

Towards a unified X.509-based Cloud Authorization

Steven Timm

Project leader, FermiCloud project

Ted Hesselroth

X509 developer, FermiCloud project

OGF32, Salt Lake City, July 16, 2011

Acknowledgements and disclaimers

- FermiCloud project staff: Ted Hesselroth, Faarooq Lowe, Dan Yocum, Keith Chadwick, Parag Mhashilkar, Tanya Levshina, Gabriele Garzoglio
- OpenNebula developers, in particular Tino Vasquez who has been our liaison.
- My first OGF—please be nice if I get some acronyms or technical terms wrong.
- Any talk on cloud software is obsolete as soon as it is written

Outline

- Brief description of FermiCloud project
- Fermilab and FermiCloud security requirements
- Review of existing cloud AuthN/AuthZ
- Current X.509 authentication deployment in OpenNebula
- Plans for X.509 authorization

The FermiCloud Project

- Infrastructure-as-a-service on demand
- Consolidated eight racks of legacy integration and developer machines into one rack
- Virtual machines are integrated into Fermilab site network, and open to the Internet.
- Phase 1: technology evaluation, requirements, pilot service—complete
- Phase 2: scale up service and make it production quality—in progress.
- Phase 3: high availability, redundancy, in planning.

Fermilab Security Enclaves

- General Computing Enclave
 - Normal login access
 - Strongly authenticated via Kerberos 5
 - Kerberos 5 authentication extended to grid via Fermilab Kerberos Certificate Authority (SLCS).
- Open Science Enclave
 - Running arbitrary jobs via the grid
 - X.509 authentication/authorization, any IGTF cert.
- Both Enclaves
 - Approved OS baseline, required patch levels
 - Automated vulnerability scanning

FermiCloud Security Requirements

- No stored secrets in virtual machine
- New virtual machines start in “network jail”, get patched and virus-scanned before getting on network
- Auto wake-up dormant machines for patches
- Interprocess communication between cloud daemons secure.

FermiCloud Authentication Requirements

Launching a virtual machine:

- The equivalent of running a grid job with arbitrary code
- X.509 or Kerberos 5 authentication required
- Implemented in practice with X.509
- Logging into the virtual machine:
 - Kerberos 5 authentication required
- Authorization—
 - Only pre-authorized users get to launch virtual machines. FermiCloud has to create your account.

Existing open-source cloud authentication

- EC2 SOAP API (Eucalyptus, OpenStack?, Nimbus)
 - Uses X.509 certs BUT
 - Self-signed
 - Passwordless private key
 - No idea of IGTF CA's, CRL's
 - Often distributed in insecure ways.
- EC2 Query API (Eucalyptus, OpenStack?, Nimbus, OpenNebula)
 - Can be wrapped with TLS wrapper but by default not https:
 - Uses access key/secret key pair
 - “secret key” has to be stored unencrypted in a file
- Nimbus WSRF interface full GSI authentication with grid-mapfile.
- OCCI?

FermiCloud Plan, X.509-based AuthN

- We chose OpenNebula for wide diversity of virtual machines it can make but despite its security and API limitations.
- Most 3rd-party tools use EC2 Query API so we have to make it work
- Use pluggable authentication features of OpenNebula to use internal X.509 authentication.
- “secret key” in user database-> X.509 DN.
- 3 components modified thus far: command line, “econe” query daemon, and SunStone GUI
- Patches contributed back to OpenNebula, in trunk, expected to be part of OpenNebula 3.0 release.
- Clients: HybridFox works without modification. Condor-G modifying EC2_GAHP to support, first tests worked.

X.509 Authentication Details

- For “econe” and “Sunstone”
 - X.509 certificate authenticated by Apache mod_ssl
 - DN of certificate passed to OpenNebula core for normal “password” check
 - Server creates login token valid for subsequent operations.
 - Ruby plugin for X.509 authentication used
- For CLI
 - Present certificate via X509_USER_CERT
 - Create authentication token via oneauth command
 - Ruby plugin for X.509 used.
 - Also created certificate-based login for the admin user.
- For OCCI
 - Haven't attempted yet but expect that the strategy used with “econe” would work.
- Currently requires cert/key, trying to make it work with delegated X.509 proxies as well.

Towards X.509-based Authorization

- Authorization interoperability protocol currently used in OSG, EGI grids among others. Fermilab was part of effort. (GFD159)
- Clients make XACML-based callout to authorization servers
- We want to implement VO and role-based authorization (and resource quotas) as we move towards a bigger cloud.
- Open-source cloud software needs to clean up its AuthZ code anyway, lots of little MySQL databases, all different.

Current AuthZ work

- Make Ruby bindings to the existing client callout routines.
- Determine XACML parameters that should be sent to the server. Anticipate they will be very similar to what is currently used for grid jobs.
- Determine what modifications to code are necessary to make OpenNebula, and other clouds.
- Rumor has it that OpenNebula 3.0 release will already include pluggable authorization as well as authentication, if so that will make it much easier.
- OpenNebula 3.0 also adding support for groups and quotas