CINECA

consorzio interuniversitario

How to handle X.509 certificates in CINECA

Giacomo Mariani

hpc-mw@cineca.it

23/02/2011

Table of contents

1 Introduction

This document provides an overview of X.509 certificates and how to use them in CINECA, starting from the request of a personal keystore towards the steps needed to adopt it with different tools.

2 X.509 Certificate

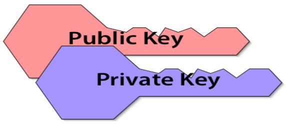
X.509 Certificates are documents based on the X.509 standard: the widest used for accounting and authentication mechanism [1].

In fact a certificate:

- provides a way to **secure authentication** between nodes of a computational system;
- support security **across organizational boundaries**, thus avoiding a centrally-managed security system;
- support "**single sign-on**" for users, including delegation of credentials for computations that involve multiple resources and/or sites.

This is possible by means of an asymmetric cryptography base model (PKI) : a user (or entity) holds a **key pair (called keystore)**:

- one **private key**, known only to the user
- one **public key**, distributable to the world



A message encrypted with one key requires the other key for decryption.

As the name suggests: public keys can be freely distributed, allowing messages to be encrypted just for/to you, while your private key doesn't have to go around.

An **X.509 certificate** is a standardized way to handle your keys. It is a file which contains the **sign** of a **trusted issuer,** called **certificate authority** (CA). Examples of CA are **INFN CA [2]** and **Cacert [3].** It's worth noting that **INFN CA** is the only italian CA recognized abroad Europe, through **EUGridPMA [4]** that is an organization which is states who has to be trusted in various european projects: in particular **DEISA [5]**, **PRACE [6]** and other european **HPC** initiatives. This means that if you want to access PRACE resource, you need to hold a INFN certificate. Moreover CINECA has an internal Registration Authority (RA), based at the DSET, which can directly connect SCAI employees to INFN.

Your keys are not the only information contained in your certificate. In fact it will always contain also the following information:

- Entity's qualified name.

- Entity’s public key.
- Name of the issuing CA.
- Signature of issuing CA.
- Validity dates (start and end dates).
- Your name (in the DN, that is “Distinguished Name”, format).

The DN is very important: it let you be identified, universally around the world! It is written as a “Backslash Separated Values” string which states who you are, which are your organization and country, who gave you that certificate and so on. For example, a sample INFN certificate reports: “/C=IT/O=INFN/OU=Personal Certificate/L=CINECA-SAP/CN=Giacomo Mariani”.

3 How to get your certificate

Usually the CA of your country has established a network of *Registration Authorities* (RA) where users can apply for their certificate. Users who need a X.509 certificate should contact either a RA which is closely related to the user's institution or the user should contact a RA who is closer. If a corresponding RA is not available or can not be identified (e.g., check the list of *Registration Authorities* that is provided by the national CA), the national CA should be contacted directly.

Details concerning the generation and submission of a certificate request depends on the procedures that the national CA of your country requires. You can find your national CA at the EUGridPMA website, the authority which maintains a list of CAs: http://www.eugridpma.org/members/worldmap.

3.1 How to get your INFN certificate if you are a CINECA employee

The creation of a new certificate will require only **four steps**:

1. Ask for a new certificate through Service Desk:

   https://servicedesk.cineca.it/node/992 (selecting for “Tipo di certificato” the voice “personale” and for “Tipo di operazione sul certificato” the voice ”richiesta”).
2. Wait for DSET to call you and go there with your ID (identity card).
3. In not more than 15 days you will receive an email from INFN.
4. Use your browser to open the link content in the previous email.

You got it!

If you want to see or manage your new certificate, in your browser, you just have to follow a path similar to the following:

- for **Firefox**: `Edit → Preferences → Advanced → Encryption → View Certificates → Your Certificates`

- for **Iexplorer**: Tool → Internet Option → Content → Certificates

4 Certificates management tools

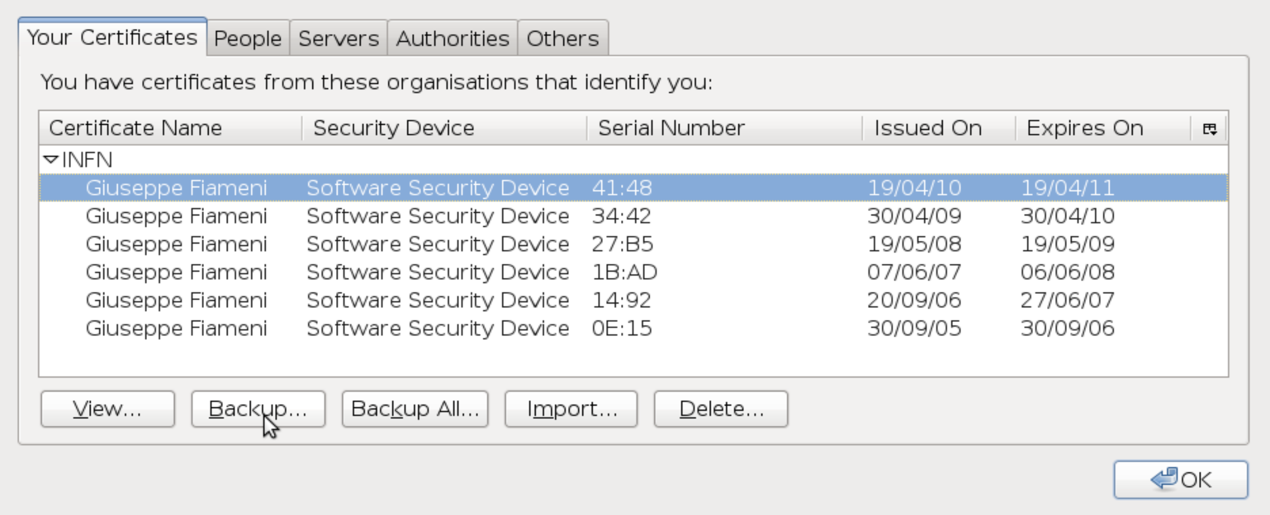
In order to enjoy this more modern way of authenticate yourself you have to get your certificate and instruct your software on where to find it. It's like your ID card, you should show it to who wants to know your identity, at least the first time you meet! Moreover, if you want your passport in order to go far abroad you should own an ID... that's something like changing the format of your certificates to have it understood by different software.

So, now you have a certificate, issued by INFN CA, saved in your browser. Using the browser tools you have now to “export” (or backup) it.

Firefox
Edit → Preferences → Advanced → Encryption → View Certificates → Your Certificates → Backup”

Iexplorer
Tool → Internet Option → Content → Certificates → Export

This gives you a (potentially crypted) file with a **p12** extension, your ID card.



You have certificates from these organisations that identify you:

Certificate Name	Security Device	Serial Number	Issued On	Expires On
INFN				
Giuseppe Fiameni	Software Security Device	41:48	19/04/10	19/04/11
Giuseppe Fiameni	Software Security Device	34:42	30/04/09	30/04/10
Giuseppe Fiameni	Software Security Device	27:B5	19/05/08	19/05/09
Giuseppe Fiameni	Software Security Device	1B:AD	07/06/07	06/06/08
Giuseppe Fiameni	Software Security Device	14:92	20/09/06	27/06/07
Giuseppe Fiameni	Software Security Device	0E:15	30/09/05	30/09/06

Drawing 1: A screenshot of a Certificate-Backup in Firefox.

4.1 OpenSSL [11]

OpenSSL is the most powerful tool for key-managing (and more), but is a command line tool. If you are a *NIX user you can get it with your usaul packageager, if it's not installed by default. Vice versa, if you are a Windows user, you can get it at [12]: you will need [13] and, if not already installed, [14]. Anyway you only need two commands in order to achieve the **pem** certificate, your passport:

```
bash$ openssl pkcs12 -clcerts -nokeys -in cert.p12 -out
```

```
usercert.pem
Enter Import Password: <password utilizzata per il backup>
MAC verified OK

bash$ openssl pkcs12 -nocerts -in cert.p12 -out userkey.pem
Enter Import Password: <password utilizzata per il backup>
MAC verified OK
Enter PEM pass phrase: <password per criptare chiave privata>
Verifying - Enter PEM pass phrase: <password per criptare chiave
privata>
```

4.2 Use case

Now you have to give this “public” form of your certificate to any application that requires it. In the following section the two most common use cases are discuscussed: let's now see how to configure GridFTP, GSI-SSH and UNICORE6 in order to use our certificates, i.e. where to put our certificate.

4.2.1 Globus Toolkit

GridFTP (a very efficient file transfer protocol) and GSI-SSH (a certificate based ssh-like tool destuned to grid environment) are part of the “Globus Toolkit” and, as all the other executables (globus-url-copy, uberFTP, and their graphycal interfaces, for example) use a common identification mechanism: **GSI-proxy**.

A GSI-proxy is a child of your X.509 Certificate, like a signed and dated copy of your ID card: this allows **restricted delegation**, i.e. the authentication of your processes in the computational infrastructure (for example the **DEISA** network).

Moreover, while a certificate usually lasts a year, a GSI-proxy certificate usually lasts 12 hours: this minimize the risk if your proxy is stolen because a thief will be able to act like you only for a short period of time.

As a primary step you have to copy your **pem** files (see below) on your workstation:

1. create a certificate directory:
   - `mkdir ~/.globus/`
2. put there your **pem** certificate and key and give them the right permissions as in the following example:
   - `$ ls -l .globus/`
   - `-rw-r--r--  1 jack  jack    1806 Sep 28 12:20 usercert.pem`

- -rw------- 1 jack jack   1751 Sep 28 12:19 userkey.pem

If you plain to use a GUI, like UberFTP or CoG [7], the software will ask you for the passphrase and create the prxy for you[1].

If you use command-line utilities you have to:

1. set the globus environment for *bash*
   - user$ export GLOBUS_LOCATION=/PATH/to/GLOBUS_INSTALLATION_DIRECTORY/
   - user$. $GLOBUS_LOCATION/etc/globus-user-env.sh

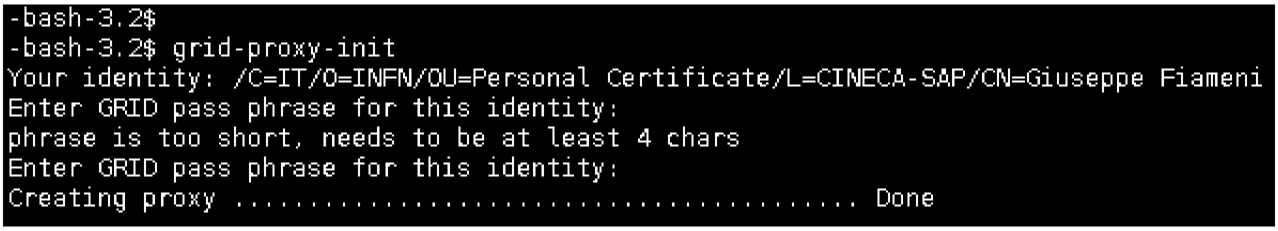
or

2. set the glocus environment for *csh*
   - user$ setenv GLOBUS_LOCATION /PATH/to/GLOBUS_INSTALLATION_DIRECTORY/
   - user$ source $GLOBUS_LOCATION/etc/globus-user-env.csh

and

3. create your proxy:
   - $ grid-proxy-init

Now you can inspect your proxy, inserting the passphrase you used to crypt your key, in order to verify your identity:



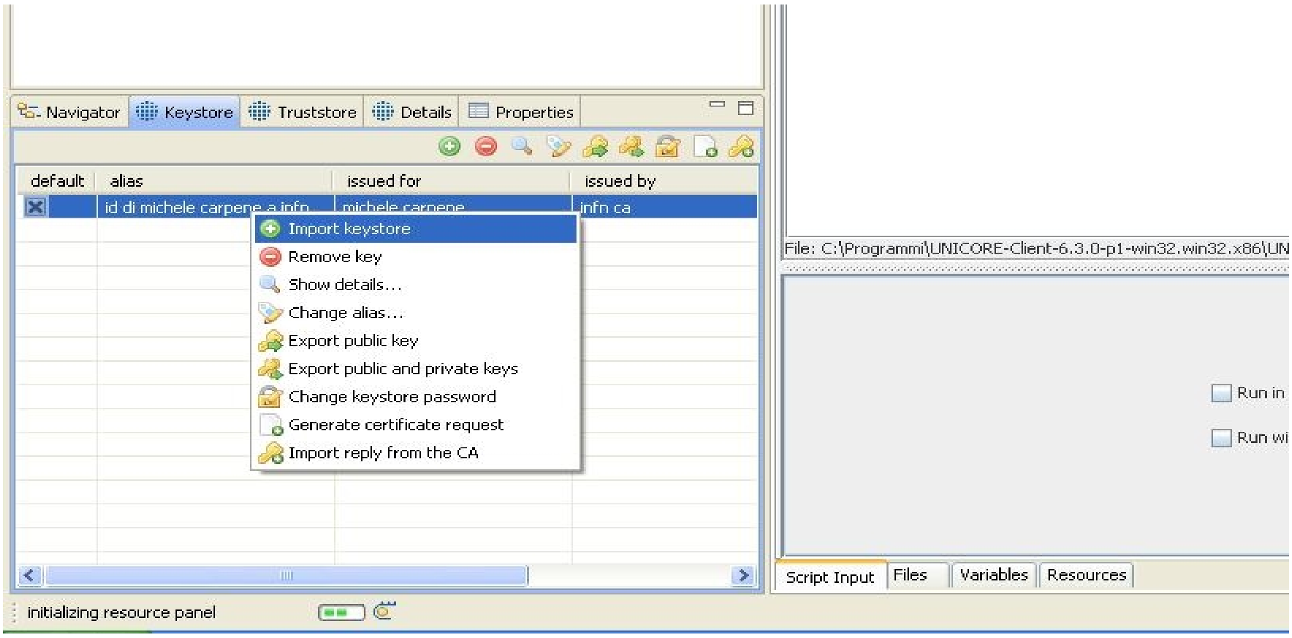
```
-bash-3.2$
-bash-3.2$ grid-proxy-init
Your identity: /C=IT/O=INFN/OU=Personal Certificate/L=CINECA-SAP/CN=Giuseppe Fiameni
Enter GRID pass phrase for this identity:
phrase is too short, needs to be at least 4 chars
Enter GRID pass phrase for this identity:
Creating proxy ......................................... Done
```

4.2.2 UNICORE6 [8]

UNICORE6 (a very nice graphical interface for job submission) can use your p12 certificate. You only need to right click on the **Keystore** window in the UNICORE6 GUI (also known as Rich Client) [8], select import and locate your cert in the popped-up file-manager application[2].

[1] If your client installation is not provided with a list of trusted CA, you should download it: http://winnetou.sara.nl/deisa/certs/globuscerts.tar.gz

[2] If your client installation is not provided with a list of trusted CA, you should download if: http://winnetou.sara.nl/deisa/certs/keystore.jks and import it in the **Truststore** window.



default	alias	issued for	issued by
☒	id di michele carpene a infn	michele carpene	infn ca

File: C:\Programmi\UNICORE-Client-6.3.0-p1-win32.win32.x86\UN

initializing resource panel

5 Advanced tools to manage certificates

As stated above, it is often useful to have your certificate converted in the pem format. A pem certificate is usually made of two files: the certificate (*usercert.pem*) and the private key (*userkey.pem*). The private key is typically protected with a strong password, called passphrase.

5.1 portecle [9]

Portecle is a user friendly Java GUI application for creating, managing and examining keystores, keys, certificates, certificate requests, certificate revocation lists and more.

It supports the following functionality:

- Create, load, save, and convert keystores.
- Generate DSA and RSA key pair entries with self-signed version 1 X.509 certificates.
- Import X.509 certificate files as trusted certificates.
- Import key pairs from PKCS #12 and PEM bundle files.
- Clone and change the password of key pair entries and keystores.
- View the details of certificates contained within keystore entries, certificate files, and SSL/TLS connections.
- Export keystore entries in a variety of formats.
- Generate and view certification requests (CSRs).

- Import Certificate Authority (CA) replies.
- Change the password of key pair entries and keystores.
- Delete, clone, and rename keystore entries.
- View the details of certificate revocation list (CRL) files.

Operations specific to a keystore entry can be accessed by right-clicking on the particular entry in the table and selecting the required operation from the resultant pop-up menu. The options available in the pop-up menu differ depending on the keystore entry type. For example Trusted Certificate entries can be examined, deleted or renamed. Key Pair entries can additionally have their passwords set, be used to generate CSRs (Certificate signing request), etc. Key entries can only be deleted at the time of writing.

5.2 Mozilla Key Manager [10]

This is a plug-in for the Mozilla Foundation browser, Firefox: it gives you an easy to use GUI all inside your internet environment!

6 Bibliography

[1]http://www.cs.auckland.ac.nz/~pgut001/pubs/x509guide.txt

[2]http://security.fi.infn.it/CA/

[3]http://www.cacert.org/

[4]http://www.eugridpma.org/

[5]http://work.deisa.eu/pub/index.shtml

[6]http://www.prace-project.eu/

[7]https://hpc.cineca.it/sites/default/files/CINECA-gridftp-guide.pdf

[8]http://www.unicore.eu/download/unicore6

[9]http://portecle.sourceforge.net/

[10] https://addons.mozilla.org/en-us/firefox/addon/key-manager/

[11] http://www.openssl.org/

[12] http://www.slproweb.com/products/Win32OpenSSL.html

[13] http://www.slproweb.com/download/Win32OpenSSL_Light-0_9_8r.exe

[14] http://www.microsoft.com/downloads/en/confirmation.aspx?familyid=9B2DA534-3E03-4391-8A4D-074B9F2BC1BF&displaylang=en

7 List of keywords/abbreviations

- **der** (Distinguished Encoding Rules) extension: the associated file is a Base64-encoded certificates, usually in binary DER form.
- **pem** (Privacy Enhanced Mail) extension: the associated files are Base64 encoded DER files: the certificate, enclosed between “-----BEGIN CERTIFICATE-----” and “-----END CERTIFICATE-----”, and the key, enclosed between “-----BEGIN RSA PRIVATE KEY----- ” and “-----END RSA PRIVATE KEY----- ”.
- **p12** (one of the family of standards called Public-Key Cryptography Standards) extension: the associated file may contain certificate(s) (public) and private keys (password protected)
- **CA** (Certification Autority): an entity that issues digital certificates.
- **RA** (Registration Authority): an assurer which binds a physical person to a certificate.
- **CSR** (Certificate signing request): a message sent from an applicant to a certificate authority in order to apply for a digital identity certificate.
- **CRL** (certificate revocation list): it is a list of certificates (or, more specifically, a list of serial numbers for certificates) that have been revoked, and therefore should not be relied upon.