KeyVOMS:
Managing Secure Cloud Federation

Dr. Craig A. Lee, Senior Scientist, lee@aero.org

N. Desai, A. Brethorst

The Aerospace Corporation

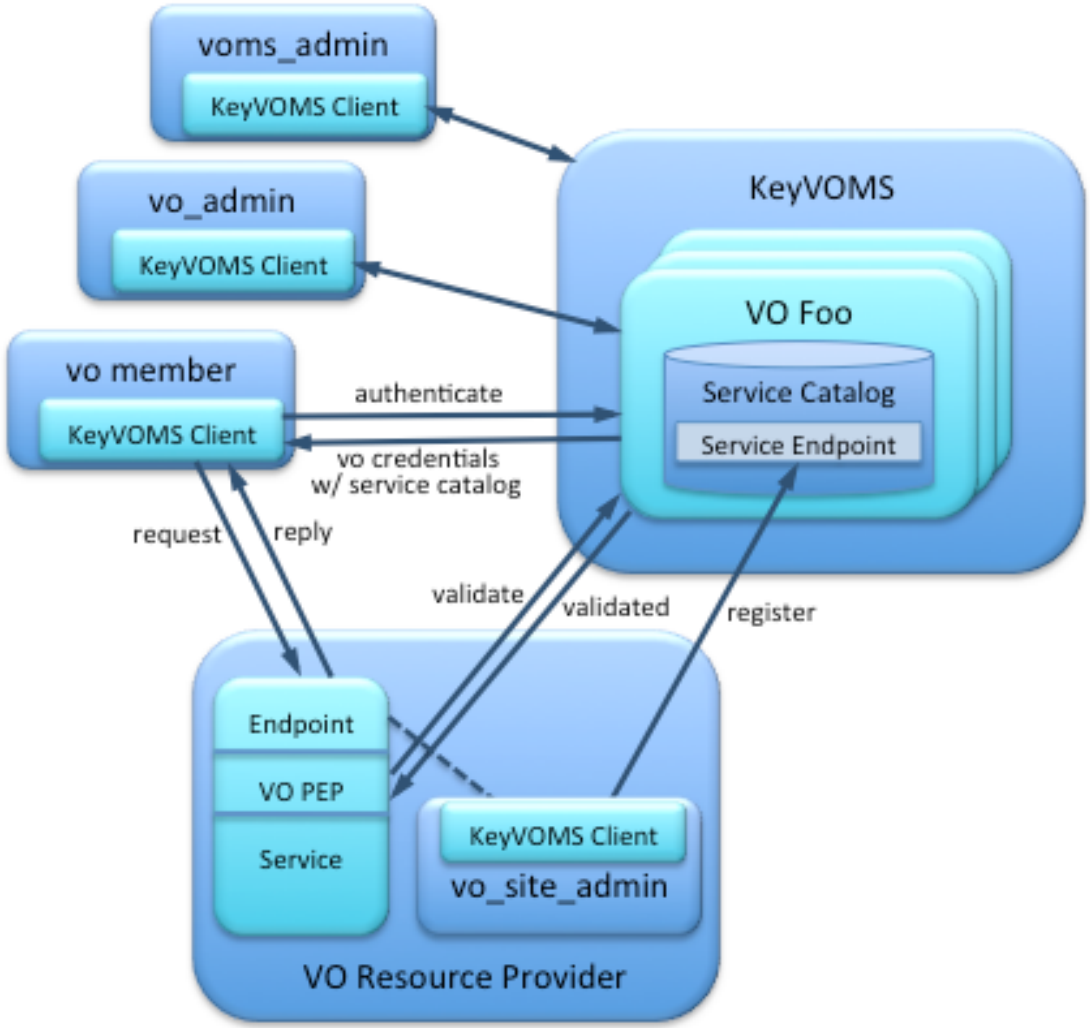
(a California nonprofit corporation that operates a federally funded research and development center)

Observations

- Different clouds will be deployed and used by different organizations
  - This is just a fact of life
- Eventually these organizations and agencies will need/want to cooperate, i.e., share data and resources
  - This will require cloud federation at both the infrastructure-level and the application-level
- "Poster Child" use case: International Disaster Response Efforts
- Cloud technology is fundamentally about *on-demand provisioning*
  - Servers, storage, communication, platforms, services, …
- Fundamental cloud technology says *nothing* about:
  - Federated Identity Management
  - Federated Resource Management
  - Single Sign-on
  - Delegation of Trust
  - Managing the Trust Ecosystem
- Such federation technologies must be added to the cloud stack!
  - A *Virtual Organization (VO)* is a security and collaboration context spanning multiple sites to address these issues

The KeyVOMS Architecture: Managing Secure Service Discovery and Access across Multiple Sites

- OpenStack Keystone v3 re-purposed as a stand-alone, VO Management Service: KeyVOMS
  - *Domain used as a VO*
  - Inherits support for PKI, certificate caching, revocation lists, etc.
- New rule set enforces three pre-defined roles:
  - voms_admin
  - vo_admin
  - vo_site_admin
- Modular VO Policy Enforcement Point built
  - Based on WSGI
- RSS Feeder is the first demo app



Target Architecture and KeyVOMS Comparison: Managing Objects

	VOs	Members	Attributes	Services	Endpoints
voms_admin	✓	✓	✓	✓	✓
vo_admin		within VO	within VO	within VO	within VO
vo_site_admin			list within VO	owned	owned
other vo members					

TABLE I
TARGET MODEL AUTHORIZATIONS TO MANAGE OBJECTS.

	Domains	Projects	Users	Roles	Services	Endpoints
voms_admin	✓	✓	✓	✓	✓	✓
vo_admin		✓	✓	owned	owned	owned
vo_site_admin		list			owned	owned
other vo members						

TABLE IV
KEYVOMS AUTHORIZATIONS TO MANAGE OBJECTS.

Target Architecture and KeyVOMS Comparison: Managing Object Associations

	User/Attribute	Attribute/Endpoint
voms_admin	✓	✓
vo_admin	within VO	within VO
vo_site_admin		attributes within VO with owned endpoints
other vo members		

TABLE II
TARGET MODEL AUTHORIZATIONS TO MANAGE OBJECT ASSOCIATIONS.

	User/Role	User/Project	Project/Endpoint
voms_admin	✓	✓	✓
vo_admin	grant, revoke of owned roles	✓	grant, revoke of owned endpoints
vo_site_admin			grant, revoke of owned endpoints
other vo members			

TABLE V
KEYVOMS AUTHORIZATIONS TO MANAGE OBJECT ASSOCIATIONS.

Target Architecture and KeyVOMS Comparison: Returned Service Catalog

	Catalog
voms_admin	complete
vo_admin	within VO
vo_site_admin	filtered
other vo members	filtered

TABLE III
TARGET MODEL SERVICE CATALOG RETURNED ON AUTHENTICATION.

	Catalog
voms_admin	complete
vo_admin	filtered
vo_site_admin	filtered
other vo members	filtered

TABLE VI
KEYVOMS SERVICE CATALOG RETURNED ON AUTHENTICATION.

Further Development Issues

- KeyVOMS addresses the issues of service discovery and access
  - Needs to be integrated with federated identity management on the front-end
- To realize the target architecture, roles, services and endpoints need to be domain-specific
  - Two flavors of roles, services and endpoints: system-wide and domain-specific?
- Without domain-specific roles, services and endpoints, clients must keep track of the roles, services and endpoints that they create and "own"
  - Creates potential consistency issues between client and server
- Maintaining role attributes on services and endpoints would enable simpler user-endpoint attribute matching for initial service discovery
  - Endpoint filtering is being used since it exists but adds complexity
  - Three object associations must be managed, rather than two
- Trust is required between resource providers and KeyVOMS
  - KeyVOMS is essentially a VO identity provider
- KeyVOMS is a centralized, third-party design
  - A distributed, P2P design is also possible
- KeyVOMS relies on a VO PEP being installed to protect each service
  - Needs to be as modular and easy-to-use as possible
- Ease of adoption is a driving issue
  - While "Same Thing Everywhere" is necessary to achieve the widest possible interoperability (e.g., TCP/IP, HTTP, etc.), it is a non-starter for near-term adoption
  - A *broker approach* offers much easier near-term adoption