Introduction to the Architecture of the European DataGrid

francesco.giacomini@cnaf.infn.it

A Definition

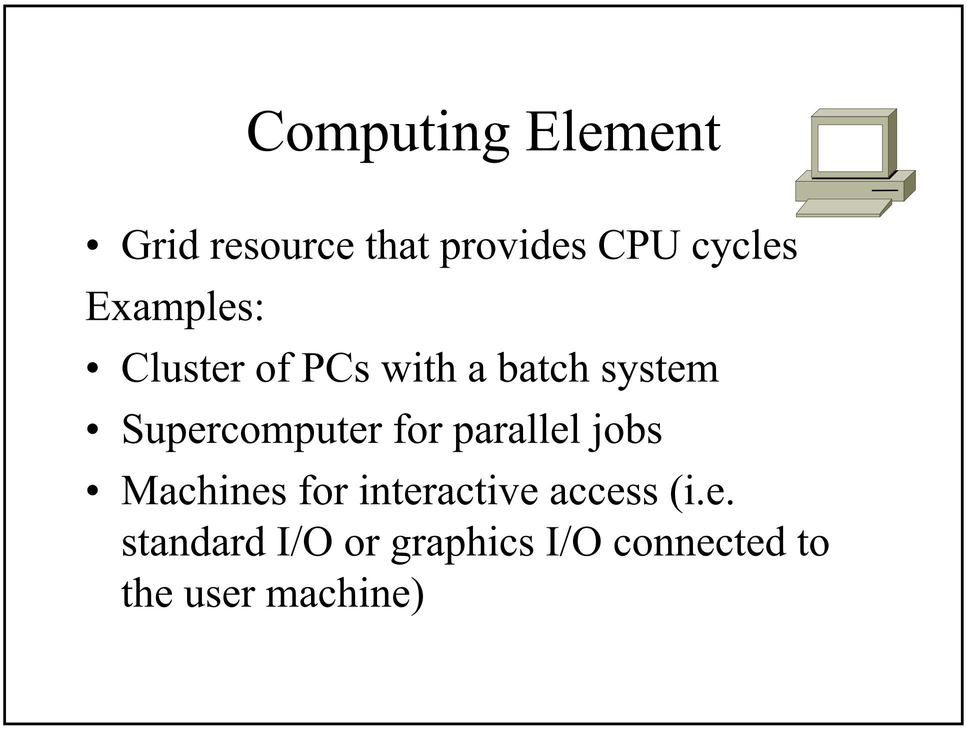
GRID: the sharing and coordinated use of resources within large, dynamic, multi-institutional communities

Example

```
[
  Executable = “cmsim”;
  InputData =
    “lfn://cms.org/pythia_Higgs.ntpl”;
  Requirements =
    member(other.RunTimeEnvironment,
           “CMS-1.0.0”);
]
```

Basic Resources

- Need to execute a program
  ⇒ Computing Element
- Need to access data
  ⇒ Storage Element
- Need to move data
  ⇒ Network

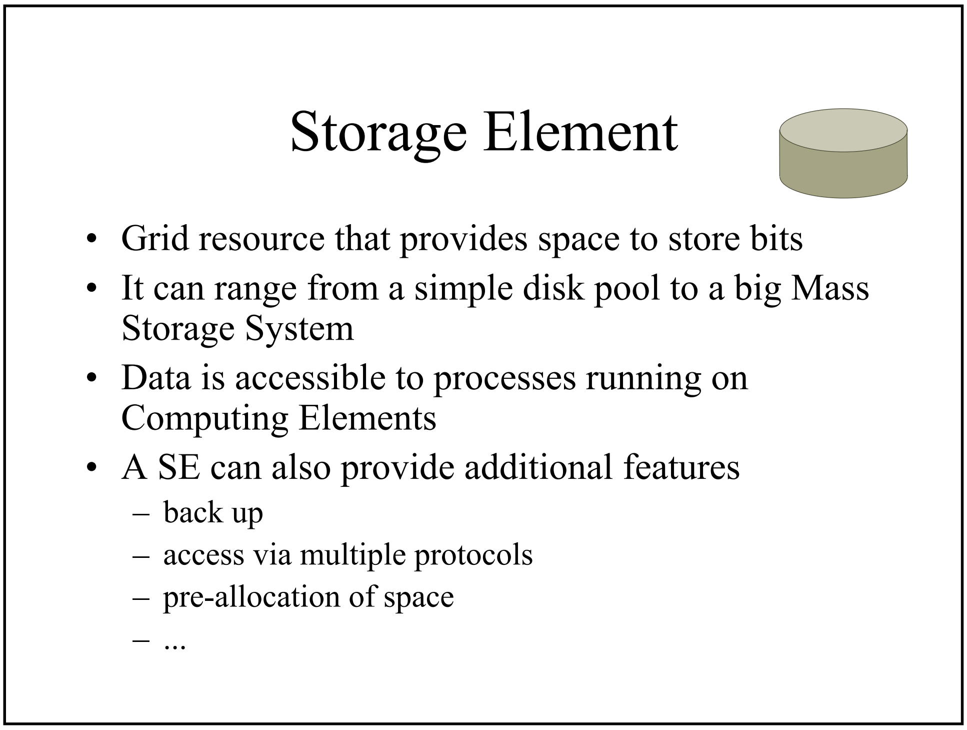


Computing Element

- Grid resource that provides CPU cycles

Examples:

- Cluster of PCs with a batch system
- Supercomputer for parallel jobs
- Machines for interactive access (i.e. standard I/O or graphics I/O connected to the user machine)



Storage Element

- Grid resource that provides space to store bits
- It can range from a simple disk pool to a big Mass Storage System
- Data is accessible to processes running on Computing Elements
- A SE can also provide additional features
  - back up
  - access via multiple protocols
  - pre-allocation of space
  - ...

Network

- The network provides connectivity between other resources and higher level services (see later)
- It can also be seen as a component that supports quality-of-service (QoS) requests.

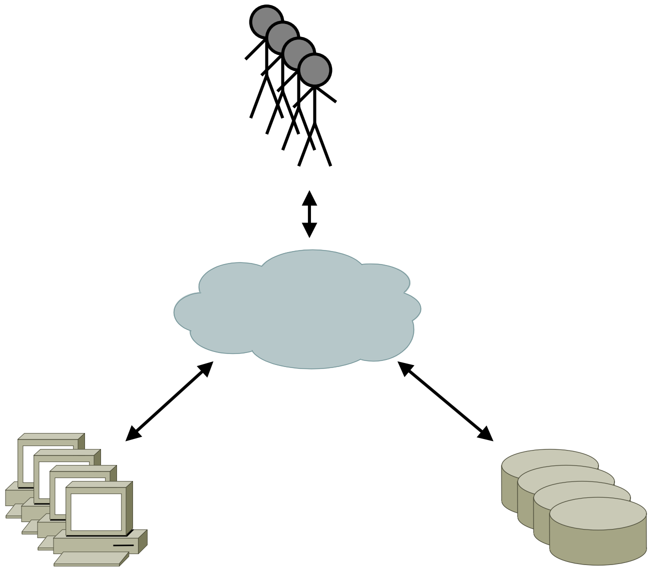
Definition of Grid Resource

- A Grid Resource is a fundamental service
  - it does not depend on others in order to work
  - it has knowledge only of itself
- A Grid Resource provides a **standard interface** (both **protocol** and **API**) that is common to that type of resource
  - all Computing Elements talk the same protocol (the CE protocol), independently of the underlying batch system
  - all Storage Elements talk the same protocol (the SE protocol), independently of the underlying Mass Storage System

Grid and Resources

- A resource is always under the control of the local manager
- The Grid is an infrastructure that leverages the use of distributed resources
  - it does not affect the local management
  - it does not compromise local policies (especially security)
  - Grid activity can coexist with local activity

The Big Picture



What the Grid Offers

- Independence from execution location
  - The user doesn’t want to know where (what CE) a job will run
- Independence from data location
  - The user doesn’t want to know where (what SE) data is
- Security
  - authentication, authorization, confidentiality

Execution Location Independence

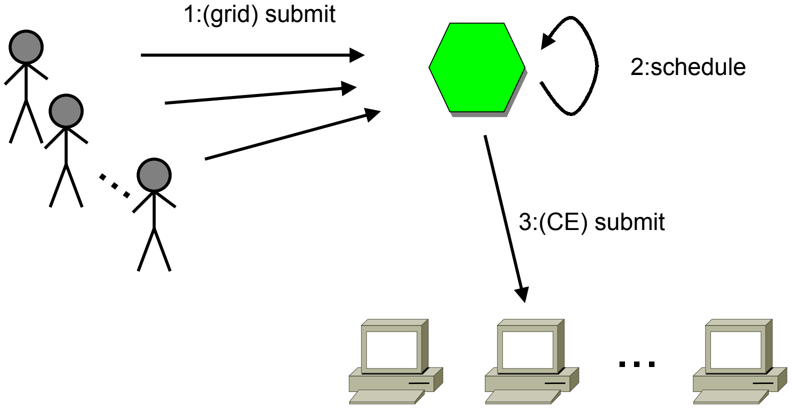
Example

```
[
  Executable = “cmsim”;
  InputData =
    “lfn://cms.org/pythia_Higgs.ntpl”;
  Requirements =
    member(other.RunTimeEnvironment,
           “CMS-1.0.0”);
]
```

Workload Management Service

- A Resource Broker tries to find a good match between the job requirements and preferences and the available resources, in particular CEs
- The Job Submission Service then guarantees a reliable job submission and monitoring

The Big Picture



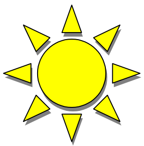
Scheduling Criteria

1. Authorization information
2. Data availability
3. Job requirements
4. Job preferences
5. Accounting
6. Heuristics

Resource Broker

- What are the available resources?
- What is their status?
- In general, multiple RBs see the same resources
  - an RB does not have complete control of the resource

Monitoring/Information System

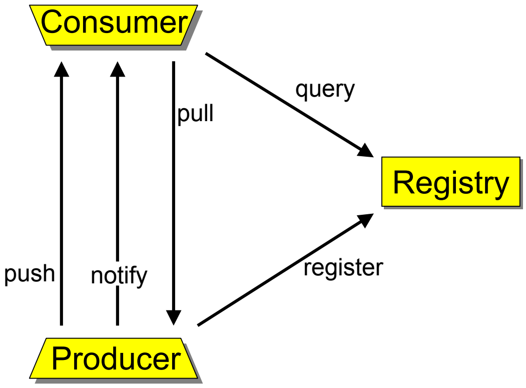


Three types of participants:

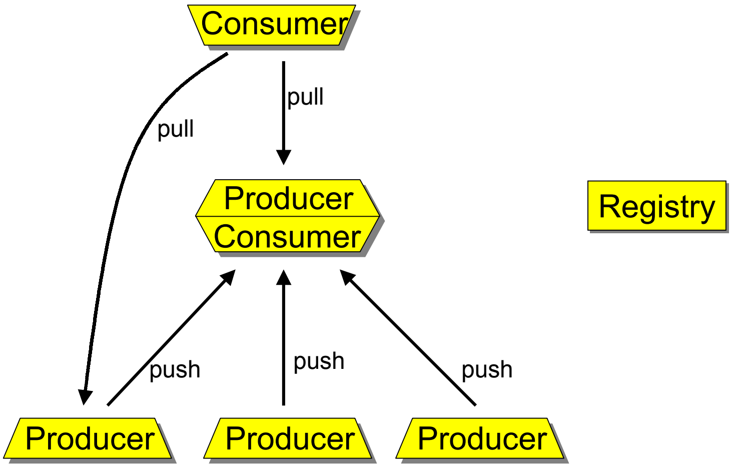
1. Producers make info available to the Grid
2. Consumers make use of such info
3. A registry contains location of producers and the type of info they provide

- Consumers query the registry to locate producers and then obtain data directly from producers

Monitoring/Information System Basic Configuration



Monitoring/Information System Compound Configuration

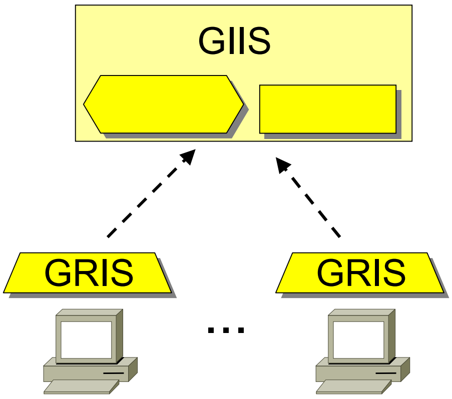


Producers/Consumers

- Example: a Computing Element is a Producer of information, for:
  - number of nodes
  - queue length
  - CPU type
  - OS type, ...
- Example: a Resource Broker is a Consumer of information
  - but it can also be a Producer, e.g. if there are multiple RBs and they publish some info

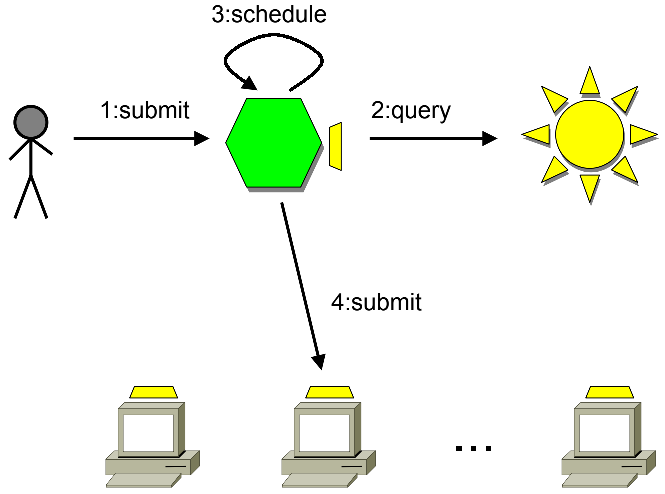
IS in DataGrid

Currently based on a directory service (LDAP)



- Grid Resource Information Service produces info
- Grid Information Index Service keeps a pointer to the registered GRISes and caches info

The Big Picture



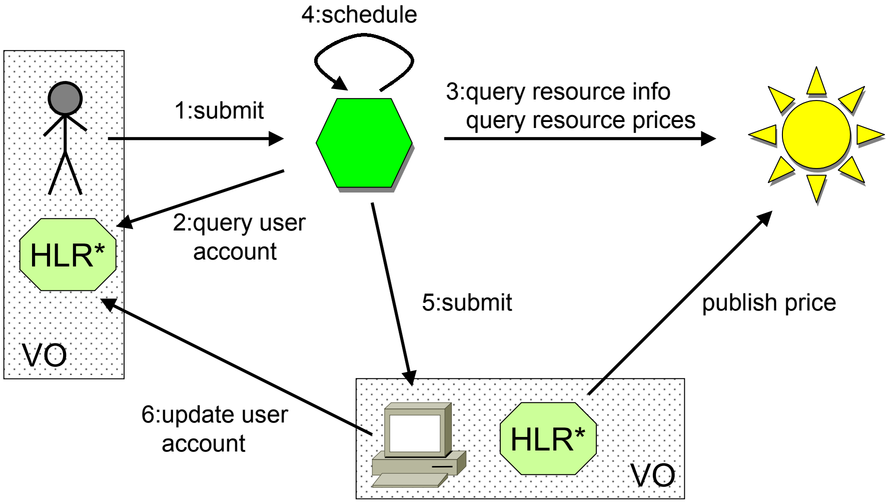
Logging and Bookkeeping

- Special instance of an information system component
- The LB service is a database of events concerning jobs and the other services of the WMS (RB and JSS)
- Provides status info for jobs
- Provides auditing info for RB and JSS
- Designed to be highly reliable and available

Accounting

- Provide an infrastructure to keep track of resource utilization
- For auditing, statistics, management
- But also as support for scheduling
  - economic model: what is the resource that offers the best price/performance ratio?

The Big Picture



* HLR: Home Location Register

Data Location Independence

Example

```
[
  Executable = “cmsim”;
  InputData =
    “lfn://cms.org/pythia_Higgs.ntpl”;
  Requirements =
    member(other.RunTimeEnvironment,
           “CMS-1.0.0”);
]
```

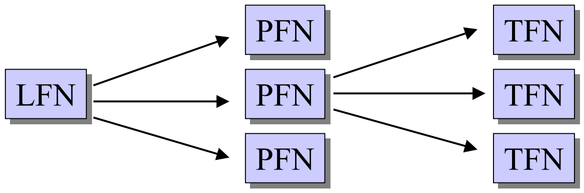
File-based Data System

- Currently the file is the unit of data access
  - an application may need to map objects to files
- The same file (master) can exist in multiple copies (replicas)
  - ✪ efficiency
  - ☆ consistency
  - ☆ management
- No intention to build a distributed file system

Many Names for a File...

- Logical File Name: name for a set of replicas
  - `lfn://<VO>/<path>`
  - `lfn://cms.org/pythia_Higgs.ntpl`
- Physical File Name: location of a replica
  - `pfn://<SE>/<path>`
  - `pfn://se.cern.ch/cms/p_H.ntpl`
- Transport File Name: access name for a replica
  - `<protocol>://<hostname>:<port>/<path>`
  - `gsiftp://se.cern.ch:1234/d1/cms/p_H.ntpl`

LFN to PFN to TFN



LFN

PFN

PFN

PFN

TFN

TFN

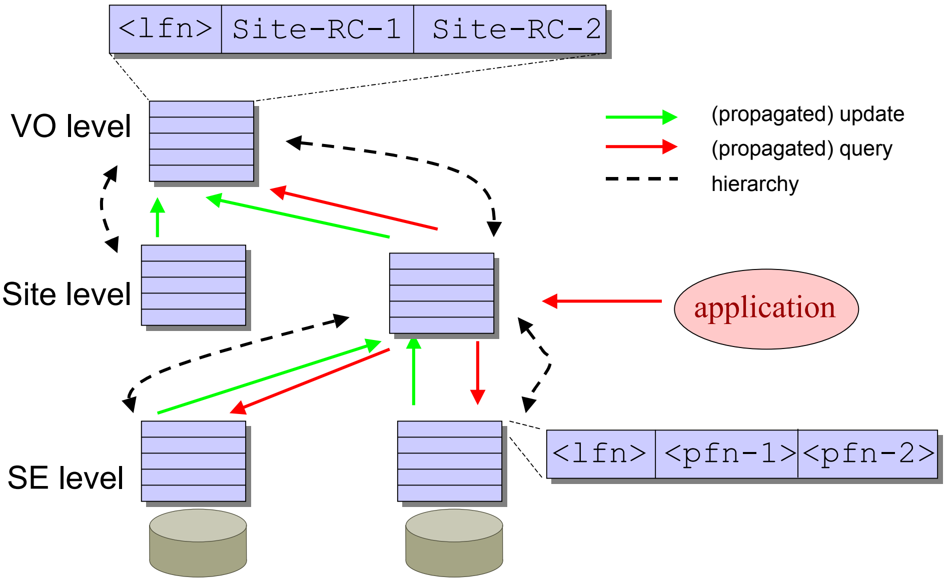
TFN

- LFN to PFN: Replica Catalog
- PFN to TFN: ?

Replica Catalog

- Database that keeps the mapping between LFNs and PFNs
- File attributes for LFNs and PFNs include:
  - master or replica
  - ownership
  - access rights
  - time stamps
  - checksums
  - file type, ...
- Different implementation are possible: centralized, replicated, partitioned, hybrid, ...

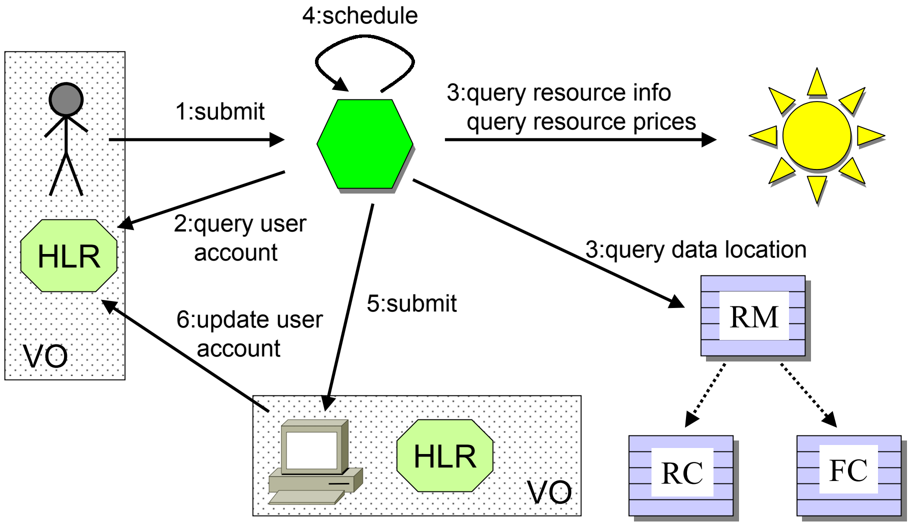
Example of RC Implementation



Data Access

- Local or remote
- If local, a file may need to be moved using a File Copier
- A Replica Manager abstracts the File Copier and the Replica Catalog interfaces and coordinates their actions:
  - copy file
  - update Replica Catalog

The Big Picture



Grid FTP

- Supports GSI (see Security)
- Better performance
  - Use of parallel streams
  - Better tuning of the TCP window
- Allows a partial transfer
- Interrupted transfer recovery
- Supports third party transfers

Security

thanks to Roberto Cecchini
INFN-Firenze

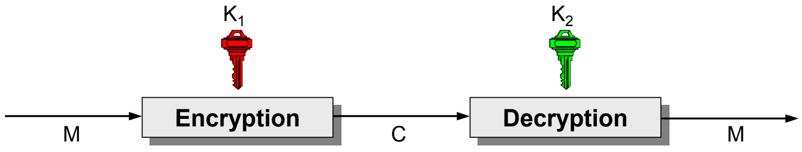
Security Needs

- Authentication
  - prove the identity of an entity (user, process, host, service, ...)
- Confidentiality
  - a third party cannot understand the communication
- Integrity
  - data is not modified during communication
- Non-repudiability
  - the sender cannot claim he didn’t send the data
- Authorization
  - an entity can do only what it is allowed to do

Cryptography

- Mathematical tool that provides some important building blocks for the implementation of a security infrastructure:
  - encryption
    - symmetric key
    - public key
  - one-way hash

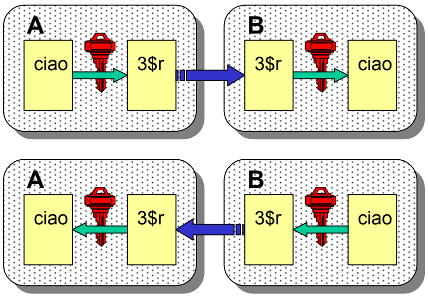
Encryption



- Plaintext: M
- Ciphertext: C
- Encryption with key K_1: $E_{K_1}(M) = C$
- Decryption with key K_2: $D_{K_2}(C) = M$

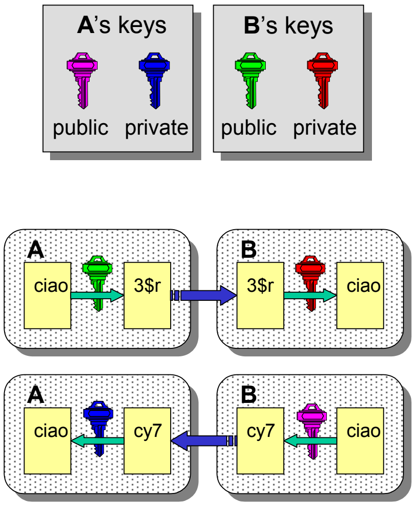
Symmetric Key Encryption

- $K_1 = K_2$
  - encryption and decryption keys are the same
  - the key is a shared secret between A and B
- Examples:
  - DES
  - 3DES
  - Rijndael (AES)
  - Blowfish



Public Key Encryption

- $K_1 \neq K_2$
  - every user has two keys: one private and one public
  - the private key is known only to the user
  - the public key is... public
  - it's practically impossible to derive the private key from the public key
  - based on the difficulty to factorize large numbers in prime factors (example: RSA)



One-Way Hash Functions

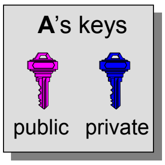
- Function (H) that given as input a variable-length message (M) produce as output a fixed-string (h)
  - H(M) = h
  - the length of h must be at least 128 bits
- Examples:
  - MD5
  - SHA

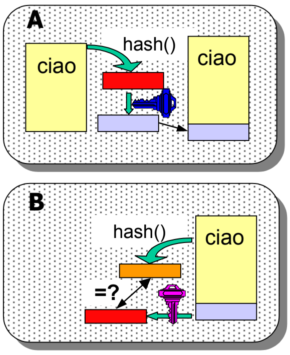
Hash Function Characteristics

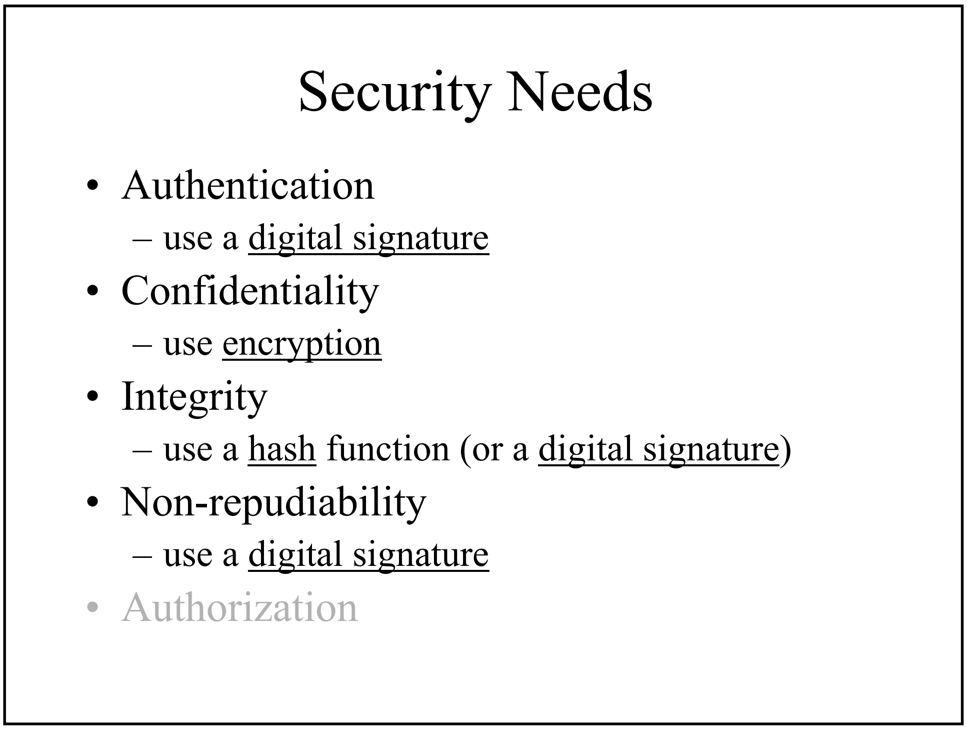
1. given M, it’s easy to calculate $H(M) = h$
2. given h, it’s difficult to calculate $M = H^{-1}(h)$
3. given M, it’s difficult to find M’ such that $H(M) = H(M’)$

Digital Signature

- Combination of a hash function and public key encryption

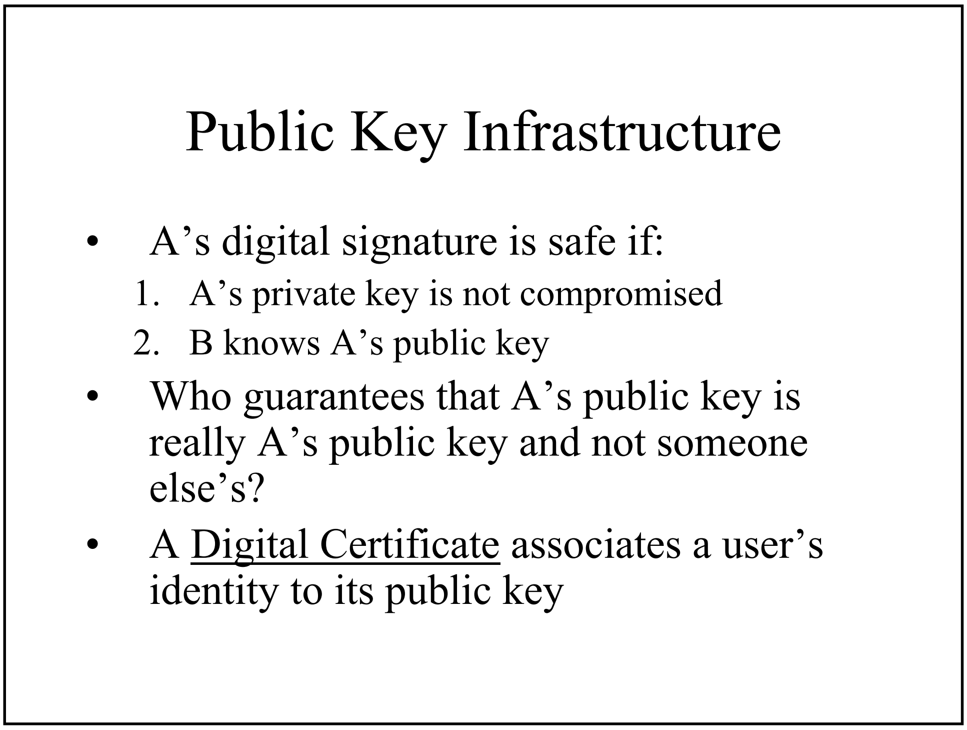






Security Needs

- Authentication
  - use a digital signature
- Confidentiality
  - use encryption
- Integrity
  - use a hash function (or a digital signature)
- Non-repudiability
  - use a digital signature
- Authorization



Public Key Infrastructure

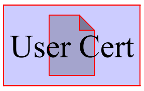
- A’s digital signature is safe if:
  1. A’s private key is not compromised
  2. B knows A’s public key
- Who guarantees that A’s public key is really A’s public key and not someone else’s?
- A Digital Certificate associates a user’s identity to its public key

PKI

- A third party (Certification Authority) guarantees the contents of a certificate are correct
- Both A and B trust the CA
- Two models:
  - X.509: hierarchical organization
  - PGP: “web of trust”
- In Grid the X.509 model is used

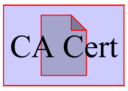
X.509 Certificates

- An X.509 certificate contains:
  - identity of the owner
  - validity time
  - owner’s public key
  - info on the Certification Authority
  - digital signature of the CA
- Certificates are published in a directory (e.g. LDAP or WWW) managed by the CA



Certificate Chain

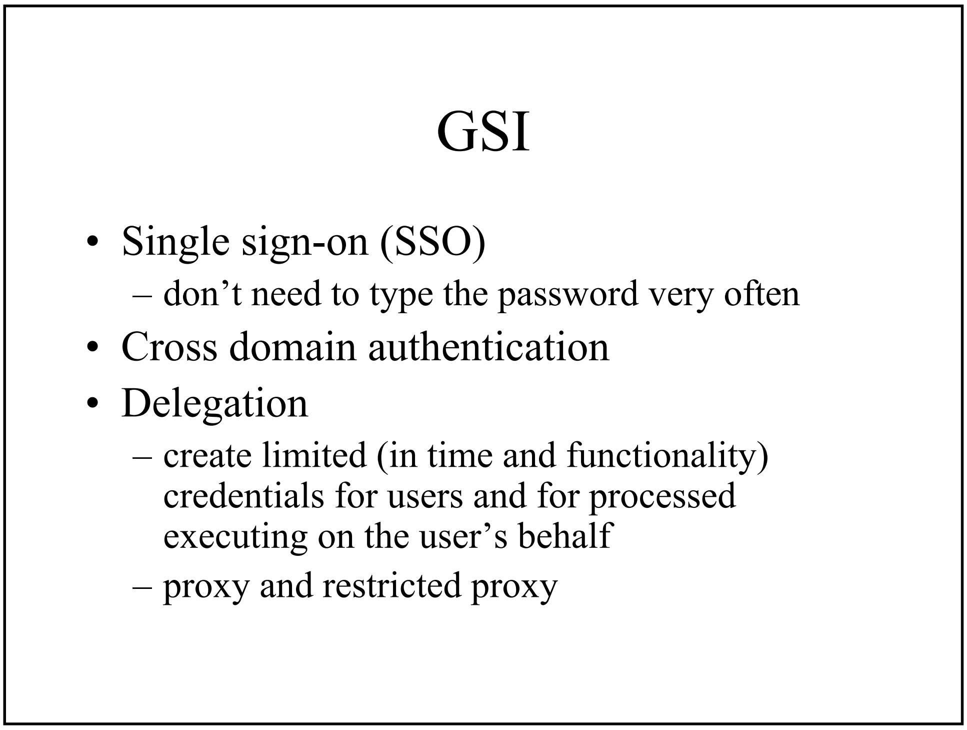
- A CA has its own certificate, signed by another CA
  - the verification of a user certificate requires the verification of all the steps in the chain
- A CA can self-signed its certificate
  - Root CA
  - The certificates of Root CAs are well known and difficult to forge


CA Cert

Root CAs

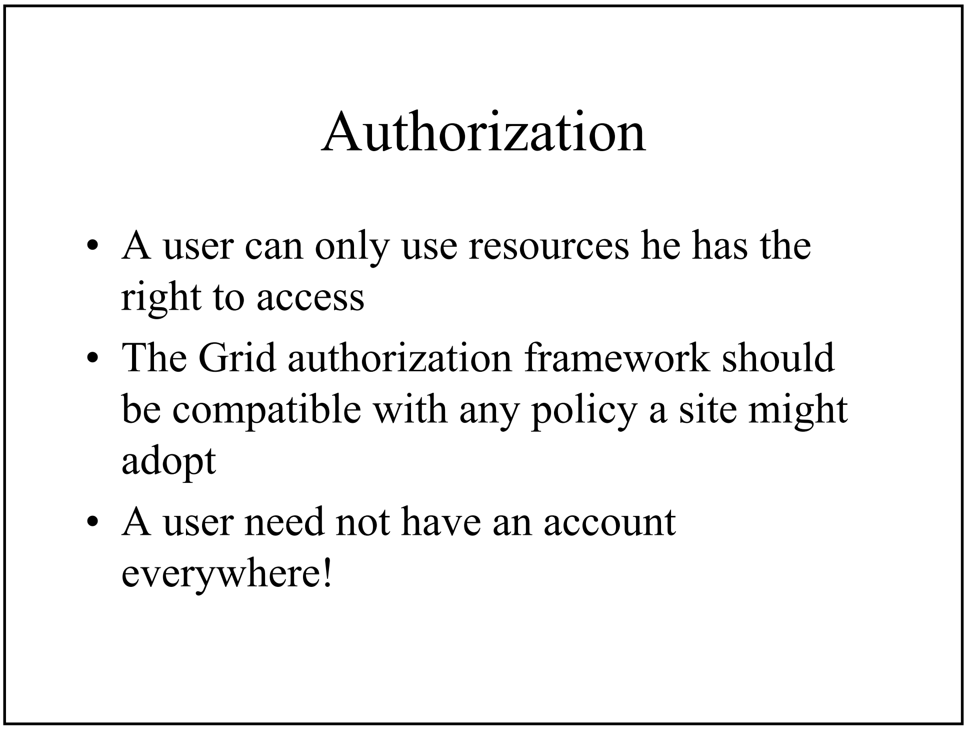
Grid Security Infrastructure

- GSI is based on an X.509 PKI
- Every user/host involved in the Grid has an X.509 certificate
- Certificates are signed by trusted CAs
- Each site trust the CAs it wants
- Every Grid transaction is mutually authenticated



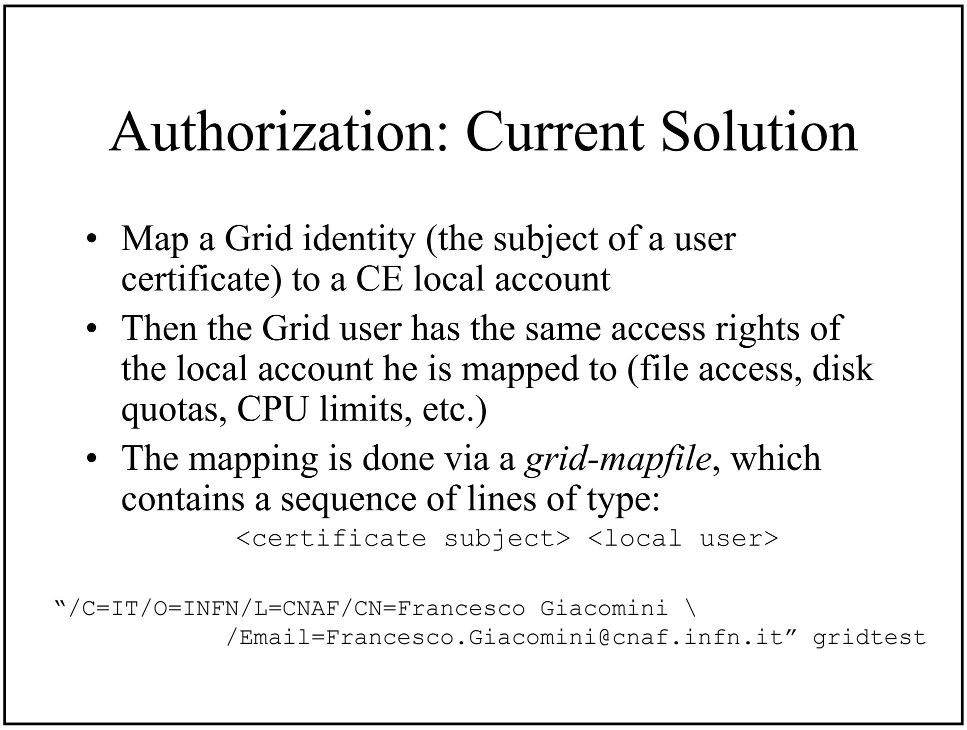
GSI

- Single sign-on (SSO)
  - don't need to type the password very often
- Cross domain authentication
- Delegation
  - create limited (in time and functionality) credentials for users and for processed executing on the user's behalf
  - proxy and restricted proxy



Authorization

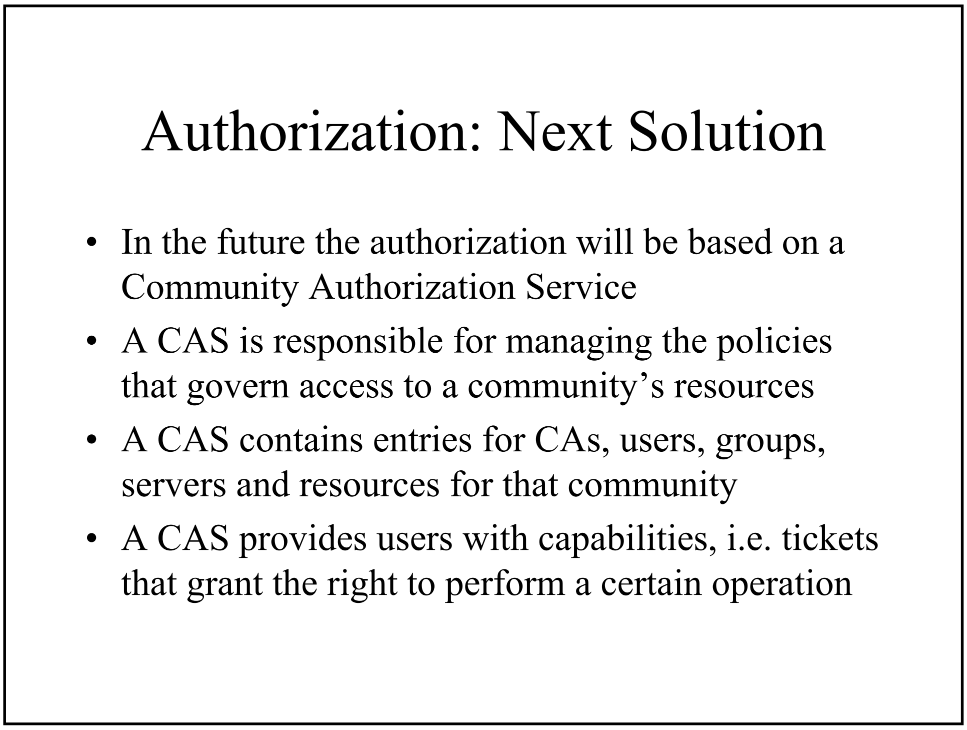
- A user can only use resources he has the right to access
- The Grid authorization framework should be compatible with any policy a site might adopt
- A user need not have an account everywhere!



Authorization: Current Solution

- Map a Grid identity (the subject of a user certificate) to a CE local account
- Then the Grid user has the same access rights of the local account he is mapped to (file access, disk quotas, CPU limits, etc.)
- The mapping is done via a *grid-mapfile*, which contains a sequence of lines of type:

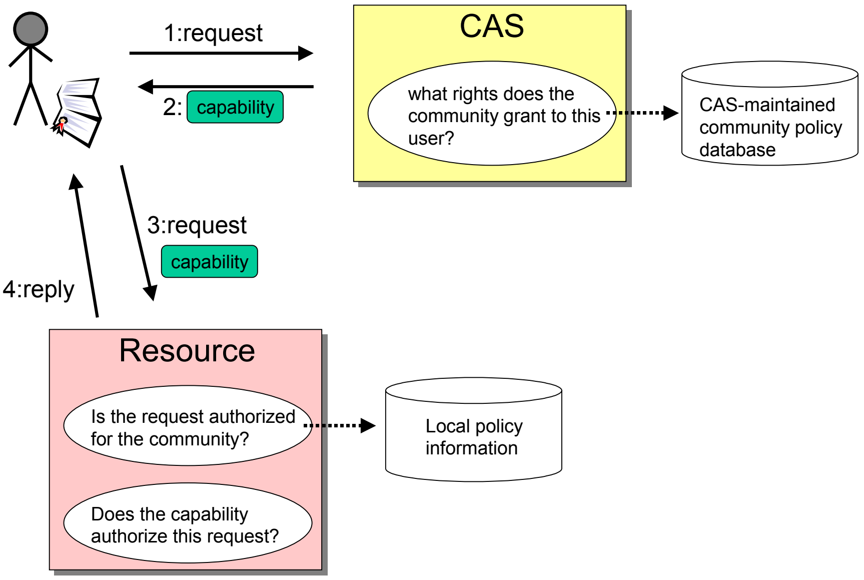
```
        <certificate subject> <local user>

“/C=IT/O=INFN/L=CNAF/CN=Francesco Giacomini \
            /Email=Francesco.Giacomini@cnaf.infn.it” gridtest
```



Authorization: Next Solution

- In the future the authorization will be based on a Community Authorization Service
- A CAS is responsible for managing the policies that govern access to a community’s resources
- A CAS contains entries for CAs, users, groups, servers and resources for that community
- A CAS provides users with capabilities, i.e. tickets that grant the right to perform a certain operation

How the CAS works



1:request

2: capability

CAS

what rights does the community grant to this user?

CAS-maintained community policy database

3:request

capability

4:reply

Resource

Is the request authorized for the community?

Does the capability authorize this request?

Local policy information

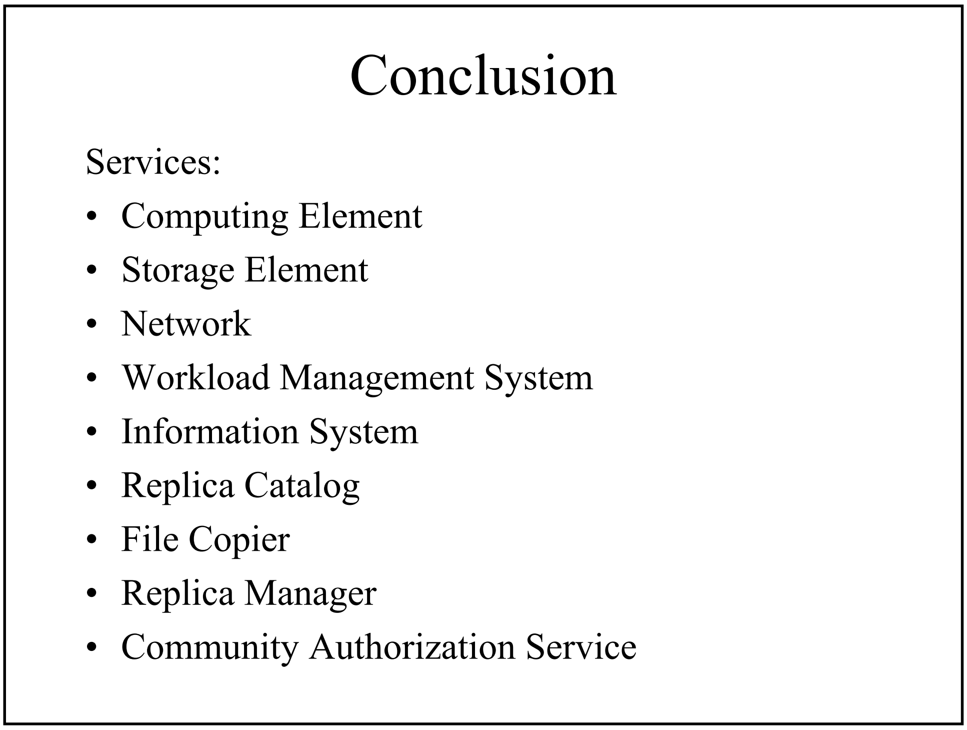
Conclusion

- The goal of the Grid is to provide an infrastructure to allow the sharing and coordinated use of resources within large, dynamic, multi-institutional communities for demanding applications

⇒ standard interfaces (protocol and API)

⇒ transparent access (e.g. for data and computation)

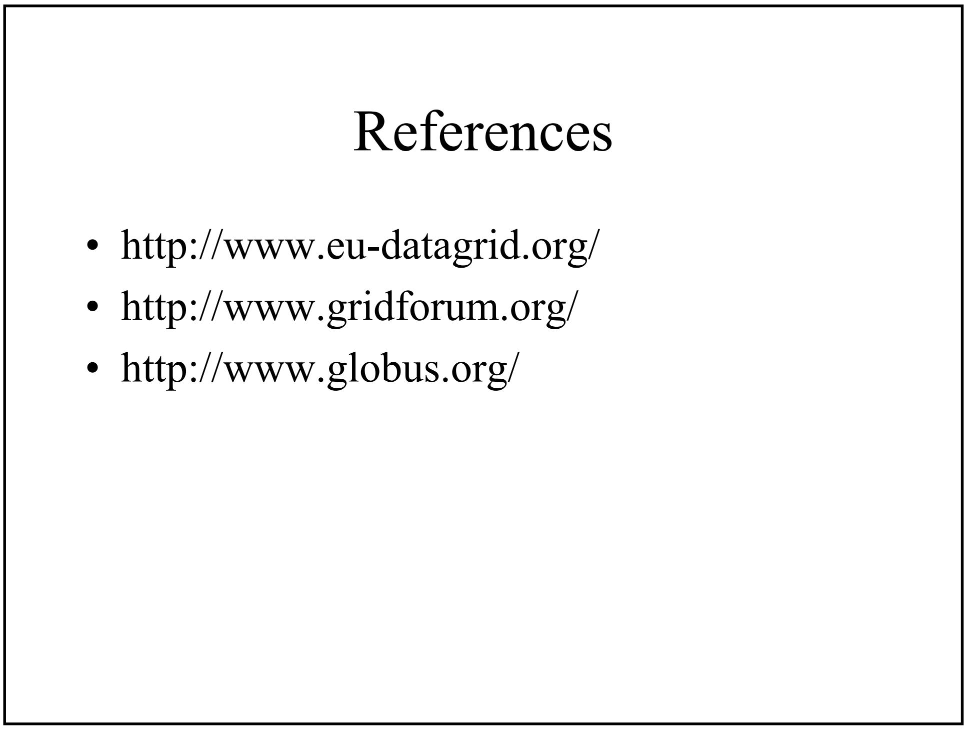
⇒ security



Conclusion

Services:

- Computing Element
- Storage Element
- Network
- Workload Management System
- Information System
- Replica Catalog
- File Copier
- Replica Manager
- Community Authorization Service



References

- http://www.eu-datagrid.org/
- http://www.gridforum.org/
- http://www.globus.org/