Resource Scheduling, Authentication and Authorization in Large Institutional Grids

Abhijit Bose, Ph.D.
Associate Director, Michigan Grid Research and Infrastructure Development (MGRID)
and
Center for Advanced Computing (CAC)
The University of Michigan
Ann Arbor, MI 48109
abose@eecs.umich.edu
Multiple Grid efforts at the U of M
- Cluster Computing (ATLAS, CAC/NPACI, DZero)
- Automated network configuration and testing, Network QoS reservation (CITI, ITCom)
- Remote Instrument (SI – NEES Earthquake Grid)
- Collaborative tools (SI – CHEF Collaboration portal)
- Data base searches (Bioinformatics, MCBI)

Current Grid technology is designed for small communities (100s of users)
- Integration challenge for U-M (tens of thousands of users)
The Common Problem
The Promise of Grids

So far, small and incremental steps towards this goal.

http://www.mgrid.umich.edu  abose@eecs.umich.edu
Why MGRID?

Grid software (Globus etc.) is difficult to run, complex to install and manage

Promote ease of use
More time to do science, instead of IT management

How to prototype the Grid to fit into UM IT environment

Large (> 100,000) user base for Grid service
Produce a generalized Grid service

http://www.mgrid.umich.edu  abose@eecs.umich.edu
Why MGRID

Middleware issues are difficult (AAAA)
  Authorization, Authentication, Accounting, Auditing
Leverage existing security and group services
Add Fine grained policy driven access control
  Let the owners of resources control their resource
    Who, what, where, when, and how
    But make it easy for them to do so

http://www.mgrid.umich.edu  abose@eecs.umich.edu
Why MGRID

• Campus wide meta scheduler for common resources
  – Computers, Collaboration Technology, Laboratory Instruments
• Enhance accounting capabilities
• Add auditing capabilities
• Use MGRID testbed to pursue distributed systems and applications research
MGRID Partnership

• Goal: build pilot institutional grid

Founding Partners

External Sponsors

NFS
NMI/NSF
NEES
Mid-America Earthquake Center

http://www.mgrid.umich.edu  abose@eecs.umich.edu
MGRID Overview

- Portal
- Security
- MGRID
- Scheduling
- Applications
MGRID PROJECTS

CORE INFRASTRUCTURE

Kerberos Leveraged PKI
- kx509 Clients, KCT and mod_KCT Apache web server modules authenticate users against Globus Gatekeepers (password-less)

MARS
- robust task scheduling and resource management forecasting algorithm to predict resource-level scheduling parameters such as queue lengths, turn-around times, and resource utilization.
- research fault-tolerant scheduling of tasks

GridNFS
- integrates distributed file system (NFSv4) and flexible identity management to meet the needs of grid-based virtual organizations.

http://www.mgrid.umich.edu  abose@eecs.umich.edu
CORE INFRASTRUCTURE

WALDEN
eliminates the need to manage user identities on hosts that participate in a grid environment. This is accomplished by moving user authentication to the client, replacing the static mapping between X.509 identities (Distinguished Names) and local user names in the Globus grid-mapfile with a dynamic approach using secure LDAP.

Accounting
allows usage reports on disparate scheduler log formats, such as PBSPro and Condor. Usage logs are translated into a common, standard XML format (defined by GGF UR-WG).
MGRID PROJECTS

MGRID APPLICATIONS AT MICHIGAN

ATLAS
UltraLight
NEESGrid
BioPhysics (Gaussian, Protein Folding)
Chemistry
Agent-Based Simulations, Financial Modeling
NTAP
Secure Multipoint Video-Conferencing

PORTAL SOFTWARE

MGRID Portal (CHEF-based)
SAKAI/MGRID

http://www.mgrid.umich.edu  abose@eecs.umich.edu
Existing U of M Services

Uniqname
  Unique campus wide user name to UID
Kerberos V5 (multiple cells)
KX509
Group Services
  AFS PTS, LDAP
Directory services
  LDAP
MGRID Portal

Proxy KX509 credentials, keep the Globus client off workstations
Ease of use for U of M faculty, staff, and students

Kerberos + kx509 + browser = Grid access

Single point for PKI management

CA self-signed keys
CA policy files

Single entry point for Grid resources

http://www.mgrid.umich.edu  abose@eecs.umich.edu
MGRID Portal

User workstation
  KX509 to obtain user X509 credentials
  KX509 Certificate available to browser
Additions to OpenSSL (in 9.0.7), required on MGRID Portal
  SSL handshake recorded
MGRID Portal SSL configured to require user X509 credentials
MGRID Portal

SSL Handshake transcript
Contains all packets exchanged
Allows KCT (Kerberos Credential Translator) to repeat user certificate verification
Handshake time stamp used

Apache module, mod_kct
Sends ssl handshake transcript to KCT service
Requests KCA Kerberos service ticket
MGRID Portal

Apache module, mod_kx509
  Uses the KCA TGS
  Obtains user proxy KX509 credentials
  Places them in a ticket file

Apache module, mod_php
  Creates RSL, uses KX509 credentials

CHEF runs in Tomcat
  Communicates with Apache through mod_jk
  Creates RSL, uses KX509 or MyProxy credentials
MGRID Portal

Hides complexity from user
Individual or Organizational presentation
CHEF
Easily extensible
Add new Grid applications
With generic Grid resource, can run any back-end program
Built on strong security

http://www.mgrid.umich.edu  abose@eecs.umich.edu
MGRID Architecture

Apache
- mod ssl
- mod kct
- mod kx509
- mod jk
- mod php

Tomcat
- CHEF

Kerberos V5
- KCT
- KCA
- KDC

User Workstation
- Browser
  - libpkcs11
  - kx509
  - kinit

Meta Scheduler
- GateKeeper
  - Resource Mng
- Resource
  - Grid Resource

Authorization
- LDAP
  - Authorization

MGRID Portal
- Apache
- Tomcat
- CHEF

SSL
Client Certificate required

Browser
- kinit

Meta Scheduler
- GateKeeper
- Resource
- Grid Resource
Walden: A Scalable Solution for Grid Account Management
**Problem Statement**

Many disparate, decentralized clusters
- ATLAS (15 nodes, 39 CPUs)
- CAC Hypnos Cluster (128 nodes, 256 processors)
- CAC Morpheus Cluster (67 nodes, 134 processors)
- CAC Nyx Cluster (132 nodes, 264 processors)
- MCBI CTAlliance Cluster (59 nodes, 118 processors)
- CCS G5 Cluster (24 nodes, 48 processors)

... and many more ...

[Image of cluster configurations]

---

http://www.mgrid.umich.edu  abose@eecs.umich.edu
Problem Statement

How do we securely authenticate, authorize and provide user access to grids across disparate administrative and geographical domains?

Globus GSI uses public key cryptography and digital signatures for secure communications and single sign-on.

University of Michigan provides Kerberos authentication for users.
Current Work in Grid Authorization

Grid authorization options

- PERMIS uses of X.509 Attribute Certificates
- PRIMA uses X.509 Attribute Certificates
- Shibboleth no built-in authorization engine; limited scope (web browser)
- VOMS uses X.509 Attribute Certificates
- CAS Community Authorization Service
- XACML powerful policy engine; requires custom PEP (Policy Enforcement Point) & PDP (Policy Decision Point)
Kerberos Authentication

Login Phase: Once Per Session

1. Alice → KDC
   “I am Alice”
2. KDC → Alice
   \( TGT = \{ \text{Alice, TGS, } K_{A,TGS}, K_{TGS}, T, K_{A,KCT} \} \)

Accessing Services: Every time a new/current kerberized service is requested

3. Alice → TGS
   Alice, Bob, TGT, \( \{ T \}_{K_{A,TGS}} \)
4. KCT → Alice
   \( TKT = \{ \text{Alice, Bob, } K_{A,B}, T, T\}_{K_{A,TGS,TGS}}, \{ K_{A,B} \}_{K_{A,TGS}} \)
5. Alice → Bob
   “I am Alice”, TKT, \( \{ T \}_{K_{A,B}} \)

TGS: Ticket Granting Service (often same entity as KDC)

- \( K_{A} \): Shared key between Alice and KDC (derived from Alice’s password upon login)
- \( K_{A,TGS} \): Session key for Alice and KDC
- \( K_{TGS} \): Shared key between KDC and TGS
- \( K_{A,B} \): Session key for Alice and Bob
- \( T \): Timestamp to prevent replay attacks (requires synchronized clocks)
KX.509 Certificates

The story so far... User has a Kerberos ticket on the workstation she is logged into. But Globus uses X.509 certificates — how does User use Globus-enabled services?

KX.509, developed at CITI, University of Michigan is a Kerberized client program (resides on local workstation) that generates an X.509 certificate and a private key based on the existing Kerberos ticket: both are normally stored in the same Kerberos ticket cache
the temporary <X.509 certificate, private key> are destroyed when Kerberos ticket expires

Therefore, by adopting KX.509, an Kerberos-based organization can deploy and use Globus-enabled services without changing its security infrastructure. Kerberos is the most widely deployed network authentication system currently in use.
Authorization Issues

Globus provides a static grid-mapfile for coarse-grained authorization.

Each grid-mapfile is locally maintained on each resource, mapping a user's X.509 DN to a local account.

Users either share local accounts, providing little accountability, or are granted unique local accounts, creating administrative problems.

How to provide fine-grained authorization with one-to-one user-account mapping?
Walden Authorization

- **Fine-Grained authorization** module based on XACML standard (XACML-based policy engine)
- Cluster owners have complete administrative control over who uses their resources
- **Policy files** define rules based on group membership, time of day, resource load, etc.
- Local account management is unnecessary
- **Group membership** can be assigned from one or several secure LDAP servers
Walden Authorization

Step 1: Obtain a Kerberos V Ticket Granting Ticket (TGT), which is then used to obtain and cache a KX.509 certificate.

Step 2: Submit a job request to Globus gatekeeper

Step 3: Gatekeeper invokes gridmap callout function, forwarding authorization request to Walden module.

Policy Enforcement Point (PEP) formats and sends request to Policy Decision Point (PDP).

PDP retrieves XACML policy (if necessary) from central policy repository
Walden Authorization

Step 4-5: **Policy Decision Point (PDP)** retrieves a 'bag of attributes' corresponding to user from secure LDAP server, and extensible to many other sources.

User attributes (e.g. Group Membership) is compared against authorization request.

PDP returns a response of Permit, Deny, or indeterminate, along with any obligations.
Walden Authorization

Step 6-7: **Policy Enforcement Point (PEP) parses response and obligations.**

If no defined obligations, PEP binds user to (permanent) local account from secure LDAP query.

If guest user obligation defined, PEP binds user to available guest account.
Walden Authorization

Step 8: If the user is authorized, the local account identity is returned to globus (otherwise, authorization is denied).

Step 9: The globus gatekeeper submits the authorized job request to the grid cluster, using the defined permanent or guest user account.
The story so far... Users are authenticated via Kerberos, a KX.509 credential is used by the authorization process, and globus is used to submit the job.

But what if the user doesn't want to install an entire globus client on their workstation?

But what if the user doesn't want to figure out the Globus Resource Specification Language (RSL)? Enter the MGRID portal...
MGRID Portal Architecture

MGRID Portal

Apache
- mod ssl
- mod kct
- mod kx509
- mod jp

Tomcat

CHEF

LDAP

Kerberos V5
- KCT
- KCA
- KDC

Kerberos
- kx509
- kinit

User Workstation
- Browser
- libpkcs11

Grid Resource
- GateKeeper
- Resource Mng
- Resource

LDAP
- Walden

SSL
- Client Certificate required

1

GSI

SASL

2

3

4

5

6

7
Walden provides...

Scalable solution that integrates with existing University of Michigan authentication
Secure authentication and authorization
Extensible XACML policy engine
Resource owners maintain administrative control over resources, optionally using existing Directory Services
Guest/Template user account management
Support for fluid Virtual Organizations
MARS: A Metascheduler for Distributed Resources in Campus Grids
WHY MÉTASCHEDULING?

Condor Pool → MetaScheduler → PBS → Sun Grid Engine

Grid Users:

http://www.mgrid.umich.edu  abose@eecs.umich
Metascheduling Advantages

Task Priorities: Allow On-Demand Scheduling

Examples: Satellite and radar data processing, Adaptive Simulations, Disaster Management (requires local resource preemption)

Optimize resource usage across many platforms

Campus-wide scheduling

Co-Scheduling of Resources

Computations, Data transfers, Network reservations, Sensors, Instruments

Integration of many local scheduling policies and frameworks

For the User: **Single Point of Job Submission!!** (GRID Portal)

http://www.mgrid.umich.edu  abose@eecs.umich
MARS DESIGN GOALS

Extensible Architecture

- Multiple standards for job description: JSDL, DRMAA, GRAAP
- Remote communications
- New scheduling algorithms can be easily incorporated

On-Demand Task Scheduling

- Resources on-demand (prioritized task queues and pre-emption of lower priority tasks)

Resource Usage Forecasting

- Can lead to better scheduling decisions across multiple systems
- MARS currently uses low-pass filters (exponential smoothing)
MGRID ARCHITECTURE

MGRID Portal

Apache
- mod ssl
- mod kct
- mod kx509
- mod jk
- mod php

Tomcat
- CHEF

LDAP
- Authorization

MARS Scheduler
- MARS Scheduler
- KCT
- KCA
- KDC
- GSI

User Workstation
- Browser
- kinit
- libpkcs11
- kx509

LDAP
- Authorization
- SASL
- Resource Mng
- Resource
- Grid Resource

Browser

SSL
- Client Certificate required

GSI
- Kerberos

KCT
- KCA
- KDC

Kerberos
- kinit

SASL
- GateKeeper

Authorization

http://www.mgrid.umich.edu
abose@eecs.umich.edu
MARS COMPONENTS

Task Resource Blocks (TRBs) allow task prioritization, global monitoring and indexing of tasks.

- Each task gets a TRB that includes a MARS JobID
- Individual schedulers assign their own JobIDs
- TRBs encapsulate these IDs
RESOURCE USAGE PREDICTION

Model transient behavior of local scheduler parameters from data collected by Resource Update Agents

Current parameters (compute resources only):

- CPU Utilization
  Maximum and average queue lengths of waiting tasks
  Maximum and average task turnaround times for queues

Use: Exponential Smoothing of time-series data of above parameters

\[
\overline{Q(t + \Delta t)} = \alpha Q(t) + (1 - \alpha) \overline{Q(t)}
\]

\( \overline{Q(t)}, \overline{Q(t + \Delta t)} \): smoothed observations
\( Q(t) \): actual observation
\( \alpha \): smoothing parameter (evaluated from LS fit to time series data)
SCHEDULING ALGORITHMS

Minimum Completion Time (MCT)

\[ e_{ij} = \text{expected execution time for task } t_i \text{ on resource } r_j \]

\[ s_{ij} = \text{estimated start time (depends on number of tasks in queue)} \]

\[ c_{ij} = \text{completion time} \]

Find resource \( r_k \) for task \( t_i \) find the minimum of all \( c_{ij} \) s:

\[
(t_i, r_k) = \min_{j=1,m} \left(c_{ij} = s_{ij} + e_{ij}\right)
\]

Easy to implement but not efficient as we will see later
Genetic Algorithm Scheduler

Allows metascheduler administrator to build schedules based upon complicated metrics

We modified a parallel GA solver (PGAPack) to implement

*Easy to implement new algorithms in the framework*
Opaque State as a parameter

Introduce new parameters to optimize
Architectural considerations
Network topology considerations
Administrator may supply custom fitness function that utilizes new parameters
OFFLINE WORKLOAD COMPARISONS

Parallel Workload on 700 CPUs

MCT is not desirable from resource owner's point of view!
OFFLINE WORKLOAD COMPARISONS

Serial Workload on 700 CPUs

Note difference in resource utilization between two workloads
## Offline Workload Comparisons

Comparison of maximum wait-times for any task

<table>
<thead>
<tr>
<th>Workload</th>
<th>Algorithm</th>
<th>Max Wait-time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serial</td>
<td>MCT</td>
<td>615828</td>
</tr>
<tr>
<td></td>
<td>GA</td>
<td>(613800*, 986400)</td>
</tr>
<tr>
<td>Parallel</td>
<td>MCT</td>
<td>1839621</td>
</tr>
<tr>
<td></td>
<td>GA</td>
<td>(1468800*, 2458800)</td>
</tr>
<tr>
<td>Benchmark</td>
<td>MCT</td>
<td>879445</td>
</tr>
<tr>
<td></td>
<td>GA</td>
<td>(739800*, 1150200)</td>
</tr>
</tbody>
</table>

* Minimum wait-time

GA: maximize resource utilization, MCT: minimize completion time

http://www.mgrid.umich.edu  abose@eecs.umich
ONLINE WORKLOAD COMPARISONS BETWEEN MARS AND PBS

Three months of submitted workload traces for a 120-CPU AMD Athlon cluster at NPACI/CAC (mixed workload)
ONLINE WORKLOAD COMPARISONS BETWEEN MARS AND PBS

Three months of submitted workload traces for a 120-CPU AMD Athlon cluster at NPACI/CAC (mixed workload)
Integration of Walden authorization callout and MARS plug-ins for LRM

Redesign of Ingress and Egress modules to accommodate general
resources such as compute, storage and network resources

Ongoing study of characterization of archived workloads at various
NPACI and Teragrid sites

Time-series modeling techniques such as ARMA processes for LRM
resource state predictions

Inter-domain scheduling by managing TRB queues on distributed
MARS agents

Received NSF Award in October, 2004 for fault-tolerant scheduling,
online workload characterization and resource-level prediction
Example Application:

CITI Network Testing And Performance (NTAP)
MGRID NTAP Project

NTAP: Network Testing and Performance
Purpose: Provide a secure and extensible network test and performance tool invocation service at U-M  (Can we drop code on routers?)
Service based on Globus
Runs on dedicated nodes attached to routers in a VLAN environment
MGRID NTAP Project

Initial work implemented a bandwidth reservation tool:

- Securely modifies network switch configurations
- Implements role-based authorization
- Includes scheduler for future reservations
- Based on GARA
  - General-purpose Architecture for Reservation and Allocation
- Layered on Globus
MGRID NTAP Project

Added modular fine grained authorization
  Keynote policy engine/AFS PTS group service
  PERMIS policy engine/LDAP group service
  Integrated with Walden Authorization Framework

Added signed group membership RSL payload

Generalized from bandwidth reservation to the ability to run arbitrary programs at a Grid service endpoint
  Designed to easily add functionality
  Network testing tools being run
    Iperf, traceroute, ping, owamp, etc
MGRID NTAP Project

Multihomed PMP support (policy among peers)
- One routing table per VLAN
- Routing policy selects routing table based on source address of outgoing packet
- Emulates a default route per virtual interface

Path discovery
- Use traceroute to obtain routing information
- Use network topology databases to map network segments to PMP pairs

http://www.mgrid.umich.edu  abose@eecs.umich.edu
MGRID NTAP Project

PERMIS authorization
  User, Target, Action
  Attribute, policy certificates
  Policy engine

Production hardening
  Error handling/recovery
  Cleanup/restart
  Log file management
  Deployment packaging
MGRID NTAP Project

Performance measurement
  Deployment to ITCom lab
Output Database
  Permanent, secure storage of results
  Searches and aggregations
  Throughput/latency matrix
Host Endpoint Testing
  The last mile segment
Secure download of signed binaries

http://www.mgrid.umich.edu  abose@eecs.umich.edu
MGRID NTAP Project

Campus Prototype:

Host A
192.168.10.1
192.168.10.50

R1
192.168.10.1
10.1.1.1/30

PMP 1
192.168.10.50

R2
10.1.1.2/30

PMP 2
192.168.20.50

Host B
192.168.20.99

192.168.10.1
10.1.1.1/30

192.168.20.50

10.1.1.2/30

192.168.20.99
MGRID: Ongoing and Planned Work

Campus-wide grid prototype based on MGRID software
- Goal is to have 1000 CPUs by Summer, 2005

Continue focus on applications (Biomedicine, Computational Chemistry, Parallel Monte Carlo/Computational Finance, Agent Simulations, Distributed Visualization)

New Projects in 2005:
- Fault-tolerant Grid Infrastructure
  - develop fault models (resources, grid protocols, data handling, AAA etc.)
  - fault-injection modules in MGRID components (MARS, Walden)
  - scheduling in presence of application- and resource-level faults
  - transparent VO-level checkpointing

Cross-Institutional (cross-domain) resource sharing and collaboration

http://www.mgrid.umich.edu  abose@eecs.umich.edu