CN=Tofigh Azemoon" azemoon
"/C=FR/O=CNRS/OU=LPC/CN=Yannick Legre/Email=legre@clermont.in2p3.fr" yannick
```

DN -> local userid mapping

Using a Service



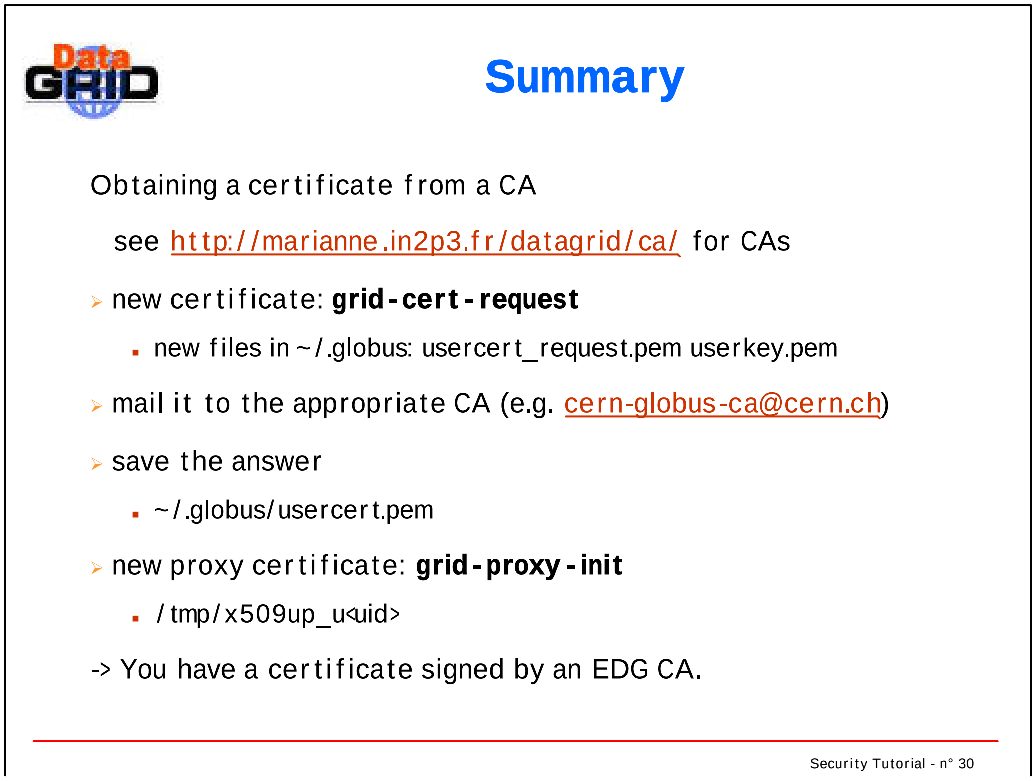
Security Tutorial - n° 29

1. user requests a certificate (once in every two-three years)
2. user sends the requests to the CA, which sends back the signed certificate
3. conversion to browser digestable PKCS12 format
4. registration in the EDG LDAP-VO (once for the lifetime of the VO – you may change the certificate keys!)
5. generats a proxy certificate (24 hours lifetime)
6. host/service requests a certificate
7. host/service sends the requests to the CA, which sends back the signed certificate
8. retrival of the trusted CA certificates and their CRLs (automatically updated every night/week)
9. generating a gridmap file from the LDAP database for authorization and mapping (automatically updated every night/week)
10. user contacts a service: they exchange their certificates to authenticate each other; the service bases its authorization decision on the gridmap file (... currently)

Real life example:

The traveller goes to the destination country and shows up his passport at the border. The officer checks its validity by comparing it to the example passport from the issuing country. He also looks at the list of revoked passports (coming from the originating country) and the visa, which allows the traveller to enter the country.

The traveller may also ask the officer to identify itself, so the authentication is also symmetric in the real life.



Summary

Obtaining a certificate from a CA

see http://marianne.in2p3.fr/datagrid/ca/ for CAs

- new certificate: **grid-cert-request**
  - new files in ~/.globus: usercert_request.pem userkey.pem
- mail it to the appropriate CA (e.g. cern-globus-ca@cern.ch)
- save the answer
  - ~/.globus/usercert.pem
- new proxy certificate: **grid-proxy-init**
  - /tmp/x509up_u<uid>

-> You have a certificate signed by an EDG CA.

Security Tutorial - n° 30

create a new certificate request in ~/.globus

print the request

```
openssl req –in ~/.globus/usercert_request.pem –
text
```

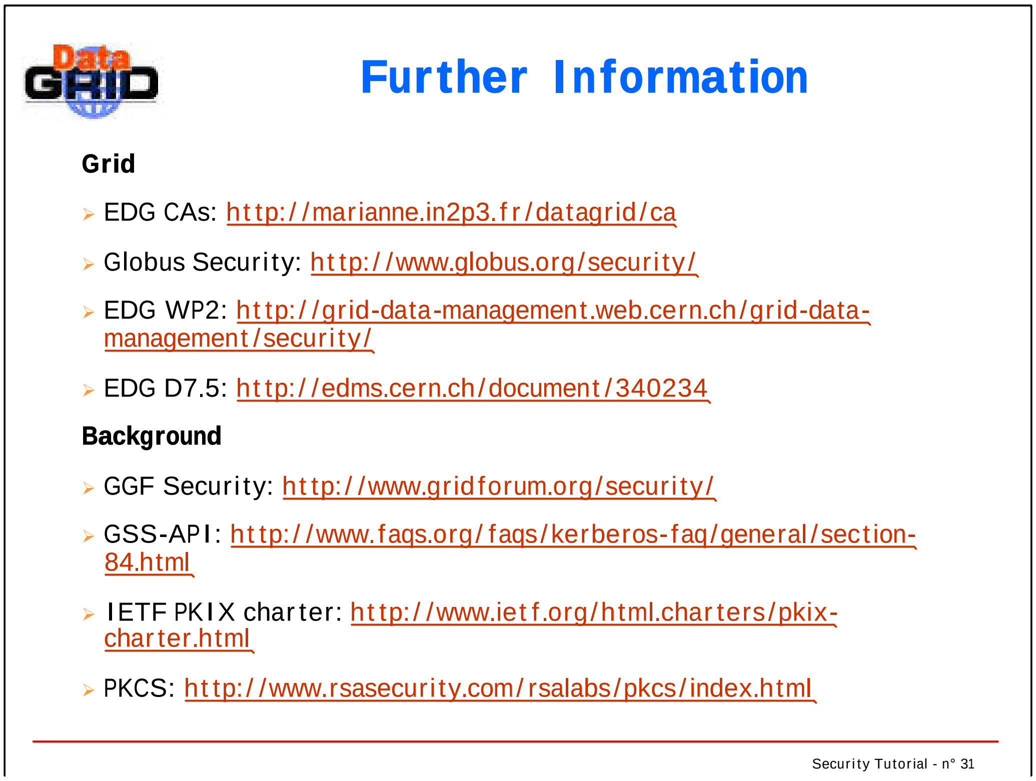
print the certificate

```
openssl x509 –in ~/.globus/usercert.pem –text
```

print the proxy certificate

```
openssl x509 –in /tmp/x509up_u`id -u` -text
```

show the differences in the validity periods!



Further Information

Grid

- EDG CAs: http://marianne.in2p3.fr/datagrid/ca
- Globus Security: http://www.globus.org/security/
- EDG WP2: http://grid-data-management.web.cern.ch/grid-data-management/security/
- EDG D7.5: http://edms.cern.ch/document/340234

Background

- GGF Security: http://www.gridforum.org/security/
- GSS-API: http://www.faqs.org/faqs/kerberos-faq/general/section-84.html
- IETF PKIX charter: http://www.ietf.org/html.charters/pkix-charter.html
- PKCS: http://www.rsasecurity.com/rsalabs/pkcs/index.html