Grid Computing Environments

Grid: a system supporting the coordinated resource sharing and problem-solving in dynamic, multi-institutional virtual organizations.

Note: some of this material was taken from the Globus tutorial at www.globus.org

A Scenario

A fundamental change in the problems needed to be solved:

- multidisciplinary nature → composition of expertise
- larger problem scale → composition of computing resources

Mary is a university researcher in North America who is collaborating with two colleagues - Hans at a corporate R&D facility in Europe and Ling at a research institute in Asia. Mary has a new research model that she would like to validate using data collected by Ling and compare the results to that of the proprietary model developed by Hans for his corporation. Each of the three collaborators have previously constructed software components that provide access to the resources they are contributing to the collaboration. Mary and Hans are using a shared workspace system to interact synchronously in performing the computations, viewing the results, and saving the results for later study. Ling is not available at the time of the interaction and needs to provide appropriate access so that her data is available to Mary and Hans. The results, stored by Mary, should also be available later for study by Ling.
Grid Computing Environments

**Virtual Organization**: a set of individuals and/or institutions collaborating to achieve a common goal within a set of rules defining the controlled sharing of computational resources.

Characteristics of virtual organizations:

- flexible relationships (client-server, P2P, bokered, …)
- complex sharing rules (access control, delegation, …)
- varied resources (programs, storage, devices, …)
- diverse usage modes (single vs. multi-user, performance vs. cost sensitive, synchronous vs. asynchronous, …)

---

**Grid vs. Web**

**Web**: Uniform naming/access to documents

**Grid**: Uniform, high-performance access to computational resources

*On-demand creation of powerful virtual organizations*
**Grid vs. Web**

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Grid</th>
<th>Web</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seamless naming</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Uniform security/authentication</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Information services</td>
<td>Yes</td>
<td>Yes/no</td>
</tr>
<tr>
<td>Co-scheduling</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Accounting/authorization</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>User services</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Event services</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Global shell</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>

**Layered Grid Architecture**

"Coordinating multiple resources": ubiquitous infrastructure services, app-specific distributed services

"Sharing single resources": negotiating access, controlling use

"Talking to things": communication (Internet protocols) & security

"Controlling things locally": Access to, & control of, resources
“Hourglass” principle

Protocols, Services, and Interfaces
Protocols, Services, Interfaces:

- **Protocol**-mediated access to resources
  - Mask local heterogeneities
  - Extensible to allow for advanced features
  - Negotiate multi-domain security, policy
  - “Grid-enabled” resources speak protocols
  - Multiple implementations are possible

- Broad deployment of protocols facilitates creation of **Services** that provide integrated view of distributed resources

- **Interfaces (APIs/SKDs)** use protocols and services to enable specific classes of applications

---

**Globus**

```
Collective
  ↓
Resource
  ↓
Connectivity
  ↓
Fabric
```

- Grid Information Index Servers (GIIS)
- Replica management
- Certificate repository (MyProxy)
- Co-allocation library (DUROC)

- Grid Resource Information Service (GRIS)
- Grid Resource Access and Management (GRAM)
- GridFTP

- Internet protocols
- Globus Security Infrastructure (GSI)

- NSF’s National Technology Grid
- NASA’s Information Power Grid
Fabric Layer

• A diverse mix of resources that may be shared
  – Individual computers, Condor pools, file systems,
    archives, metadata catalogs, networks, sensors, etc., etc.
• Few constraints on low-level technology:
  connectivity and resource level protocols form the
  “neck in the hourglass”
• Defined by interfaces not physical characteristics

Connectivity Layer

• Communication
  – Internet protocols: IP, DNS, routing, etc.
• Security: Grid Security Infrastructure (GSI)
  – Uniform authentication & authorization
    mechanisms in multi-institutional setting
  – Single sign-on, delegation, identity mapping
  – Public key technology, SSL, X.509, GSS-API
  – Supporting infrastructure: Certificate Authorities,
    key management, etc.
Why Grid Security is Hard

• Resources being used may be extremely valuable & the problems being solved extremely sensitive
• Resources are often located in distinct administrative domains
  – Each resource may have own policies & procedures
• Set of resources used by a single computation may be large, dynamic, and unpredictable
  – Not just client/server
• It must be broadly available & applicable
  – Standard, well-tested, well-understood protocols
  – Integration with wide variety of tools

Grid Security Requirements

<table>
<thead>
<tr>
<th>User View</th>
<th>Resource Owner View</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Easy to use</td>
<td>1) Specify local access control</td>
</tr>
<tr>
<td>2) Single sign-on</td>
<td>2) Auditing, accounting, etc.</td>
</tr>
<tr>
<td>3) Run applications</td>
<td>3) Integration w/ local system</td>
</tr>
<tr>
<td>ftp,ssh,MPI,Condor,Web,...</td>
<td>Kerberos, AFS, license mgr.</td>
</tr>
<tr>
<td>4) User based trust model</td>
<td>4) Protection from compromised resources</td>
</tr>
<tr>
<td>5) Proxies/agents (delegation)</td>
<td></td>
</tr>
</tbody>
</table>

Developer View

API/SDK with authentication, flexible message protection, flexible communication, delegation, ...
  Direct calls to various security functions (e.g. GSS-API)
  Or security integrated into higher-level SDKs:
    E.g. GlobusIO, Condor-G, MPICH-G2, HDF5, etc.
Secure Remote Startup

1. Exchange certificates, authenticate, delegate
2. Check gridmap file
3. Lookup service
4. Run service program (e.g. jobmanager)

Resource Layer

- Grid Resource Allocation Mgmt (GRAM)
  - Remote allocation, reservation, monitoring, control of compute resources
- GridFTP protocol (FTP extensions)
  - High-performance data access & transport
- Grid Resource Information Service (GRIS)
  - Access to structure & state information
- Network reservation, monitoring, control
- All integrated with GSI: authentication, authorization, policy, delegation
**Metacomputing Directory Services**

- Resources run a standard information service (GRIS) which speaks LDAP and provides information about the resource (no searching).
- GIIS provides a “caching” service much like a web search engine. Resources register with GIIS and GIIS pulls information from them when requested by a client and the cache as expired.
- GIIS provides the collective-level indexing/searching function.

**Resource Management**

- The Globus Resource Allocation Manager (GRAM) protocol and client API allows programs to be started on remote resources, despite local heterogeneity.
- Resource Specification Language (RSL) is used to communicate requirements.
- A layered architecture allows application-specific resource brokers and co-allocators to be defined in terms of GRAM services
  - Integrated with Condor, PBS, MPICH-G2, …
GridFTP

- Suite of communication libraries and related tools that support
  - GSI, Kerberos security
  - Third-party transfers
  - Parameter set/negotiate
  - Partial file access
  - Reliability/restart
  - Large file support
  - Data channel reuse
  - Integrated instrumentation
  - Loggin/audit trail
  - Parallel transfers
  - Striping (cf DPSS)
  - Policy-based access control
  - Server-side computation
  - Proxies (firewall, load bal)

- All based on a standard, widely deployed protocol
GridFTP

• Why FTP?
  – Ubiquity enables interoperation with many commodity tools
  – Already supports many desired features, easily extended to support others
  – Well understood and supported
• We use the term GridFTP to refer to
  – Transfer protocol which meets requirements
  – Family of tools which implement the protocol
• Note GridFTP > FTP

GridFTP: Basic Approach

• FTP protocol is defined by several IETF RFCs
• Start with most commonly used subset
  – Standard FTP: get/put etc., 3rd-party transfer
• Implement standard but often unused features
  – GSS binding, extended directory listing, simple restart
• Extend in various ways, while preserving interoperability with existing servers
  – Striped/parallel data channels, partial file, automatic & manual TCP buffer setting, progress monitoring, extended restart
Collective Layer

- Index servers aka metadirectory services
  - Custom views on dynamic resource collections assembled by a community
- Resource brokers (e.g., Condor Matchmaker)
  - Resource discovery and allocation
- Replica catalogs
- Co-reservation and co-allocation services

Replica Management

- Maintain a mapping between logical names for files and collections and one or more physical locations
- Important for many applications
  - Example: CERN HLT data
    - Multiple petabytes of data per year
    - Copy of everything at CERN (Tier 0)
    - Subsets at national centers (Tier 1)
    - Smaller regional centers (Tier 2)
    - Individual researchers will have copies
DUROC

- Simultaneous allocation of a resource set
  - Handled via optimistic co-allocation based on free nodes or queue prediction
  - In the future, advance reservations will also be supported
- globusrun will co-allocate specific multi-requests
  - Uses a Globus component called the Dynamically Updated Request Online Co-allocator (DUROC)