Federated Access to US CyberInfrastructure
Jim Basney
jbasney@illinois.edu

This material is based upon work supported by the National Science Foundation under grant number 0943633. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.
CILogon Project Goal

- Enable campus logon to CyberInfrastructure (CI)
  - Use researchers’ existing security credentials at their home institution
  - Ease credential management for researchers and CI providers
InCommon is the federation for U.S. research and education, providing higher education and their commercial and non-profit partners with a common trust framework for access to online resources.

264 InCommon Participants

Almost 5 million end-users (faculty, staff, students)
A Roadmap for Using NSF Cyberinfrastructure with InCommon

A helpful guide for CI projects

http://incommon.org/nsfroadmap
Prior Work: go.teragrid.org

- Campus login to TeraGrid
- 35 campuses so far
- Relies on TeraGrid identity vetting
- In production since September 2009
- 1000+ certificates issued so far to 65+ users
- IGTF accredited
- Integration with portal.teragrid.org underway
- IDtrust 2010 paper: “Federated Login to TeraGrid” (http://middleware.internet2.edu/idtrust/2010/)
New Service: cilogon.org

- No TeraGrid account required
- Supports InCommon and OpenID authentication
- Delivers certificates to desktop, browser, and portals
- Available certificate lifetimes: from 1 hour to 13 months
- Supports close integration with CI projects
- Available now!
- FAQ: www.cilogon.org/faq
**CILogon Portal Delegation**

- Grid Portals and Science Gateways provide web interfaces to CI
  - Portals/Gateways need certificates to access CI on researchers’ behalf
- CILogon Delegation Service allows researchers to approve certificate issuance to portals (via OAuth)
- www.cilogon.org/portal-delegation
An OAuth Service for Issuing Certificates to Science Gateways for TeraGrid Users

Jim Basney and Jeff Gaynor
{jbasney,gaynor}@illinois.edu

National Center for Supercomputing Applications
University of Illinois at Urbana-Champaign

This material is based upon work supported by the National Science Foundation under grant number 0932251.
Goals

• Support use of *individual TeraGrid accounts* via gateways
  • Independent of support for gateway *community accounts*
  • For more accurate accounting, greater resource access

• Avoid disclosing TeraGrid user passwords to gateways
  • Avoid risk to long-lived credentials (i.e., user passwords)
  • Use TeraGrid passwords only on systems operated by TeraGrid

• Use standard security protocols: TLS, OAuth
  • More trustworthy
  • Ease of integration for gateway developers

http://security.ncsa.illinois.edu/teragrid-oauth/
Current Approach  ➔ New Approach

http://security.ncsa.illinois.edu/teragrid-oauth/
Benefits

• **Security WG** concerns about password disclosure to external science gateway sites are addressed

• **Science Gateways** can support individual TeraGrid account access via standard protocols

• **Resource Providers** can support user access via gateways using existing certificate-based interfaces

• **Users** can access their individual TeraGrid accounts via gateways using their TeraGrid Portal login

http://security.ncsa.illinois.edu/teragrid-oauth/
OAuth Example

Web User (Resource Owner)

1 Request Access to Photos
2 Authenticate & Grant Access to Photos
3 Token

Photo Printing Service (Client)

4 Token
5 Token

Photo Sharing Service (Server)

6 Photos

http://security.ncsa.illinois.edu/teragrid-oauth/
The diagram illustrates the flow of a process involving OAuth2.0 and MyProxy. Here is a breakdown of the process:

1. **User's browser** initiates the process by making a request to the **OAuth client**.
2. The **OAuth client** responds with a **temp_token**.
3. The **user** then proceeds to authenticate and approve the request.
4. The **OAuth client** then requests the **MyProxy username and password**.
5. The **OAuth client** uses the received tokens to obtain an **access_token**.
6. Finally, the **OAuth client** uses the **access_token** to make a request to the **MyProxy server**.

The diagram includes several actions and tokens, such as:

This process ensures secure and efficient communication between different systems.
Distributed Web Security for Science Gateways

Jim Basney (NCSA)
Rion Dooley (TACC)
Jeff Gaynor (NCSA)
Suresh Marru (IU)
Marlon Pierce (IU)

This material is based upon work supported by the National Science Foundation under grant number 1127210.
Science Gateway Security Project

• Primary Deliverable: A standards-compliant OAuth service implementation to securely delegate, deliver, and renew credentials to science gateways on a user's behalf.
  • Including optional MyProxy integration
  • Including client libraries and modules for web frameworks

• Timeline:
  • August 2011: Project Start
  • February 2012: Initial MyProxy OAuth release
  • August 2012: Initial release of general software components
  • August 2013: Feature complete software releases
  • August 2014: Final software releases
Current Approach  New Approach

1. Web Browser
   - MyProxy password
2. Science Gateway
   - MyProxy password
3. MyProxy Server
   - certificate
4. Grid Service
   - access using certificate

1. Web Browser
   - request certificate
2. Science Gateway
   - OAuth token
3. MyProxy OAuth Front-End
   - OAuth token
4. MyProxy Server (unmodified)
   - certificate
5. Grid Service
   - access using certificate
Certificate Delegation via OAuth (Option A)

1. MyProxy password
2. OAuth authorization code
3. OAuth authorization code + client credentials
4. OAuth access token
5. OAuth access token + certificate request
6. certificate
7. access using certificate
8. OAuth access token
9. OAuth access token + certificate request
10. certificate
11. access using certificate

Web Browser

OAuth Authorization Server

MyProxy Server (unmodified)

OAuth Resource Server

Portal

OAuth Client

GFAC

Grid Service

Grid Service

www.sciencegatewaysecurity.org
Certificate Delegation via OAuth (Option B)

4. OAuth Authorization Server responds with OAuth access token.
5. Web Browser requests certificate with OAuth access token + certificate request.
6. OAuth Resource Server responds with certificate.
7. Grid Service requests access using certificate.
8. OAuth Authorization Server responds with OAuth client credentials.
10. OAuth Resource Server responds with OAuth access token + certificate request.
11. OAuth Resource Server responds with proxy certificate.
Integration with External Authentication

**LDAP/Kerberos**

1. Browser authentication
2. Validate Authentication
3. OAuth authorization code

**SAML/OpenID**

1. Authenticate
2. Authentication Assertion
3. OAuth authorization code

Identity Provider

OAuth Authorization Server

www.sciencegatewaysecurity.org
Science Gateway Security Project

- Other planned OAuth deliverables
  - Secure access to gateway REST services
    - Authorizing access to services via OAuth tokens instead of certs
  - Certificate renewal
    - Using OAuth refresh tokens
- Community engagement
  - UltraScan, iPlant, GridChem/ParamChem
  - XSEDE, Globus Online

www.sciencegatewaysecurity.org