Introduction to Programming and Computing for Scientists

Oxana Smirnova

Lund University

Tutorial 7b: Grid certificates and jobs

Update of 2014-12-19

- Turns out that LU does not issue certificates to the students
  - Still unclear whether it is due to policies or a technical issue
- Backup solution: we will use pre-installed certificates on the Iridium cluster
  - We issued the certificates ourselves, so they are not good for any real purpose: we are not a trusted Certificate Authority
- Catch: you will have to do everything on a remote machine (Iridium)
  - No graphical desktop, you'll have to type everything!
  - Actually, this is how people work in Cloud Virtual Machines

- Log in to Iridium:

```
ssh -X progcourseN@pptest-iridium.lunarc.lu.se
```

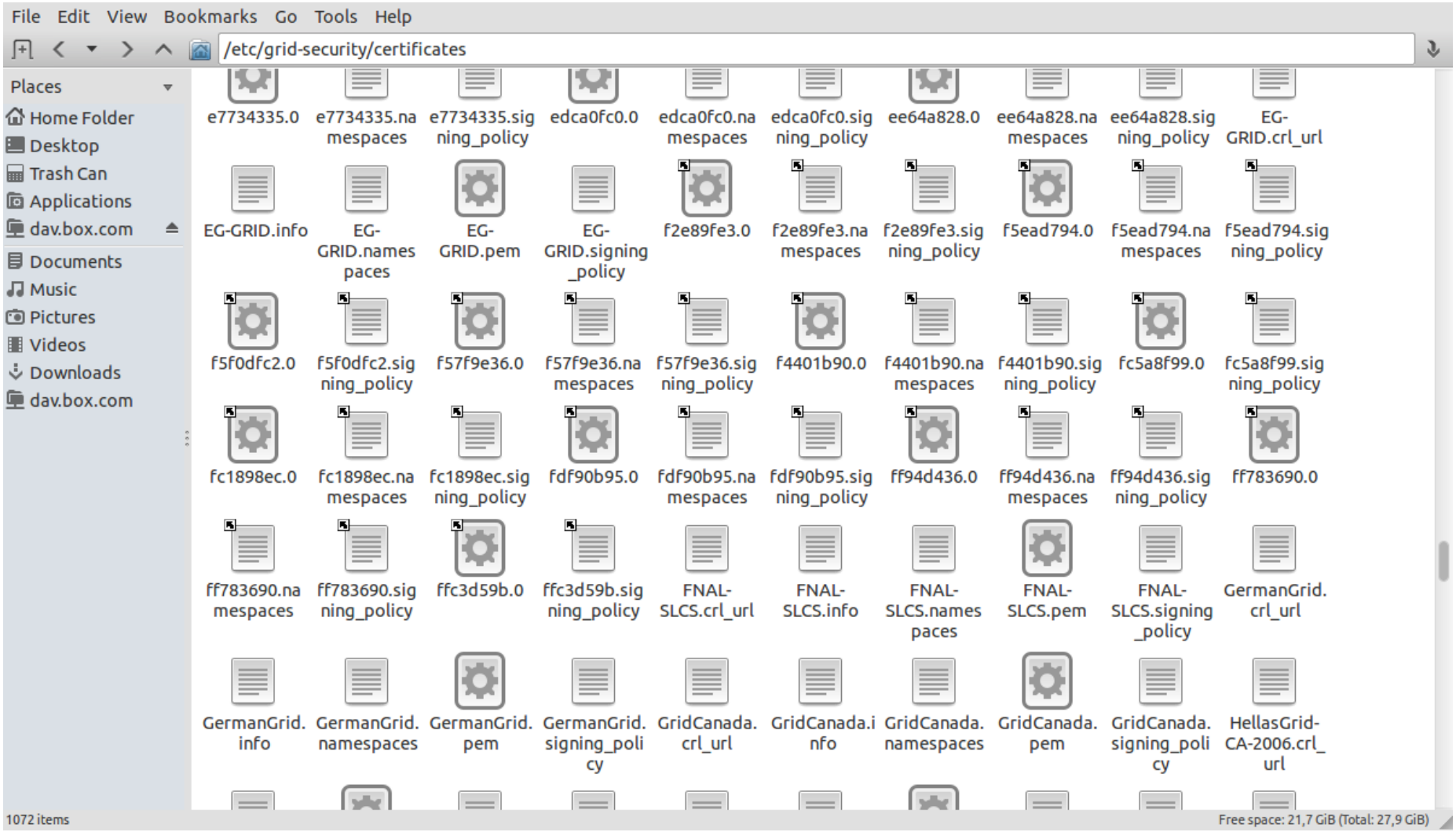
- Some useful commands – a reminder:

```
ls -al
mkdir something
cd something
cp ~/dir/file1 file2
geany &
firefox &
```

Step 1: Install public Certificate Authority certificates

- Before doing anything on the Grid, you will need to obtain **IGTF Certificate Authorities certificates**
  - Packages are available from IGTF and some Grid repositories
    - The packages include Certificate Revocation Lists (CRLs)
  - Regular updates for CRL and IGTF packages must be in place
    - Usually happens automagically
  - **Our computer has them already**
    - Inspect `/etc/grid-security/certificates`
      - Hint: use `ls -al`

/etc/grid-security/certificates in the course Virtual Machine



Step 2: get your own keys and certificates

- Private key and public certificate
  - Grid uses PEM encoding for keys and certificates (ASCII)
  - Standard file names: **`userkey.pem`** and **`usercert.pem`**
    - Note: public key is inside the CA-signed certificate **`usercert.pem`**
- PKCS#12 formatted certificate containing private and public keys, as well as CA signature and CRL info
  - PKCS#12 certificate (**`.p12`**) is used mostly by browsers, but can also replace PEM files in some Grid tools
  - One can create PKCS#12 from PEM files
  - One can also extract PEM files from PKCS#12

Work with the public and private keys: browser gymnastics

- Create a hidden directory **`~/.globus`**
  - This is the default location for Grid certificates
- Find the directory called **`certs`** in your home: it will contain two keys:
  `userkey-progcourseN.pem`
  `usercert-progcourseN.pem`
- Copy these files to **`userkey.pem`** and **`usercert.pem`** in **`~/.globus`**
  - Check that **`userkey.pem`** is readable only by you!
    - Hint: use **`ls -al`**
- Use **`openssl`** command to create a **`.p12`** certificate (one line):

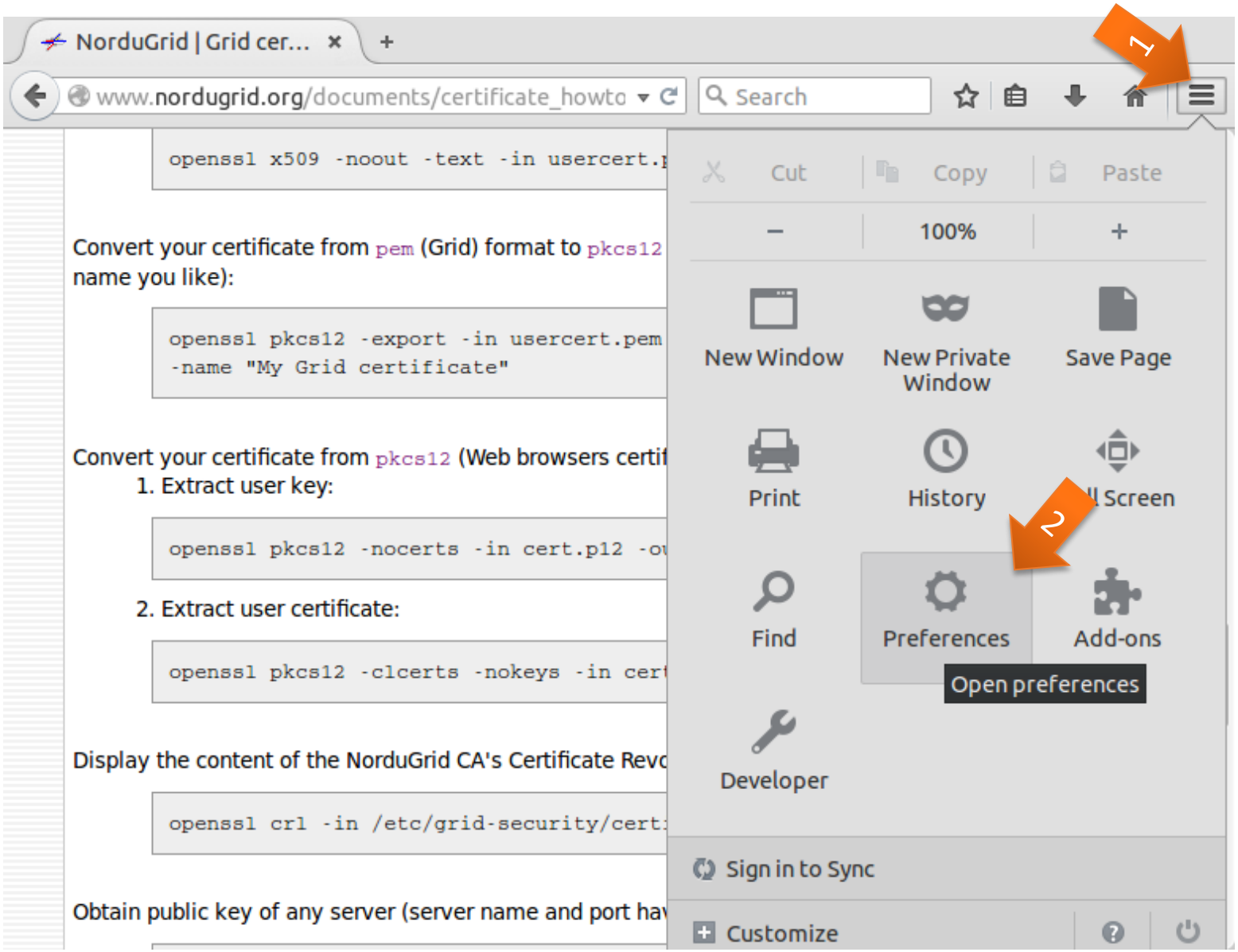
```
openssl pkcs12 -export -in usercert.pem -inkey
userkey.pem -out cert.p12 -name "My toy certificate"
```

  - Hint: Google for "grid certificate howto" to find where to copy-and-paste from
  - "*pass phrase*" for PEM key is the same as for SVN
  - "*Export password*" is the one you will use in the browser
    - You can use the same password in both cases

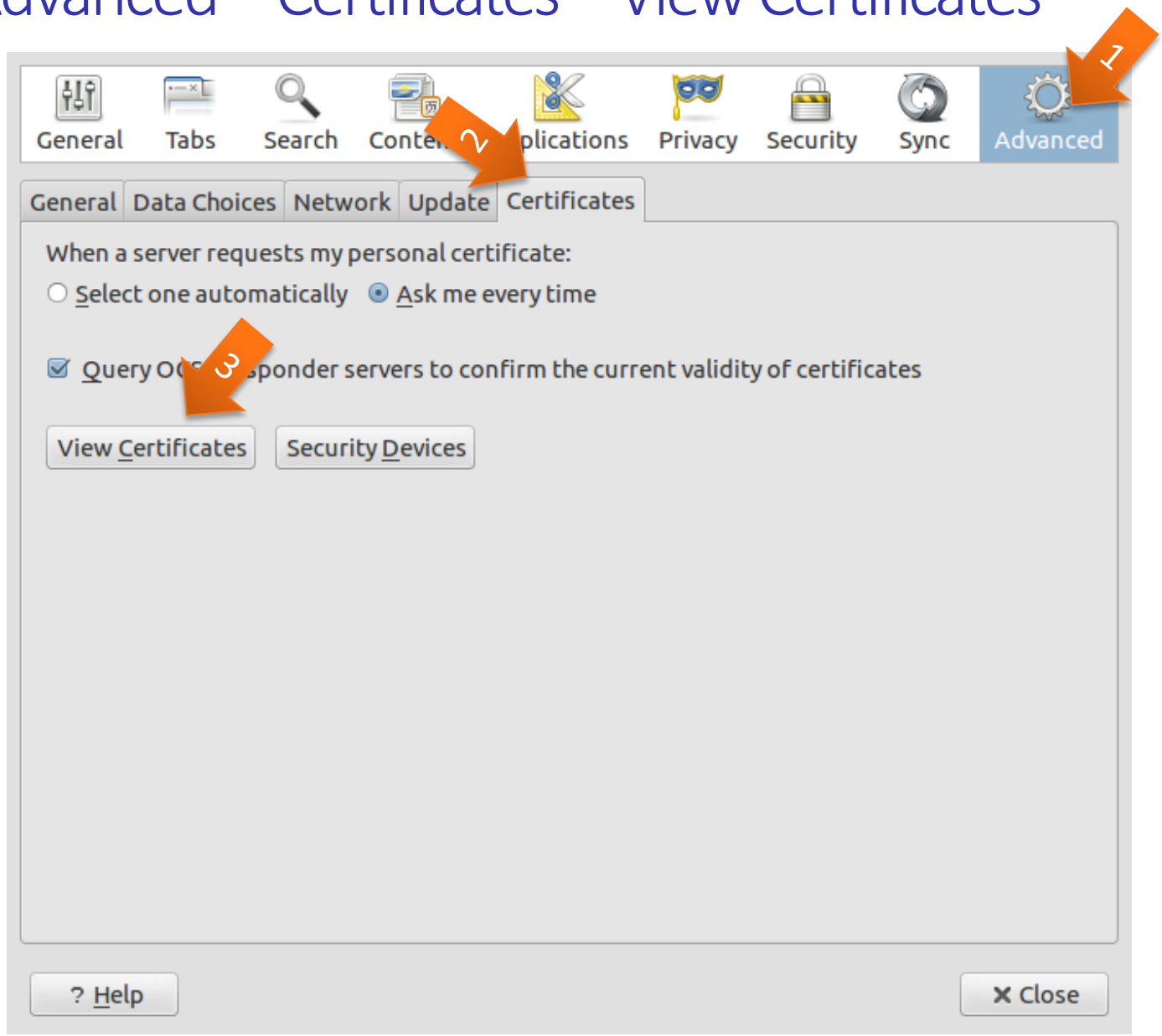
Summary of the steps:

```
File  Edit  Tabs  Help
courseuser@Lubuntu-VirtualBox:~$ mkdir .globus
courseuser@Lubuntu-VirtualBox:~$ cd .globus
courseuser@Lubuntu-VirtualBox:~/.globus$ cp ~/certs/usercert-progcourse7.pem usercert.pem
courseuser@Lubuntu-VirtualBox:~/.globus$ cp ~/certs/userkey-progcourse7.pem userkey.pem
courseuser@Lubuntu-VirtualBox:~/.globus$ ls -al
total 16
drwxrwxr-x  2 courseuser courseuser 4096 dec 18 21:44 .
drwxr-xr-x 24 courseuser courseuser 4096 dec 18 21:43 ..
-rw-r--r--  1 courseuser courseuser 2085 dec 18 21:44 usercert.pem
-r--------  1 courseuser courseuser 2022 dec 18 21:44 userkey.pem
courseuser@Lubuntu-VirtualBox:~/.globus$ openssl pkcs12 -export -in usercert.pem -inkey userkey
.pem -out cert.p12 -name "My toy certificate"
Enter pass phrase for userkey.pem:
Enter Export Password:
Verifying - Enter Export Password:
courseuser@Lubuntu-VirtualBox:~/.globus$ ls -al
total 20
drwxrwxr-x  2 courseuser courseuser 4096 dec 18 21:45 .
drwxr-xr-x 24 courseuser courseuser 4096 dec 18 21:43 ..
-rw-rw-r--  1 courseuser courseuser 2986 dec 18 21:45 cert.p12
-rw-r--r--  1 courseuser courseuser 2085 dec 18 21:44 usercert.pem
-r--------  1 courseuser courseuser 2022 dec 18 21:44 userkey.pem
courseuser@Lubuntu-VirtualBox:~/.globus$
```

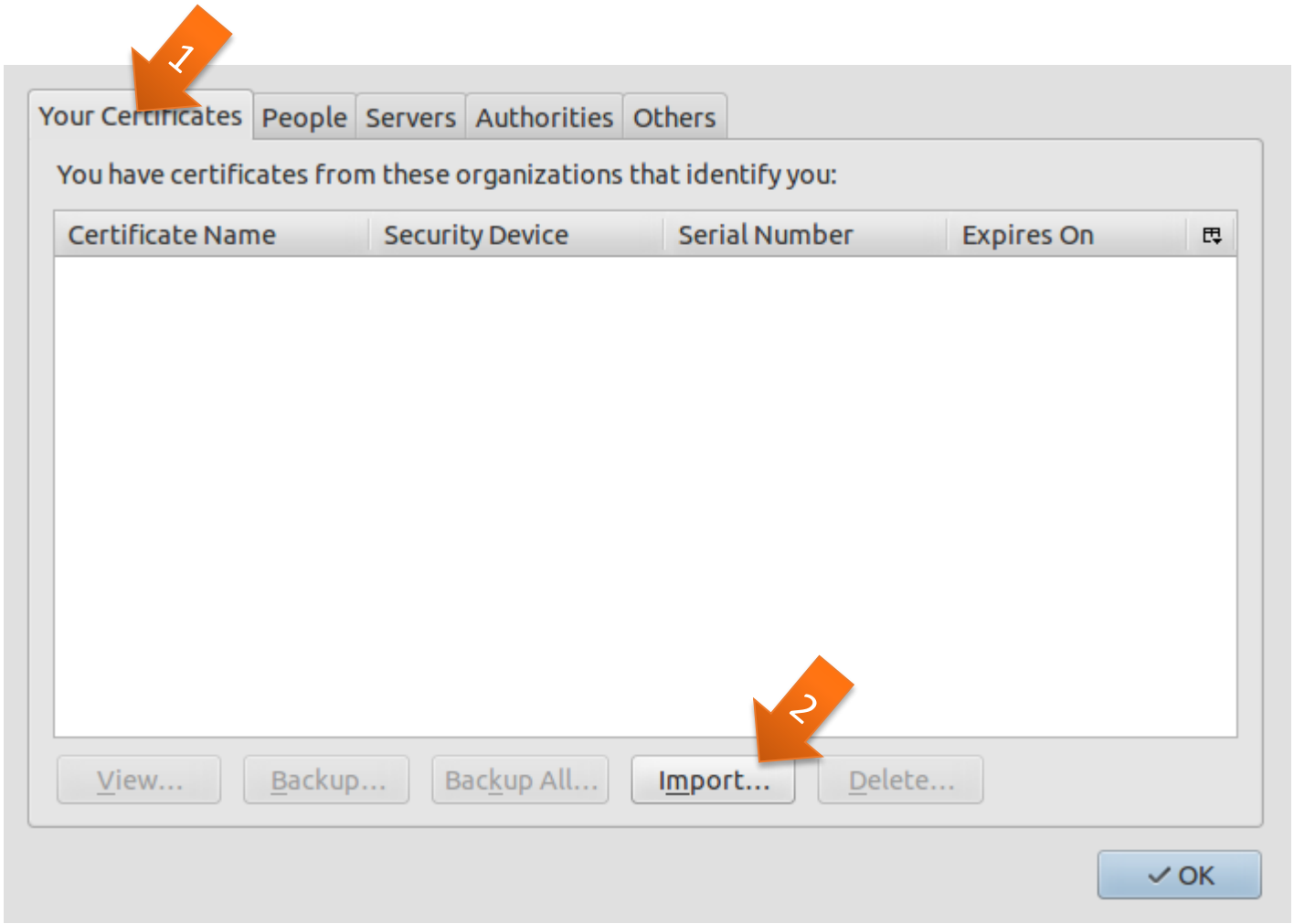
Load the certificate into the browser (start firefox)



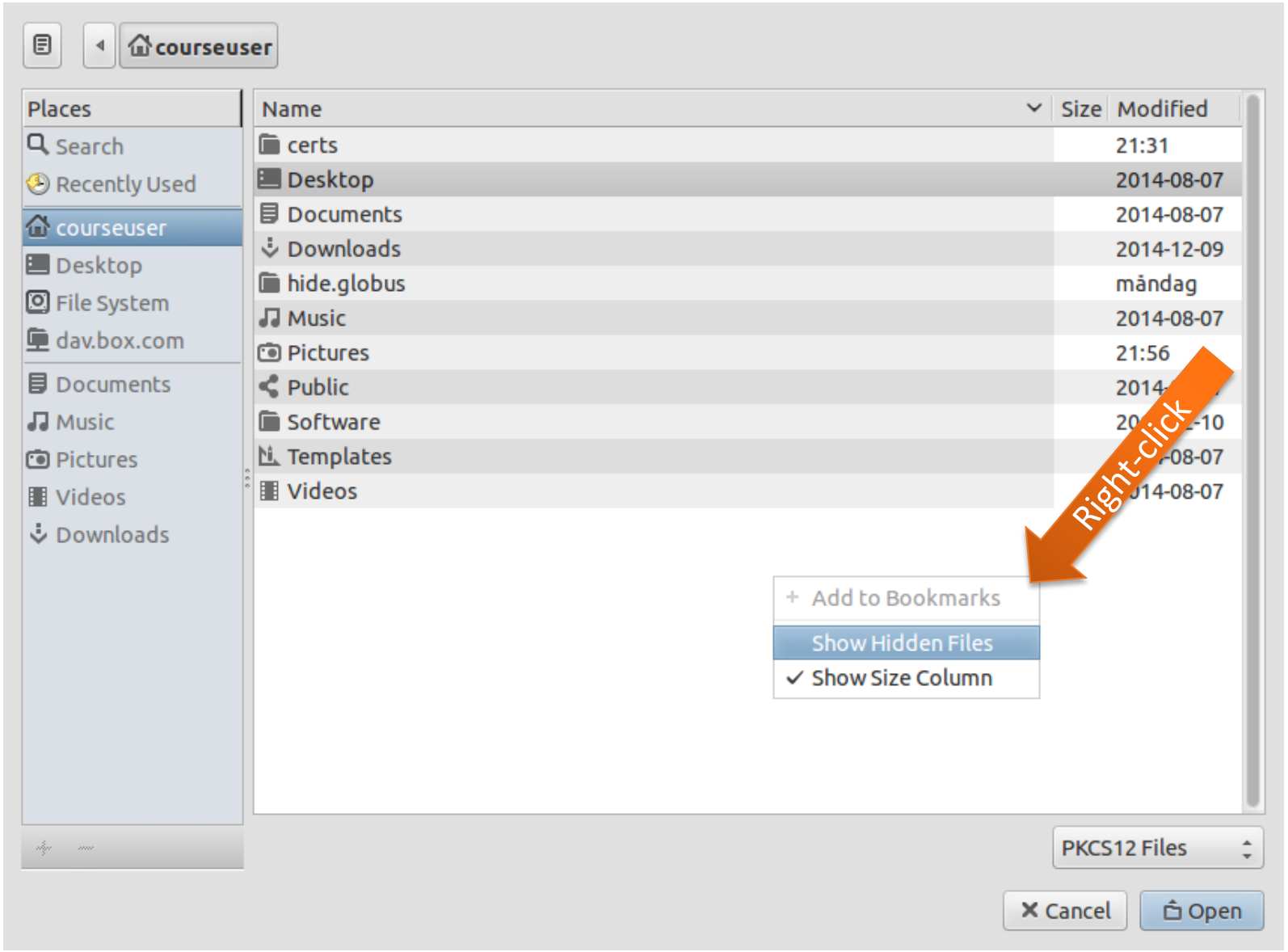
Go to Advanced – Certificates – View Certificates



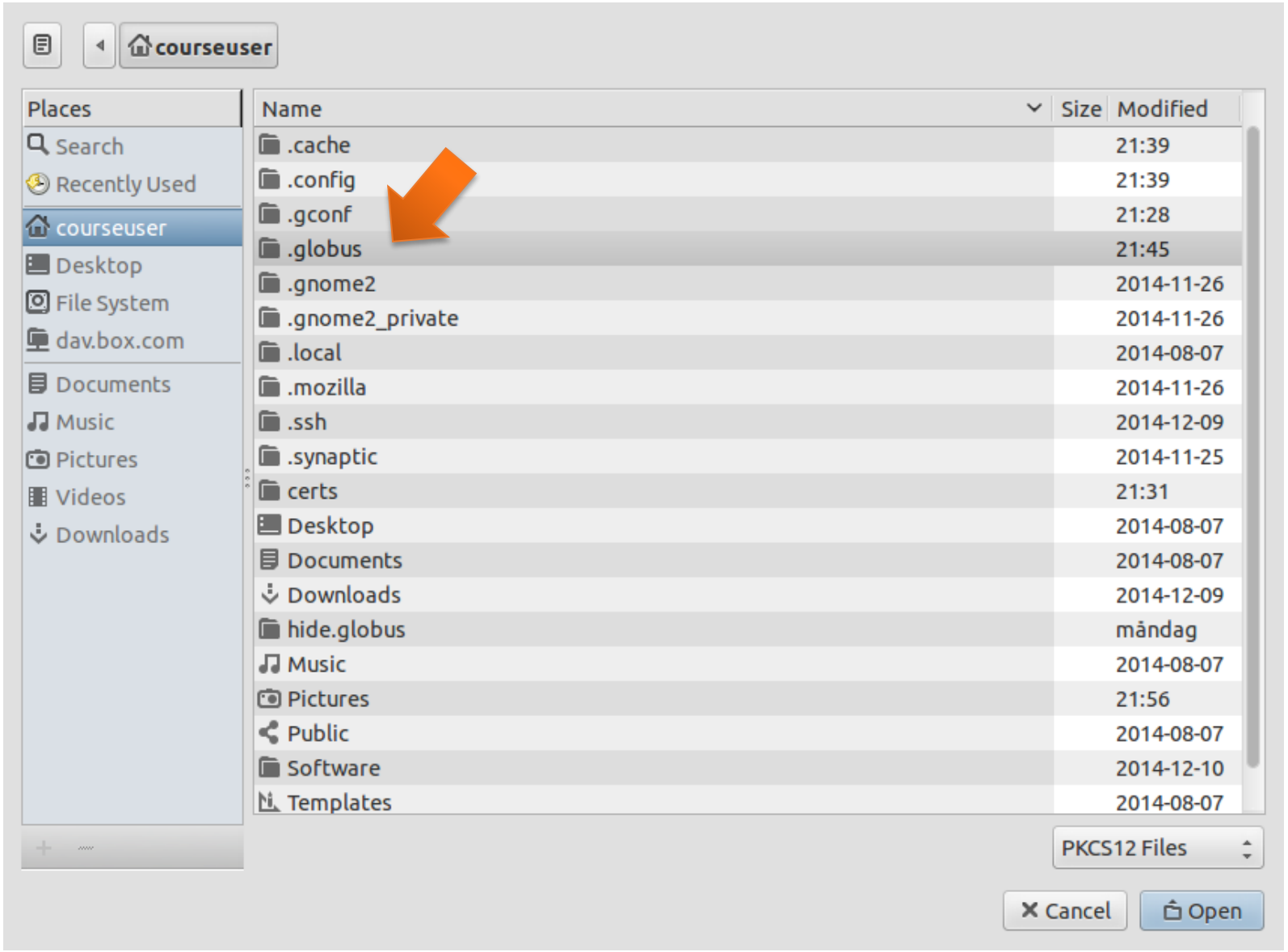
In Your Certificates, find yours, and use Import to load one



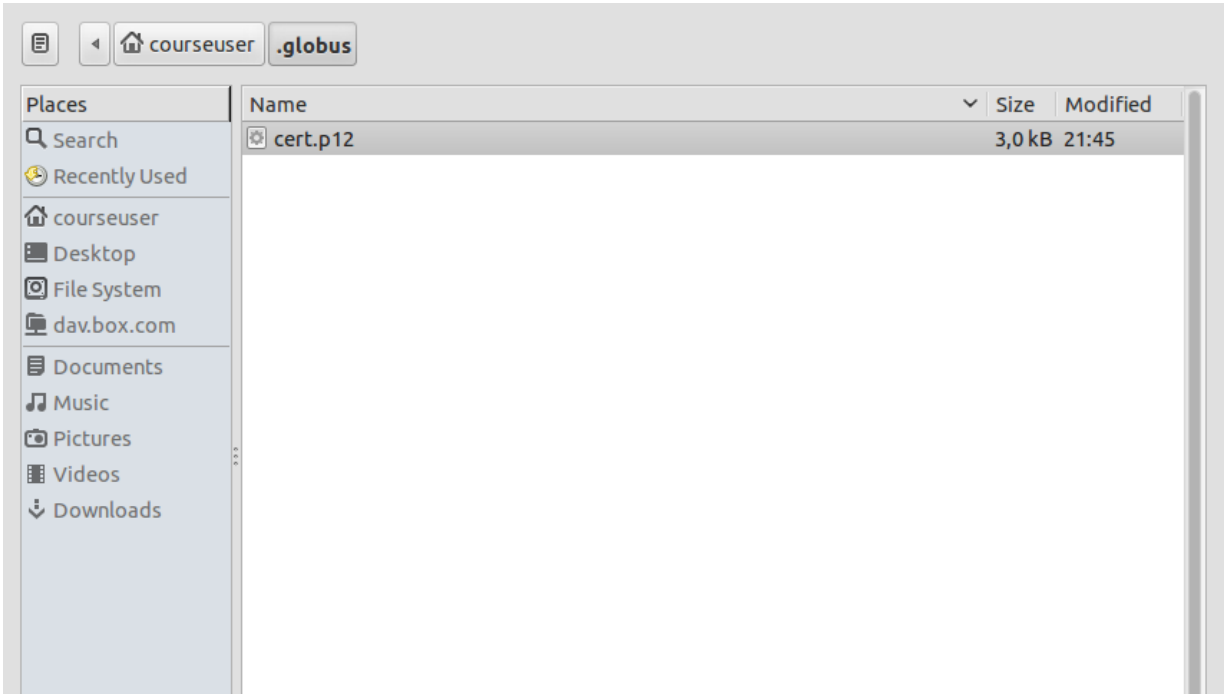
Make sure you can see hidden files: right-click and tick

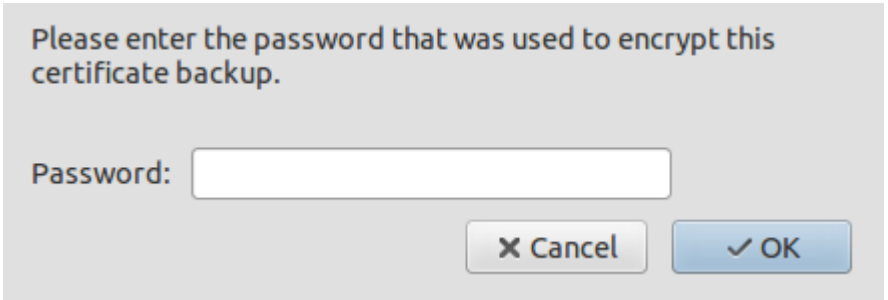


Browse down to .globus

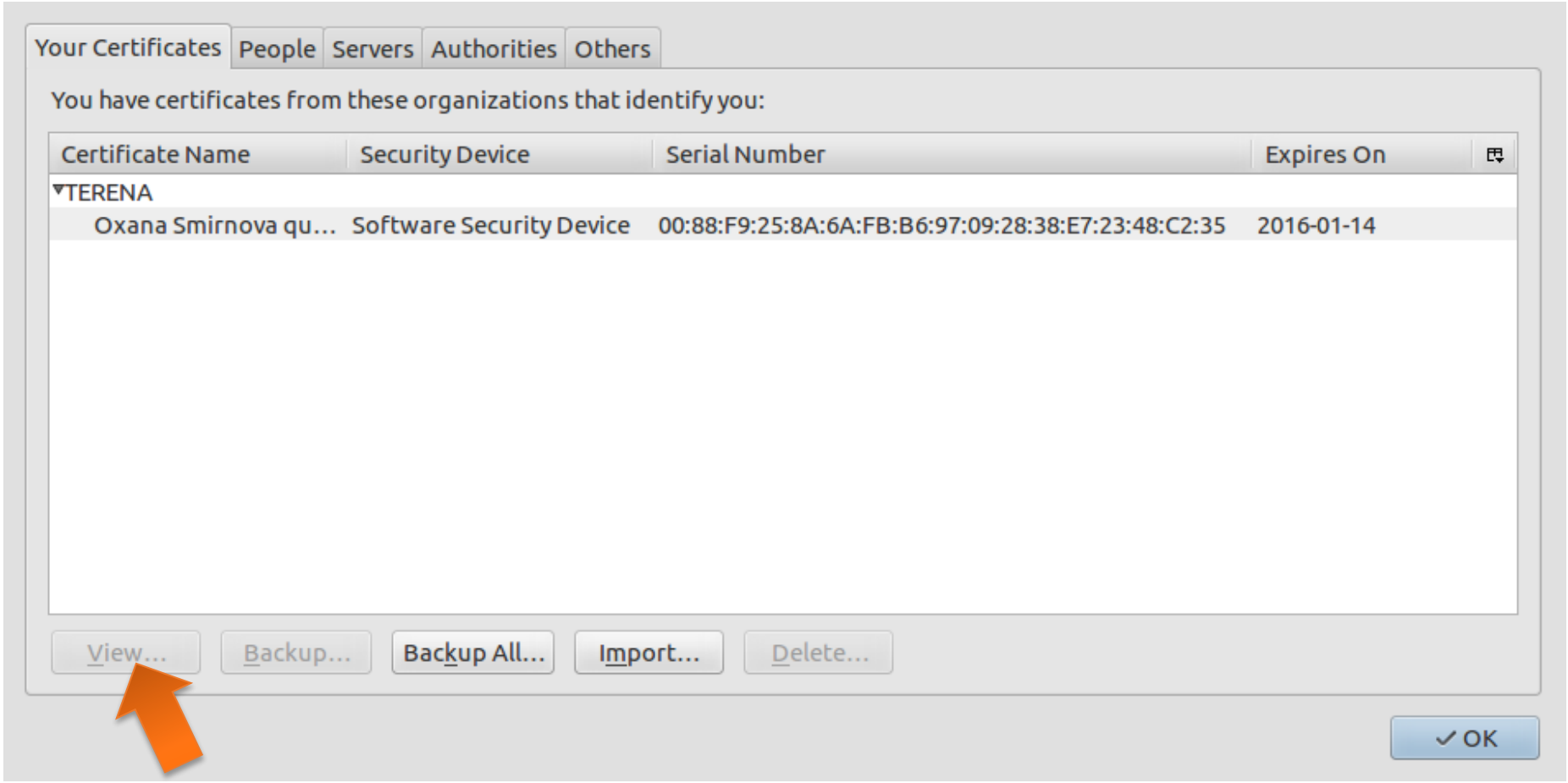


Select the .p12 file, use the password you used to create it





You should now be able to view your certificate



Take some time to view the certificate content

On to Grid: create a proxy!

```
File  Edit  Tabs  Help
courseuser@Lubuntu-VirtualBox:~$ arcproxy
Enter pass phrase for private key:
Your identity: /DC=org/DC=terena/DC=tcs/C=SE/O=Lunds Universitet/CN=Oxana Smirnova quar-osm@lu.se
Proxy generation succeeded
Your proxy is valid until: 2014-12-15 12:32:44
courseuser@Lubuntu-VirtualBox:~$
```

- Simply type **`arcproxy`** and enter your Grid password (*PEM pass phrase* for the private key)

What actually `arcproxy` does?

- A **new** private/public key pair is created for each proxy
  - When a proxy expires, a new one must be created to continue working
    - Default expiration time is 24 hours
- A proxy is then constructed of:
  1. Public certificate (with public key embedded)
     - Certificate contains modified owner's Distinguished Name (has "*proxy*" appended to the name)
       - Owner's DN: **`/C=UK/O=Grid/OU=CenterA/L=LabX/CN=john doe`**
       - Proxy DN: **`/C=UK/O=Grid/OU=CenterA/L=LabX/CN=john doe/CN=proxy`**
     - Certificate is signed by the proxy owner's **real** private key
     - Certificate contains validity period
  2. Private key
  3. Optionally, Attribute Certificates – extensions containing additional information

The tale of two proxies

- A user always has to create a proxy certificate **P1**
  - Technically, it can be sent to the server, but it is a security breach
- Any Grid server (e.g. a Computing Element) creates itself a **delegated proxy P2** for each user request:
  1. Server generates a **new** private/public key pair (yes, that's a 3rd one...)
  2. Server returns the generated public key as a certificate request to the user
  3. User's tool signs that public key and inserts user information (DN etc), thus generating a public certificate. It uses the private key of proxy **P1** for performing signing operation.
     - It can also use the actual private key, but that will require entering password every time!
  4. User's tool sends the signed public certificate back to the server
  5. Server adds generated private key to that certificate and creates a delegated proxy **P2**

What's the use of VOMS

- A Grid user must become a member of a VO
  - VOMS is the most common VO management system
- A Grid cluster administrator gets the list of authorised users from the VOMS database
- VOMS can add extra VO information to your proxy, if necessary
  - For example, your VO role, group etc
  - You should use **`arcproxy`** with special command-line options to request such extra information to be added
- In today's exercise, we won't use it because VOMS servers do not trust our toy certificates

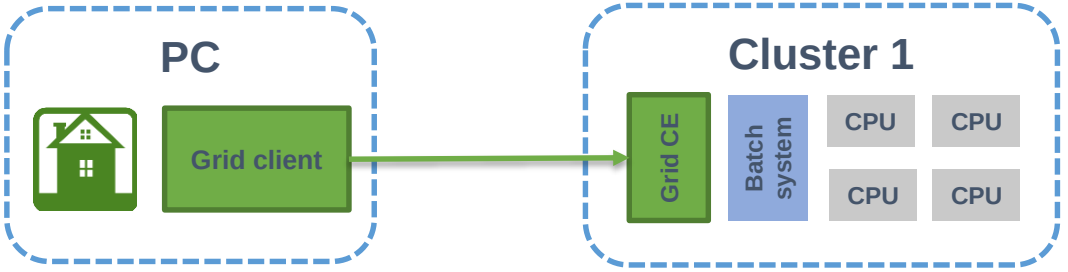
Summary of the proxies

- Luckily, all authentication and delegation procedures are a part of the protocol, you only need to create a proxy
- You have to create a proxy before every Grid activity
- Proxies expire quickly!
  - Resist temptation to create long-living proxy: this will undermine your security
- Proxies may have special extensions, specific to Virtual Organisations
- If you forget your Grid password (PEM pass phrase), and even the browser Import Password, you will have to request a new certificate

Workflow: Grid vs PC/cluster

PC/cluster		Grid	
Log in via SSH	• Different logins on different machines	Create proxy	• One for all machines
Familiarize with the environment	• OS, installed software, storage space, batch system, sysadmin etc	Create a Grid job description document	• Generalization of batch scripts, plus input/output data location etc
Customize the environment	• Pathes, environment variables, own software, scripts, data files etc	Test a couple of jobs, fix job description	
Prepare for batch submission	• Interactive execution of short jobs, optimization of batch scripts	Submit jobs to the Grid, check their status	• Same commands for all machines
Submit jobs to the batch system, check their status	• Different batch systems (or none) on different machines	Watch output appearing in the desired location	• Or fetch it manually
Log out			
Log in later to fetch the output			

Simplest Grid job submission



- Your Grid client should:
  - Create a proxy:
    - **`arcproxy`**
  - Submit the job description document to the cluster:
    - **`arcsub -c arc-iridium.lunarc.lu.se hello_grid.xrsl`**
      - **`arcsub`** will refuse submission if the cluster does not meet job requirements

See the ARC Clients manual for info about all ARC client commands:
http://www.nordugrid.org/documents/arc-ui.pdf

- The CE on the cluster should:
  - Check whether you are authorised
  - Fetch input file (if requested)
  - Convert job description to a batch script and start a batch job
  - Upload output file (if requested)

Simplest Grid job description: hello_grid.xrsl

```
&( executable = "/bin/echo" )
 ( arguments = "hello grid" )
 ( stdout = "stdout_file" )
 ( stderr = "error_file" )
 ( cputime = "13" )
 ( gmlog = "grid_log" )
 ( jobname = "hello_grid" )
```

attribute

value

- Yes, this is yet another language:
  XRSL – eXtended Resource Specification Language
  - File extension is **`.xrsl`**
- XRSL is not a standard language, but no standard exists
  - There are many other Grid languages and meta-languages
  - XRSL is an ARC extension of the original Grid language by Globus
    - It was actually modelled on the LDAP database query language
    - Is a list of attribute-value pairs

Main attributes of job description

Job attribute description	Attribute name (XRSL)	Example value
Main executable (binary or script)	**executable**	MyAnalysis.py
Arguments of the executable	**arguments**	-i input.dat -o output.dat
Input files	**inputfiles**	https://store.lu.se/physlab/2012/file1.dat
Output files	**outputfiles**	https://store.lu.se/physlab/2014/file1.dat
Standard input file	**stdin**	stdin.txt
Standard output file	**stdout**	stdout.txt
Standard error file	**stderr**	stderr.txt
Time (used by CPU)	**cputime**	1 hour
Memory (maximum needed, Mbytes)	**memory**	1000
Disk space (maximum needed, Mbytes)	**disk**	1000
Job name	**jobname**	My data analysis
Number of slots (cores) for the job	**count**	36
and many others: ARC job description language XRSL has 37 attributes, see http://www.nordugrid.org/documents/xrsl.pdf		

Create and submit your hello_grid.xrsl

- Prepare job description for the “Hello Grid” task:
  - Use Geany to create a file **`hello_grid.xrsl`**
  - Use at least the following XRSL attributes: **`executable`**, **`arguments`**, **`jobname`**
    - Hint: copy the example from the previous slide
- Submit your first Grid job to our Iridium cluster:
  - First, make sure you have a valid proxy:

    ```
    arcproxy -I
    ```

  - Use the **`arcsub`** command with explicit cluster selection:

    ```
    arcsub -c arc-iridium.lunarc.lu.se hello_grid.xrsl
    ```

  - Find the returned **job ID** (a long string that looks like a URL)
  - Check the job’s status:
    `arcstat <jobid>`
  - Check what the job “session directory” looks like on the cluster:
    `arcls <jobid>`
  - Check what does the job print out:
    `arccat <jobid>`

Manipulate the jobs: kill, retrieve

- Submit a couple more jobs
  - You may want to change the job names in **`hello_grid.xrsl`**
  - Or you may even want to change what do the jobs produce
- Check the status of all your jobs:

```
arcstat -a
```

- Terminate some of them and check the status afterwards:

```
arckill -k <jobid>
arcstat <jobid>
```

  - **`-k`** here means “keep the job files”, otherwise they will be wiped out
- Retrieve job results (download job output):

```
arcget -k <jobid>
```

  - **`-k`** here has the same meaning as for **`arckill`**
- Find where the downloaded files are, and look what is there
  - Inspect the content of the **`gmlog`** sub-directory: it has files useful for error diagnostics and debugging

If you have some time left

- Find the hidden directory **`~/.arc`** and file **`client.conf`** therein
- Open **`client.conf`** in Geany (or any other editor)
- Find blocks **`[registry/index1]`** , **`[registry/index2]`** etc and uncomment them and their content
  - Save **`client.conf`** and quit the editor
- Try to submit **`hello_grid.xrsl`** to the entire Grid

```
arcsub hello_grid.xrsl
```