Seminar Review
Topics

- Grid Computing Middleware
- Grid Resource Management
- Grid Computing Security
- Applications of SOA and Web Services
- Semantic Grid
- Grid & E-Science
- Grid Economics
- Cloud Computing
Grid Computing Middleware

- Definitions & functions
- Middleware components
- Globus
- gLite
What is Grid middleware?

- System software between applications and OS
  - Provide services to applications
    - Discovery
    - Execution
    - Storage
    - Data movements
    - Information
    - Service integration
    - Failure detection and recovery
    - Resource monitoring
  - Hide all complexities of the Grid environment
Purposes of Middleware

- Open, general-purpose and standard
- Standard protocols
  - Defines the contents and sequence of message exchanges used to request remote operation
  - Important and essential to achieve the interoperability that Grid depends on
- Standard APIs
  - Interfaces to code libraries
  - Facilitate construction of Grid components by allowing code components to be reused
Middleware Components

- Tools and applications
  - Directory brokering, diagnostics, and monitoring
  - Secure access to resources and services
  - Diverse resources such as computers, storage media, networks, and sensors

- User applications
  - Collective services
  - Resource and connectivity protocols
  - Fabric
Globus layered architecture

Applications

High-level Services and Tools

Testbed Status

Core Services

Local Services
Globus Toolkit 4x

- Sustainable changes on the services interoperability and infrastructure
  - Open Grid Services Architecture (OGSA)
    - Stateful Web Services
    - Enable the integration of user specific Grid services
      - Define standard interfaces
    - How to access Grid services
  - Disadvantage
    - Slow
GT: Core service architecture
Globus Toolkit

- Grid Service Specification
  - How to write, publish and use a Grid service
- GT components:
  - GT core
    - Meta-services used to implement other services and service behaviors (e.g. service creation, destruction)
  - GT base services
    - Use the GT core to implement Grid capacities: resource management, information services, data transfer, etc.
  - Other Grid services
    - Implemented by user to enable some enhancement capacities
GT: from other perspectives

Local-level services
- WS GRAM
  - Fork
  - PBS
  - LSF
- GridFTP
- RFTP
- ...

VO-level services
- Information service
- ...

GT container
GT security
What GT DOES NOT address

- GT focus on accessing local resources
- Things still missing
  - Coordination services
    - Resource/service discovery
  - Information collection
  - Resource connectivity
  - Programming models/tools
- Things to be improved
  - Performance!
gLite Toolkit

- The Enabling Grids for E-sciencE project (EGEE)
- Middleware stack that combines components developed in various related projects.
gLite middleware

- A layer between services and resources
- gLite follows a Service Oriented Architecture
gLite - guiding principles

- Service oriented approach
  - Allow for multiple interoperable implementations
- Lightweight (existing) services
  - Easily and quickly deployable
  - Use existing services where possible
    - Condor, EDG, Globus, LCG, …
- Portable
  - Being built on Scientific Linux and Windows
- Security
  - Sites and Applications
gLite - guiding principles

- Performance/Scalability & Resilience/Fault Tolerance
  - Comparable to deployed infrastructure
- Co-existence with deployed infrastructure
  - Co-existence with LCG-2 and OSG (US) are essential for the EGEE Grid services
- Site autonomy
  - Reduce dependence on ‘global, central’ services
- Open source license
gLite Architecture
## Middleware Comparison

<table>
<thead>
<tr>
<th></th>
<th>GT4</th>
<th>gLite</th>
<th>UNICORE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication</td>
<td></td>
<td>X.509 Certificates</td>
<td></td>
</tr>
<tr>
<td>Web Services</td>
<td>OGSA afford</td>
<td>no, due to 3.1</td>
<td>WSRF 1.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Manual</td>
<td></td>
</tr>
<tr>
<td>I/O</td>
<td></td>
<td>GridFTP</td>
<td></td>
</tr>
<tr>
<td>API</td>
<td>C, Java, Python</td>
<td>C</td>
<td>Java</td>
</tr>
<tr>
<td>GUI</td>
<td>web</td>
<td>Scientific Linux</td>
<td>java swing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3/4</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>IA 32/64</td>
<td></td>
</tr>
<tr>
<td>Servers</td>
<td>unix-ish</td>
<td></td>
<td>unix-ish</td>
</tr>
</tbody>
</table>
Grid Resource Management

- Definitions
- Functions
- Resource Management Models
Resources

- Physical resource: computer, disk, database, networks, scientific instruments.
- Logical resource: Executing applications, Complex workflows…
Definition

- The term resource management refers to the operations used to control how capabilities provided by Grid resources and services are made available to other entities, whether users, applications or services.
Functions

- Discovery
- Allocations
- Negotiation
- Access
Models

- Hierarchical
- Abstract Owner
- Economics
Grid Security

- Scenarios
- Symmetric keys & Asymmetric keys
- Digital signature
- Grid authentication and delegation
Grid security: a scenario
Symmetric key

plaintext → encryption → ciphertext → decryption → plaintext
Asymmetric cryptography

- Public key
- Private key

**Unique Mathematical relation**

plaintext ➔ encryption ➔ ciphertext ➔ decryption ➔ plaintext
Digital Signatures

- A message signed by the private-key

1. Data
2. Hash
3. Generate Signature
4. Private Key of Sender
5. Verify Signature
6. Public Key of Sender
Authentication models

Direct-trusted model

1: Register sender’s public key
2: Store trusted public key?
3: Message M
4: Sign(M) with sender’s private key

Third-party trusted model

1: Register sender’s public key
2: Store trusted public key?
3: Message M
4: Sign(M) with sender’s private key
5: Public key existed?

Trusted authority

Digital certificate

Key repository
Digital certificate

- Based on asymmetric cryptography
- Consist of
  - Public key
  - User identity information (user name, organization, address, ...)
  - One or more digital signature signed by some well known Certificate Authority
Certificate authority

- Responsible for
  - Positively identify entities requesting certificates
  - Issuing, removing, and archiving certificates
  - Protecting the Certificate Authority server
  - Maintaining a namespace of unique names for certificate owners
  - Serve signed certificates to those needing to authenticate entities
  - Logging activity
Grid authentication
Grid Delegation
Delegation

**Grid Host A**
1. Create secure communication
2. Request to create proxy
3. Sign proxy certificate and send back
   - Password
   - Your certificate
   - Your private key

**Grid Host B**
4. Create proxy certificate request
5. Send your certificate
6. Send your certificate and proxy certificate

**Grid Host C**
7. Path validation & get proxy public key
   - Your certificate
   - CA’s public key

8. Encrypt request & send
   - Proxy private key

9. Decrypt

10. Mapping & execution
    - local user name
    - Proxy subject
    - "<Subjects" <UserName>
    - grid-mapfile
SOA & Grid

- Service Oriented Architecture
- Web Service
- Grid Service
- Open Grid Service Architecture (OGSA)
- Web Service Resource Framework