An Example Grid Middleware
- The Globus Toolkit
Globus Toolkit™

- A software toolkit addressing key technical problems in the development of Grid enabled tools, services, and applications
  - Offer a modular “bag of technologies”
  - Enable incremental development of Grid-enabled tools and applications
  - Implement standard Grid protocols and APIs (the “core” of the hourglass)
  - Is available under liberal open source license
General Approach

• Define Grid protocols & APIs
  – Protocol-mediated access to remote resources
  – Integrate and extend existing standards
  – “On the Grid” = speak “intergrid” protocols

• Develop a reference implementation
  – Open source Globus Toolkit
  – Client and server SDKs, services, tools, etc.

• Grid-enable wide variety of tools
  – Globus Toolkit, FTP, SSH, Condor, SRB, MPI, …

• Integrate user experience gathered through deployment and application integration
Four Key Protocols

Resource Management

Information Services

Data Management

Security

MCSN – N. Tonellotto – Complements of Distributed Enabling Platforms
Globus Open Source Grid Software

Security
- Data Management
- Execution Management
- Information Services
- Common Runtime

Web Services Components
- Pre-WS Authentication Authorization
  - GridFTP
  - Replica Location Service
- OGSA-DAI
- Reliable File Transfer
- Grid Resource Allocation Mgmt (WS GRAM)
- Monitoring & Discovery System (MDS4)
- Java WS Core

Non-WS Components
- Pre-WS Authentication Authorization
  - GridFTP
  - Replica Location Service
- OGSA-DAI
- Reliable File Transfer
- Grid Resource Allocation Mgmt (Pre-WS GRAM)
- Monitoring & Discovery System (MDS2)
- C Common Libraries
- XIO

Python WS Core [contribution]
- C WS Core

Globus Open Source Grid Software Components
- Delegation Service
- Community Authorization Service
- Community Scheduler Framework [contribution]

MCSN – N. Tonellotto – Complements of Distributed Enabling Platforms

5
Grid Security
Terminology

- Authentication
- Authorization
- Integrity
- Confidentiality
- Non-repudiation
- Delegation
- Single Sign On
- Digital Signature
Public Key Infrastructure

- PKI allows you to know that a given public key belongs to a given user
- PKI builds upon asymmetric encryption:
  - Each entity has two keys: public and private
  - Data encrypted with one key can only be decrypted with the other
  - The private key is known only to the owner
- The public key is given to the world encapsulated in a X.509 certificate
  - Similar to passport or driver’s license
Certificate Authorities

- A small set of trusted entities known as Certificate Authorities (CAs) are established to sign certificates
- A Certificate Authority is an entity that exists only to sign user certificates
- The CA signs its own certificate which is distributed in a trusted manner
- Examples: Verisign, DFN, ...
- The public key from the CA certificate can then be used to verify other certificates
Certificate Issuance

• To request a certificate a user starts by generating a key pair
• The private key is stored encrypted with a pass phrase the user gives
• The public key is put into a certificate request
• The user then takes the certificate to the CA
• The CA usually includes a Registration Authority (RA) which verifies the request:
  - The name is unique with respect to the CA
  - It is the real name of the user (through ID/passport check)
• The CA then signs the certificate request and issues a certificate for the user
Why Grid Security is Hard

- Resources being used may be valuable & the problems being solved sensitive
- Resources are often located in distinct administrative domains
  - Each resource has own policies & procedures
- Set of resources used by a single computation may be large, dynamic, and unpredictable
  - Not just client/server, requires delegation
- It must be broadly available & applicable
  - Standard, well-tested, well-understood protocols; integrated with wide variety of tools
# Grid Security Requirements

**User View**

1) Easy to use
2) Single sign-on
3) Run applications
   - FTP, SSH, MPI, Condor, Web, …
4) User based trust model
5) Proxies/agents (delegation)

**Resource Owner View**

1) Specify local access control
2) Auditing, accounting, etc.
3) Integration w/ local system
   - Kerberos, AFS, license mgr.
4) Protection from compromised resources

**Developer View**

1) API/SDK with authentication, flexible message protection, flexible communication, delegation, …
   a) Direct calls to various security functions (e.g. GSS-API)
   b) Security integrated into higher-level SDKs
Grid Security Infrastructure (GSI)

- Extensions to standard protocols & APIs
  - Standards: SSL/TLS, X.509 & CA, GSS-API
  - Extensions for single sign-on and delegation

- Globus Toolkit reference implementation of GSI
  - SSLeay/OpenSSL + GSS-API + single sign-on/delegation
  - Tools and services to interface to local security
    - Simple ACLs; SSLK5/PKINIT for access to K5, AFS, ...
  - Tools for credential management
    - Login, logout, etc.
    - Smartcards
    - MyProxy: Web portal login and delegation
    - K5cert: Automatic X.509 certificate creation
Delegation: proxies (I)

• Delegation = remote creation of a (second level) proxy credential
  – New key pair generated remotely on server
  – Proxy cert and public key sent to client
  – Clients signs proxy cert and returns it
  – Server (usually) puts proxy in temp dir
• Allows remote process to authenticate on behalf of the user
  – Remote process “impersonates” the user
During delegation, the client can elect to delegate only a **limited proxy**, rather than a “full” proxy (no process creation allowed)
  - Job submission client does this

Each service decides whether it will allow authentication with a **limited proxy**
  - Job manager service requires a full proxy
  - File server allows either full or limited proxy to be used

**A restricted proxy** is a generalization of the simple limited proxies
  - Desirable to have fine-grained restrictions
  - Reduces exposure from compromised proxies

Embed restriction policy in proxy certificates
  - Policy is evaluated by resource upon proxy use
  - Reduces rights available to the proxy to a subset of those held by the user
    - A proxy no longer grants full impersonation rights
Grid Security Infrastructure is ...

Proxies and delegation (GSI extensions) for secure single sign-on

- PKI (CAs and certificates)
- SSL/TLS

- PKI for credentials

- SSL for authentication and message protection
GSI in action

“Create Processes at A and B that Communicate & Access Files at C”

- Authorize
- Map to local ID
- Create process
- Generate credentials

* With mutual authentication

MCSN – N. Tonellotto – Complements of Distributed Enabling Platforms
Grid Security in Globus
1. Obtaining a Certificate

- The program grid-cert-request is used to create a public/private key pair and unsigned certificate:
  - usercert_request.pem: Unsigned certificate file
  - userkey.pem: Encrypted private key file
- Mail usercert_request.pem to ca@globus.org
- Receive a Globus-signed certificate
- Other organizations use different approaches
  - NCSA, NPACI, NASA, etc. have their own CA
Your New Certificate

Certificate:

Data:

Version: 3 (0x2)
Serial Number: 28 (0x1c)
Signature Algorithm: md5WithRSAEncryption
Issuer: C=US, O=Globus, CN=Globus Certification Authority

Validity

Not After : Apr 22 19:21:50 1999 GMT

Subject: C=US, O=Globus, O=NACI, OU=SDSC, CN=Richard Frost

Subject Public Key Info:

Public Key Algorithm: rsaEncryption
RSA Public Key: (1024 bit)
Modulus (1024 bit):
b4:e1:54:e7:87:57:b7:d0:61
Exponent: 65537 (0x10001)

Signature Algorithm: md5WithRSAEncryption

NTP is highly recommended

Your New Certificate

MCSN – N. Tonellotto – Complements of Distributed Enabling Platforms
2. “Logging on” to the Grid

• To run programs, authenticate to Globus:
  
  ```
  % grid-proxy-init
  Enter PEM pass phrase: ******
  ```

• Creates a temporary, local, short-lived proxy credential for use by our computations (RFC 3820) grid-proxy-init creates the local proxy file.

• User enters pass phrase, which is used to decrypt private key.

• Private key is used to sign a proxy certificate with its own, new public/private key pair.
  
  – User’s private key not exposed after proxy has been signed

• Proxy placed in temp dir, read-only by user

• NOTE: No network traffic!
3. “Logging off” from the Grid

- To destroy your local proxy that was created by grid-proxy-init:
  
  ```
  % grid-proxy-destroy
  ```

- This does NOT destroy any proxies that were delegated from this proxy.
  - You cannot revoke a remote proxy
  - Usually proxies with short lifetimes are created
Important Files (I)

- /etc/grid-security
  - hostcert.pem: certificate used by the server in mutual authentication
  - hostkey.pem: private key corresponding to the server’s certificate (read-only by root)
  - grid-mapfile: maps grid subject names to local user accounts (really part of gatekeeper)
- /etc/grid-security/certificates
  - CA certificates: certs that are trusted when validating certs, and thus need not be verified
  - ca-signing-policy.conf: defines the subject names that can be signed by each CA
Important Files (II)

• $HOME/.globus
  – usercert.pem: User’s certificate (subject name, public key, CA signature)
  – userkey.pem: User’s private key (encrypted using the user’s pass phrase)
• /tmp
  – Proxy file(s): Temporary file(s) containing unencrypted proxy private key and certificate (readable only by user’s account)
Globus Toolkit 2

GRAM
- process
- jobmanager
- gatekeeper

MDS
- Resources
- GRIS
- GIIS

GridFTP
- FTP server

RSL/HTTP1.1
- LDAP
- LDAP
- gsiftp/http/https/file

job allocation
job management

resource finding

data transfer
data control

proxy
use
initialize/destroy

user

Client
Grid Information Services
Resource Discovery/Monitoring

- Distributed resources and/or clients
- Resources status subject to change
- Dynamically variable connectivity and VOs
Grid “discovery”

• Resource Discovery
  – “What kind of resources could I use?”
  – Search status

• Resource Inquiry
  – “How can I compare resources (now)?”
  – Select status

• Resource Control
  – “How can I ‘take control’ of resources”
  – Acquisition status
  – It is not part of an information service!
Grid Information Service

- Provide access to static and dynamic information regarding system components
- A basis for configuration and adaptation in heterogeneous, dynamic environments
- Resource Description Services
  - Supplies information about a specific resource
- Aggregate Directory Services
  - Supplies collection of information which was gathered from multiple resource description services
  - Customized naming and indexing
Requirements

• Performance
• Scalability
• Cost
• Uniformity
• Expressiveness
• Extensibility
• Multiple information sources
• Dynamic data
• Access flexibility
• Security
• Easy to employ
• Decentralized maintainability
Monitoring and Discovery Service

Diagram showing the relationship between Host A, Host B, Host C, MDS Client, GIIS, GRIS, Information Provider, Resources, Idapsearch, and Idapadd/delete/modify slapd within the LDAP base.
MDS-2 Information Model

Object Class

dn: ou=cnuce, loc=pisa, o=CNR
ou=cnuce
loc=palazzo A

Entry

dn: ou=people, ou=cnuce, loc=pisa, o=CNR
ou = people

dn: ou=devices, ou=cnuce, loc=pisa, o=CNR
ou = devices

dn: uid=194567, ou=people, ou=cnuce, loc=pisa, o=CNR
cn = Nicola Tonellotto
loc = Lab Arch Par
loc = stanza 68
sn = khast
uid = 194567

dn: uid=barolo, ou=devices, ou=cnuce, loc=pisa, o=CNR
cn = barolo.cnuce.cnr.it
cn = 131.114.56.6
sn = barolo
loc = stanza 68
uid = barolo

Directory Information Tree
MDS-2 Functional Model

1. Search operation
2. Returned entry
3. Result code

LDAP client → LDAP server

1. Search operation
2. First entry returned
3. Second entry returned
4. Nth entry returned
5. Result code

LDAP client → LDAP server

1. Open connection and bind
2. Result of bind operation
3. Search operation
4. Returned entry #1
5. Returned entry #2
6. Result of search operation
7. Unbind operation
8. Closes connection

LDAP client → LDAP server
MDS-2 Data Representation (I)

Resource

Entry

< hn = backus.di.unipi.it
ou = Dip. Informatica
o = Università di Pisa
c = Italia >
MDS-2 Data Representation (II)
GLUE Schema: Storage Element
Grid Resource Management
Resource Management on HPC Resources

- HPC resources are usually parallel computers or large scale clusters

- The local resource management systems (RMS) for such resources includes:
  - configuration management
  - monitoring of machine state
  - job management

- There is no standard for this resource management
- Several different proprietary solutions are in use
- Examples for job management systems:
  - PBS, LSF, NQS, LoadLeveler, Condor
HPC Management Architecture in General

Control Service
Job Master

Resource and Job
Monitoring and Management Services

Compute Resources/
Processing Nodes
Computational Job

• A job is a computational task
  – that requires processing capabilities (e.g. 64 nodes) and
  – is subject to constraints (e.g. a specific other job must finish before the start of this job)

• The job information is provided by the user
  – resource requirements
    • CPU architecture, number of nodes, speed
    • memory size per CPU
    • software libraries, licenses
    • I/O capabilities
    – job description
    – additional constraints and preferences

• The format of job description is not standardized, but usually very similar
Transition to the Grid

More resource types come into play:

• Resources are any kind of entity, service or capability to perform a specific task
  – processing nodes, memory, storage, networks, experimental devices, instruments
  – data, software, licenses
  – people

• The task/job/activity can also be of a broader meaning
  – a job may involve different resources and consists of several activities in a workflow with according dependencies

• The resources are distributed and may belong to different administrative domains

• HPC is still the key application for Grids. Consequently, the main resources in a Grid are the previously considered HPC machines with their local RMS
Scope of Grids

Source: Ian Foster

Cluster Grid  Enterprise Grid  Global Grid
Implications to Grid Resource Management

• Several security-related issues have to be considered: authentication, authorization, accounting
  – who has access to a certain resource?
  – what information can be exposed to whom?

• There is lack of global information:
  – what resources are when available for an activity?

• The resources are quite heterogeneous:
  – different RMS in use
  – individual access and usage paradigms
  – administrative policies have to be considered
Resource Management Layer

Grid Resource Management System consists of:

- **Local resource management system (Resource Layer)**
  - Basic resource management unit
  - Provide a standard interface for using remote resources
  - Grid Resource Allocation Manager (GRAM)

- **Global resource management system (Collective Layer)**
  - Coordinate all Local resource management system within multiple or distributed Virtual Organizations (VOs)
  - Provide high-level functionalities to efficiently use all of resources
    - Job Submission
    - Resource Discovery and Selection
    - Scheduling
    - Co-allocation
    - Job Monitoring, etc.
  - e.g. Meta-scheduler, Resource Broker, etc.
Grid Resource Management

Grid Middleware

Local Resource Management

Resource Broker

User/Application

Higher-Level Services

Core Grid Infrastructure Services
- Information Services
- Monitoring Services
- Security Services

Grid Resource Manager

Grid Resource Manager

Grid Resource Manager

PBS

LSF

...

Resource

Resource

Resource
Definitions

- **Resource**: entity able to execute one or more jobs on the behalf of the user

- **Client**: process using GRAM protocol to submit a job request

- **Job**: one or more processes being part of a job request

- **Job request**: a message containing the request and the specification for a job execution on a remote resource. A typical job request specifies:
  - When and where processes should be created
  - How and what processes to create
  - How to execute and terminate processes

- **Gatekeeper**: remote resources service managing incoming job requests (GT2)

- **Job Manager**: service instantiated by the gatekeeper to manage the execution and monitor the job’s processes (GT2)
The gatekeeper is a daemon process running with root privileges on each computational resource on the Grid. When it receives a job request from a client it executes the following operations:

– Performs mutual authentication
– Maps the client in a local user
– Instantiates a job manager process with local user rights
– Sends required arguments to allocate job’s processes to the job manager
GRAM components
Job Status

job status \neq\ process status
RSL Examples

`& (executable=/bin/ls)
  (directory=/tmp)
  (arguments=-l)
  (environment=(COLUMNS 40))`

`& (executable=/bin/ls)
  (count=3) (project=GlobusTeach)
  (mAxCpUtImE=10) (max_Memory=30)`

`& (executable=https://barolo.cnuce.cnr.it::65092/bin/ls)`

`& (executable=/bin/ls)
  (stdout=$(HOME)/ls.out)
  (stderr=$(HOME)/ls.err)`
The deliverables of the OGF Working Group JSDL:

• A specification for an abstract standard Job Submission Description Language (JSDL) that is independent of language bindings, including:
  – the JSDL feature set and attribute semantics,
  – the definition of the relationship between attributes,
  – and the range of attribute values.

• A normative XML Schema corresponding to the JSDL specification.

• A document of translation tables to and from the scheduling languages of a set of popular batch systems for both the job requirements and resource description attributes of those languages, which are relevant to the JSDL.
JSDL Attribute Categories

• The job attribute categories will include:
  
  – Job Identity Attributes
    • ID, owner, group, project, type, etc.
  
  – Job Resource Attributes
    • hardware, software, including applications, Web and Grid Services, etc.
  
  – Job Environment Attributes
    • environment variables, argument lists, etc.
  
  – Job Data Attributes
    • databases, files, data formats, and staging, replication, caching, and disk requirements, etc.
  
  – Job Scheduling Attributes
    • start and end times, duration, immediate dependencies etc.
  
  – Job Security Attributes
    • authentication, authorization, data encryption, etc.
And the collective layer?

• The GRAM protocol provides a standard interfaces to access local resources

• At collective layer:
  – Resource brokers
  – Metaschedules

• We will speak about scheduling in desktop computers, clusters and Grids in subsequent lectures
Metascheduler Example

Directed Acyclic Graph Manager

- DAGMan allows you to specify the dependencies between your Condor-G jobs, so it can manage them automatically for you.
- (e.g., “Don’t run job “B” until job “A” has completed successfully.”)
- A DAG is defined by a `.dag file`, listing each of its nodes and their dependencies:

  ```
  # diamond.dag
  Job A a.sub
  Job B b.sub
  Job C c.sub
  Job D d.sub
  Parent A Child B C
  Parent B C Child D
  ```

- each node will run the Condor-G job specified by its accompanying Condor submit file

Source: Miron Livny
Grid Data Management
Data Management Services

• Data access and transfer
  – **GASS**: Simple multi-protocol tool to transfer ‘normal’ files; integrated in GRAM
  – **GridFTP**: Reliable and high-performance file transfer protocol for ‘big’ files in computer networks

• Replica Management
  – **Replica Catalog**: Service to keep updated information on sets of replicated data
  – **Replica Management**: Service to create and manage sets of replicated data
GASS: Global Access to Secondary Storage

- Access to files from remote locations
- Used by GRAM to:
  - Download executables from remote sites (batch)
  - Move stdin/stdout/stderr to/from remote sites (stream)
- Components:
  - GASS file access API:
    OS-spec open/close changed with `globus_gass_open/close`;
    Read/write calls automatically managed
  - Job specification language extensions
    Specific URIS used to declare remote executables and stdin/
    stdout/stderr
  - Utilities to manage caches for remote data
GASS Architecture

main( ) {
    fd = globus_gass_open(…)
    ...
    read(fd, …)
    ...
    globus_gass_close(fd)
}

(a) GASS file access API

(b) RSL extensions

(c) Remote cache management

% globus-gass-cache

(d) Low-level APIs for customizing cache & GASS server
GASS – GRAM Integration

• Resource names coding convention
  
  https://barolo.cnuce.cnr.it:9991/~khast/myjob

  protocol    server address    file name

• Supported protocol: http, https, ftp, gsiftp

• Job specification language extensions:
  
  – executable, stdin, stdout, stderr can be local files or URI
  
  – Executable and stdin loaded from local cache before job run
  
  – stdout, stderr managed by GASS in append mode
  
  – Cache flushed at job end
Globus components in action

Globus components in action

Local Machine

- globus-job-run
- RSL multi-request

Globusrun

- RSL parser
- Broker
- RSL single request

GRAM Client

- GSI

Machines

- RSL string
- User Proxy Cert

grid-proxy-init

Remote Machine

- App
- Nexus
- AIX
- MPI

Remote Machine

- App
- Nexus
- Solaris
- MPI

GRAM Gatekeeper

- GSI

Remote Machine

GRAM Job Manager

- GASS Client

Remote Machine

GRAM Gatekeeper

- GSI

Remote Machine

GRAM Job Manager

- GASS Client

Remote Machine

GASS Server

- GASS Client

Remote Machine

GASS Server

- GASS Client
FTP protocol in a nutshell

- Control and data channel
- Transfers: client-server and third party
GridFTP Add-ons

- **Parallel Transfer**: multiple data pathway

- **Striped Transfer**: distributed and parallel data transfer
Why?

- Network throughput maximization (TCP window)

- Network losses minimization (glitch)
Data management

• Data can be **present** in different sites
• Data can be **replicated** in different locations
• How to **manage** the copies of data on the Grid?
• How to **leverage** the copies of data?

• Low level: Replica Catalog
  – A **catalog** represents files, collections and locations
  – A collection is represented by a **logical file**
  – A replica (partial or complete) of a collection is represented by a **physical file**
  – Given a **logical filename**, how to obtain the relevant **physical filename**?
    • Physical File Name (PFN): host + full path & file name
    • Logical File Name (LFN): logical name unique in the Grid
    • LFN : PFN = 1 : n
Replica Catalog

Logical Collection
C02 measurements 1998
Filename: Jan 1998
Filename: Feb 1998
...

Logical Collection
C02 measurements 1999

Location
jupiter.isi.edu
Filename: Mar 1998
Filename: Jun 1998
Filename: Oct 1998
Protocol: gsiftp
UrlConstructor: gsiftp://jupiter.isi.edu/nfs/v6/climate

Location
sprite.llnl.gov
Filename: Jan 1998
...
Filename: Dec 1998
Protocol: ftp
UrlConstructor: ftp://sprite.llnl.gov/pub/pcmrd

Logical File Parent
Logical File
Jan 1998
Size: 1468762

Logical File
Feb 1998
Replica Management Services

1. Attributes of Desired Data
2. Logical File Names
3. Locations of one or more replicas
4. Sources and destinations of candidate transfers
5. Performance Measurements and Predictions
6. Replica Management Service
7. Information Services
8. Location of selected replica
DataGrid: complete architecture

Metadata Catalog

Application

Replica Catalog

Replica Selection

Multiple Locations

Performance Information & Predictions

MDS

NWS

Attribute Specification

Logical Collection and Logical File Name

Selected Replica

GridFTP Control Channel

Disk Array

Disk Cache

Tape Library

Disk Cache

Replica Location 1

Replica Location 2

Replica Location 3

GridFTP Data Channel

MCSN – N. Tonellotto – Complements of Distributed Enabling Platforms 68