Building the Modern Research Data Portal using the Globus Platform

Steve, Vas, Greg tuecke@globus.org

Presentation material: www.globusworld.org/tutorials
Thank you to our sponsors!

U.S. Department of Energy

The University of Chicago

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

Argonne National Laboratory

powered by Amazon Web Services
Presentation material available at

https://www.globusworld.org/admin-tutorial
Cloud has transformed how software and platforms are delivered

**Software as a service:** **SaaS**
(web & mobile apps)

- Tripit
- Netflix
- Salesforce

**Platform as a service:** **PaaS**

- Amazon Web Services
- Microsoft Azure
- force.com

**Infrastructure as a service:** **IaaS**

- Amazon Web Services
- Microsoft Azure
- Google Compute Engine

PaaS enables more rapid, cheap, and scalable delivery of powerful (SaaS) apps
Research data management simplified.

Researchers
Focus on your research, not IT problems. We make it easy to move, manage, and share big data.

Learn More ➤
Get Started ➤

Resource Providers
Globus gives you more control over your data infrastructure, while providing excellent ease-of-use for your researchers.

Learn More ➤
Globus Subscriptions ➤

Our Users
Researchers and resource providers are our greatest inspiration and we love it when they say nice things about Globus.

User Quotes ➤
Case Studies ➤

Fast, Reliable, Secure File Transfer
Move files between your laptop, lab server, research computing center, national supercomputing facility, or any other storage system, using just a browser.

Learn More About File Transfer with Globus ➤

UPCOMING EVENTS
September 25, 2016 to September 28, 2016
Internet2 Tech Exchange 2016
Miami, FL
October 13, 2016 to October 15, 2016
...
Globus SaaS: Research data lifecycle

1. Researcher initiates transfer request; or requested automatically by script, science gateway.

2. Globus transfers files reliably, securely.

3. Researcher selects files to share, selects user or group, and sets access permissions.

4. Globus controls access to shared files on existing storage; no need to move files to cloud storage!

5. Collaborator logs in to Globus and accesses shared files; no local account required; download via Globus.

6. Researcher assembles data set; describes it using metadata (Dublin core and domain-specific).

7. Curator reviews and approves; data set published on campus or other system.

8. Peers, collaborators search and discover datasets; transfer and share using Globus.

- Only a Web browser required
- Use storage system of your choice
- Access using your campus credentials

Personal Computer
Publication Repository
Transfer
Discover
Publish
Share
Compute Facility
Instrument
Globus transfers files reliably, securely
Collaborator logs in to Globus and accesses shared files; no local account required; download via Globus
Researcher selects files to share, selects user or group, and sets access permissions
Researcher assembles data set; describes it using metadata (Dublin core and domain-specific)
Curator reviews and approves; data set published on campus or other system
Globus controls access to shared files on existing storage; no need to move files to cloud storage!
Globus transfers files reliably, securely
Globus SaaS Demo

• Logging into Globus with any identity
• Endpoint search
• Transfer
• HTTPS access
• Sharing with any identity
• Management Console
Platform Questions

• How do you leverage Globus services in your own applications?

• How do you extend Globus with your own services?

• How do we empower the research community to create an integrated ecosystem of services and applications?
**Example: NCAR RDA**

**NCEP Climate Forecast System Version 2 (CFSv2) Monthly Products**

ds094.2

For assistance, contact Bob Dattore (303-497-1825).

Mouse over the table headings for detailed descriptions.

<table>
<thead>
<tr>
<th>Data Description</th>
<th>Data File Downloads</th>
<th>Customizable Data Requests</th>
<th>Other Access Methods</th>
<th>NCAR-Only Access</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Union of Available Products</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Diurnal monthly means</strong></td>
<td>Web File Listing</td>
<td>Request Access Invitation</td>
<td>Get a Subset</td>
<td>GLADE File Listing</td>
</tr>
<tr>
<td><strong>Regular monthly means</strong></td>
<td>Web File Listing</td>
<td>Get a Subset</td>
<td>TDS Access</td>
<td>GLADE File Listing</td>
</tr>
<tr>
<td><strong>Selected Parameter/Level Time Series</strong></td>
<td>Web File Listing</td>
<td>Get a Subset</td>
<td>TDS Access</td>
<td>GLADE File Listing</td>
</tr>
</tbody>
</table>
Globus PaaS

Globus APIs

Data Publication & Discovery

File Sharing

File Transfer & Replication

Auth & Groups

Globus Toolkit

Globus Connect
Demo

Sample
Research Data Portal
Prototypical research data portal

- Identity Provider
- Globus Web Helper Pages
- Globus Auth
- Portal Web Server (Client)
- Portal Endpoint
- Other Endpoints
- Other Services
- Globus Cloud
- GridFTP
- REST
- HTTPS

Browser
Applications
Desktop
User's Endpoint (optional)
Firewall
Science DMZ

Login

Globus Auth

Other Services

Portal Web Server (Client)

Globus Cloud

Globus Transfer
Prototypical research data portal

- **Identity Provider**
- **Globus Web Helper Pages**
- **Globus Auth**
- **Portal Web Server (Client)**
- **Globus Transfer**
- **Other Services**
- **Other Endpoints**

**Network Protocols and Technologies**
- HTTPS
- REST
- GridFTP

**System Components**
- **Browser**
- **Applications**
- **Desktop**
- **User’s Endpoint** (optional)
- **Portal Endpoint**
- **Firewall**
- **Science DMZ**
Introduction to REST APIs

• **Remote operations on resources via HTTPS**
  – POST ~= Create (or other operations)
  – GET ~= Read
  – PUT ~= Update
  – DELETE ~= Delete

• **Globus APIs use JSON for documents and resource representations**

• **Resource named by URL**
  – Query params allow refinement (e.g., subset of fields)

• **Requests authorized via OAuth2 access token**
  – Authorization: Bearer asdflkqhqhafsdafeawk
Globus Transfer API

• Nearly all Globus Web App functionality implemented via public Transfer API
  
  docs.globus.org/api/api/transfer

• Fairly stable, but small changes coming
  – Deprecation policy
Globus Python SDK

- Python client library for the Globus Auth and Transfer REST APIs

  globus.github.io/globus-sdk-python

- Public beta, likely to change some
TransferClient class

- `globus_sdk.TransferClient` class

  ```python
  from globus_sdk import TransferClient
  tc = TransferClient()
  ```

- Handles connection management, security, framing, marshaling
TransferClient low-level calls

- Thin wrapper around REST API
  - post(), get(), update(), delete()

get(path, params=None, headers=None, auth=None, response_class=None)
  - path – path for the request, with or without leading slash
  - params – dict to be encoded as a query string
  - headers – dict of HTTP headers to add to the request
  - response_class – class for response object, overrides the client’s default_response_class
  - Returns: GlobusHTTPResponse object
TransferClient higher-level calls

- One method for each API resource and HTTP verb
- Largely direct mapping to REST API

endpoint_search(filter_fulltext=None, filter_scope=None, num_results=25, **params)
Python SDK Jupyter notebook

• Jupyter (iPython) notebook demonstrating use of Python SDK

github.com/globus/globus-jupyter-notebooks

• Overview
• Open source, enjoy
Walk-through

Jupyter Notebook
Endpoint Search

• Plain text search for endpoint
  – Searches owner, display name, keywords, description, organization, department
  – Full word and prefix match

• Limit search to pre-defined scopes
  – all, my-endpoints, recently-used, in-use, shared-by-me, shared-with-me

• Returns: List of endpoint documents
Endpoint Management

- Get endpoint (by id)
- Update endpoint
- Create & delete (shared) endpoints
- Manage endpoint servers
Endpoint Activation

- Activating endpoint means binding a credential to an endpoint for login
- Globus Connect Server endpoint that have MyProxy or MyProxy OAuth identity provider require login via web
- Auto-activate
  - Globus Connect Personal and shared endpoints use Globus-provided credential
  - An endpoint that shares an identity provider with another activated endpoint will use credential
- Must auto-activate before any API calls to endpoints
File operations

- List directory contents (ls)
- Make directory (mkdir)
- Rename

Note:
- Path encoding & UTF gotchas
- Don’t forget to auto-activate first
Task submission

- Asynchronous operations
  - Transfer
    - Sync level option
  - Delete
- Get submission_id, followed by submit
  - Once and only once submission
Task management

• Get task by id
• Get task_list
• Update task by id (label, deadline)
• Cancel task by id
• Get event list for task
• Get task pause info
Bookmarks

- Get list of bookmarks
- Create bookmark
- Get bookmark by id
- Update bookmark
- Delete bookmark by id

- Cannot perform other operations directly on bookmarks
  - Requires client-side resolution
Shared endpoint access rules (ACLs)

• Access manager role required to manage permission/ACLs

• Operations:
  – Get list of access rules
  – Get access rule by id
  – Create access rule
  – Update access rule
  – Delete access rule
Management API

• Allow endpoint administrators to monitor and manage all tasks with endpoint
  – Task API is essentially the same as for users
  – Information limited to what they could see locally

• Cancel tasks

• Pause rules
Exercise: Jupyter notebook

Install Jupyter notebook either locally or on EC2 instance

github.com/globus/globus-jupyter-notebooks.git

Modify Jupyter notebook to:

1. Find the endpoint id for XSEDE Comet
2. Set all the metadata fields on your shared endpoint
3. Set permissions to allow your neighbor to access your shared endpoint
4. Transfer all files *.txt from the tourexercise directory on the Globus Vault endpoint to any other endpoint.
5. Monitor for completion, and monitor the event log
6. Perform an ‘ls’ given a bookmark name
7. Perform a transfer akin to ‘rsync –av –delete’
8. Anything else you want to try out...
Prototypical research data portal

- **Browser**
- **Applications**
- **Desktop**
  - User’s Endpoint (optional)
- **Portal Web Server (Client)**
- **Identity Provider**
- **Globus Web Helper Pages**
- **Globus Auth**
- **Globus Transfer**
- **Other Services**
- **Other Endpoints**
- **Portal Endpoint**
- **Firewall**
- **Science DMZ**
- **GridFTP**
- **HTTPS**
- **REST**
Maximizing the value of the Science DMZ
Prototypical research data portal

- Identity Provider
- Globus Web Helper Pages
- Globus Auth
- Portal Web Server (Client)
- Globus Transfer
- Other Services
- REST
- GridFTP
- Other Endpoints
- Science DMZ
- Portal Endpoint
- Other
- Desktop
- Applications
- Browser
- User’s Endpoint (optional)
- Firewall
- HTTPS
HTTPS to Endpoints

• Each endpoint HTTPS server is a Globus Auth service (resource server)

• Web page can link to file on server
  – Browser GET will cause HTTPS server to authorize request via Globus Auth (note SSO)

• Portal (client) can request scope for endpoint resource server
  – Use access token in requests
“A single global information space”
Prototypical research data portal

- Identity Provider
- Globus Web Helper Pages
- Globus Auth
- Portal Web Server (Client)
- Globus Transfer
- Other Services
- Other Endpoints
- GridFTP
- REST
- HTTPS
- Browser
- Applications
- Desktop
- User’s Endpoint (optional)
- Portal Endpoint
- Science DMZ
- Firewall
- Globus Cloud
- Prototypical research data portal

Applications

User's Endpoint (optional)

Portal Web Server (Client)

Globus Auth

Globus Cloud

Identity Provider

HTTPS

Globus Web Helper Pages
Globus PaaS

- Data Publication & Discovery
- File Sharing
- File Transfer & Replication
- Auth & Groups
- Globus Toolkit
Challenge

• **How to provide:**
  - Login to apps
    - Web, mobile, desktop, command line
  - Protect all REST API communications
    - App → Globus service
    - App → non-Globus service
    - Service → service

• **While:**
  - Not introducing even more identities
  - Providing least privileges security model
  - Being agnostic to programming language and framework
  - Being web friendly
  - Making it easy for users and developers
Globus Auth

- Foundational identity and access management (IAM) platform service
- Simplify creation and integration of advanced apps and services
- Brokers authentication and authorization interactions between:
  - end-users
  - identity providers: InCommon, XSEDE, Google, portals
  - services: resource servers with REST APIs
  - apps: web, mobile, desktop, command line clients
  - services acting as clients to other services
Globus Auth

• Identity and access management PaaS

docs.globus.org/api/auth

• Introduction
• Developer Guide
• Reference
Based on widely used web standards

• OAuth 2.0 Authorization Framework
  – aka OAuth2

• OpenID Connect Core 1.0
  – aka OIDC

• Use various OAuth2 and OIDC libraries
  – Google OAuth Client Libraries (Java, Python, etc.), Apache mod_auth_openidc, etc.
  – Globus Python SDK
Globus account

• A Globus account is a set of identities
  – A *primary identity*
    o Identity can be primary of only one account
  – One or more *linked identities*
    o Identity can (currently) be linked to only one account

• **Account does not have own identifier**
  – An account is uniquely identified using its primary identity
Globus Auth interactions

Globus Auth interactions

1. Request authorization

- For a set of scopes
  - Login: openid, email, profile
  - HTTPS/REST APIs
- User selects identity provider
Globus Auth interactions

1. Request authorization
2. Authenticates a resource owner

- Using existing identities
  - XSEDE, University (via InCommon), Google, web app, etc.
- User can link multiple identities into a single Globus Account
- No Globus username (Globus ID) required
- Globus Auth handles naming details, e.g., ePPN vs ePTID
Globus Auth interactions

1. Request authorization
2. Authenticates a resource owner
3. Obtains authorization (consent) for a client to access a resource

- Resource is provided by a resource server
- Limited by a scope
Globus Auth Auth interactions

1. Request authorization
2. Authenticates a resource owner
3. Obtains authorization (consent) for a client to access a resource
4. Issues OAuth2 access_token to client

• Some grant types issue authorization code, which client exchanges for access token
• Access token is opaque to client
• May include a refresh token, for offline access
1. Request authorization
2. Authenticates a resource owner
3. Obtains authorization (consent) for a client to access a resource
4. Issues OAuth2 access_token to client
5. May issue OIDC id_token to client with resource owner identity

JWT id_token:
- sub: Globus Auth identity id
- iss: https://auth.globus.org
- name: full name
- preferred_username:
- e.g., tuecke@uchicago.edu
- email: email contact
- other standard OIDC claims
Globus Auth interactions

1. Request authorization
2. Authenticates a resource owner
3. Obtains authorization (consent) for a client to access a resource
4. Issues OAuth2 access_token to client
5. May issue OIDC id_token to client with resource owner identity
6. HTTPS/REST call with access_token
Globus Auth interactions

1. Request authorization
2. Authenticates a resource owner
3. Obtains authorization (consent) for a client to access a resource
4. Issues OAuth2 access_token to client
5. May issue OIDC id_token to client with resource owner identity
6. HTTPS/REST call with access_token
7. Validates access_token for resource server, gets additional info

RFC 7662: OAuth 2.0 Token Introspection response:
• active: true or false
• client_id
• scope
• sub: Globus Auth identity id
• username: user@myu.edu
• identity_set: linked identities
• email
• name
• other standard claims
Globus Auth interactions

1. Request authorization
2. Authenticates a resource owner
3. Obtains authorization (consent) for a client to access a resource
4. Issues OAuth2 access_token to client
5. May issue OIDC id_token to client with resource owner identity
6. HTTPS/REST call with access_token
7. Validates access_token for resource server, gets additional info
8. Issues dependent access tokens to resource server

• Allows resource server to act as client to other resource servers
• Service uses request access_token to get a dependent access_token for each dependent service
• Service acts as client to its dependent services
Sample Research Data Portal

- UChicago Identity Provider
- Globus Web Helper Pages
- Globus Auth
- Globus Transfer
- Demo Portal (Client)
- Graph Services
- GridFTP
- UChicago Midway Endpoints

Diagram:
- Browser
- Applications
- Desktop
- My Laptop
- Firewall
- Portal Endpoint
- Science DMZ
- HTTPS
- REST
Use case: Log in with Globus

- Similar to: “Log in with Google” “Log in with Facebook”
- Using existing identities
- Providing access to community services
Demo

Jetstream App use of Globus Auth
1. Access portal

2. Redirects user

3. User authenticates and consents

4. Authorization token

5. Authenticate using client id and secret, send authorization code

6. Access tokens

7. Authenticate with access tokens to invoke transfer service as user

Authorization Code Grant

Browser (User)

Modern Research Data Portal

Portal (Client)

Globus Auth (Authorization Server)

Globus Transfer (Resource Server)
Sample Research Data Portal

- UChicago Identity Provider
- Globus Web Helper Pages
- Globus Auth
- Globus Transfer
- Demo Portal (Client)
- Graph Services
- GridFTP
- Science DMZ
- UChicago Midway Endpoints

Diagram:
- Login: Browser ➔ Applications ➔ Desktop ➔ My Laptop ➔ Portal Endpoint
- HTTPS ➔ UChicago Identity Provider ➔ Firewall
- REST ➔ Demo Portal (Client) ➔ Graph Services
- GridFTP ➔ UChicago Midway Endpoints
Use case: Portal calling services on user’s behalf

- **Examples:**
  - Portal starting transfer for user

- **Authorization Code Grant**
  - With service scopes
  - Can also request OIDC scopes

- **Confidential client**

- **Globus SDK:**
  - To get tokens: ConfidentialAppAuthClient
  - To use tokens: AccessTokenAuthorizer
Scopes

• APIs that client is requesting access to

• Scope syntax:
  – OpenID Connect: openid, email, profile
  – urn:globus:auth:scope:<service-name>:<scope-name>

• If client requests multiple scopes
  – Token response has tokens for first scope
  – other_tokens field in response has list of token responses for other scopes
  – Client must use correct token with each request
Consent

• Resource owner authorization that a client can request access to a service scope on the resource owner's behalf within a limited scope
  – If service has dependent scopes, they are part of the consent

• User can rescind a consent at any time
  – Invalidates all access, dependent, and refresh tokens originating from the client
Identity id vs. username

• **Identity id:**
  – Guaranteed unique among all Globus Auth identities, and will never be reused
  – UUID
  – Always use this to refer to an identity

• **Identity username:**
  – Unique at any point in time
    o May change, may be re-used
  – Case-insensitive user@domain
  – Can map to/from id, for user experience

• **Auth API allows mapping back and forth**
Effective identity

• App or service can choose to operate only with identities from a particular identity provider
  – Globus Auth login will require an identity from that provider to be linked to user’s account
  – OIDC id_token uses this “effective identity”

• If app or service does not set an effective identity policy, then the primary identity of the account is used as the effective identity for that app
Branding

- Can skin Globus Auth pages

Header

Text

Default IdP
App registration

• Client_id and client_secret for service
• App display name
• Declare required scopes
  – Need long-term, offline refresh tokens?
  – May require authorization from scope admin
• OAuth2 redirect URIs
• Links for terms of service & privacy policy
• Effective identity policy (optional)

developers.globus.org
Sample Research Data Portal

Demo: Install and Register
Code walk through
Exercise: Install sample data portal

- Install sample data portal
  - either locally or on EC2 instance

  \[\text{github.com/globus/globus-sample-data-portal.git}\]

- Register your application at:
  \[\text{developers.globus.org}\]

- Instructions in the README file
Exercises: In the Portal App, find and print to console:

- Globus Auth URL the portal redirects to for login
- Globus Auth URL the portal redirects to for logout
- Username of the logged in user
- Complete id_token of the logged in user
- URL of the Globus Browse Endpoints helper page used by the portal
- Endpoint and path selected by user as destination of the transfer
- URL to submit transfer, and resulting task id
- Complete task document returned by status
Prototypical research data portal

- Identity Provider
- Globus Web Helper Pages
- Globus Auth
- Portal Web Server (Client)
- Portal Endpoint
- Other Endpoints
- Other Services
- GridFTP
- REST
- HTTPS
- Science DMZ
- Firewall
- Desktop
- Applications
- User’s Endpoint (optional)
- Browser
- Login
- Globus Cloud
Use case: Native apps

• **Examples**
  – Command line, desktop apps
  – Mobile apps
  – Jupyter notebooks
  – Any client that cannot keep a secret (downloaded)

• **Native app is registered with Globus Auth**
  – Not a confidential client

• **Native App Grant is used**
  – Variation on the Authorization Code Grant

• **Globus SDK:**
  – To get tokens: NativeAppAuthClient
  – To use tokens: AccessTokenAuthorizer
1. Run application

2. URL to authenticate

3. Authenticate and consent

4. Auth code

5. Register auth code

6. Exchange code

7. Access tokens

8. Authenticate with access tokens to invoke transfer service as user

Native App (Client)

Globus Transfer (Resource Server)

Globus Auth (Authorization Server)
Mobile apps

• **Globus Auth supports mobile apps**
  – “Log in with Globus” in mobile apps
    o RFC 7636: Proof Key for Code Exchange by OAuth Public Clients (PKCE, pronounced “pixy”)
    o Extension to OAuth2 to allow OAuth2 Authorization Code Grant to work from mobile apps
  – Uses mobile browser for web-based login
  – Mobile apps can call any service REST APIs that use Globus Auth
  – iOS and Android
  – Same approach as used by Google, Facebook, etc.
Desktop & command line apps

- Globus Auth “Native App” PKCE support
- Use browser if possible
  - “OAuth 2.0 for Native Apps”
    - draft-ietf-oauth-native-apps-02
    - Use external browser if possible
    - Embed browser in app
    - Embed mini web server in app
- Allows copy-n-paste of authorization code
  - A little like app passwords, but OAuth2 compliant
- Globus Python SDK and CLI support Native App login
- Limited support for username/password authentication
  - Not recommended
Use case: Apps that need access tokens for long time

• **Examples:**
  – Portal checks for transfer status when user is not logged in
  – Run command line app from script

• **App requests refresh tokens**

• **Globus SDK:**
  – To get token: ConfidentialAppClient or NativeAppClient
  – To use tokens: RefreshTokenAuthorizer
1. Run application

2. URL to authenticate

3. Authenticate and consent

4. Auth code

5. Register auth code

6. Exchange code, request refresh tokens

7. Access tokens and refresh tokens

8. Store refresh tokens

9. Exchange refresh token for new access tokens

10. Access tokens

11. Authenticate with access tokens to invoke transfer service as user
Refresh tokens

- **For “offline services”**
  - E.g., Globus transfer service working on your behalf even when you are offline

- **Refresh tokens issued to a particular client for use with a particular scope**

- **Client uses refresh token to get access token**
  - Confidential client: client_id and client_secret required
  - Native app: client_secret not required

- **Refresh token good for 6 months after last use**

- **Consent rescindment revokes resource token**
Exercise: Native App

https://github.com/globus/native-app-examples

• README for install instructions
• ./example_copy_paste.py
  – Copy paste code to the app
• ./example_local_server.py
  – Local server to get the code
• ./example_copy_paste_refresh_token.py
  – Stores refresh token locally, uses it to get new access tokens
Use case: App invoking services as itself

• **Examples**
  – Sample portal invoking graph service and accessing endpoints as itself
  – Robots, agents, services

• **App registers with Globus to get client id and secret**
  – Native app cannot do this, because no client_secret

• **Client Credential Grant is used**

• **Globus SDK:**
  – To get tokens: ConfidentialAppAuthClient
  – To use tokens: AccessTokenAuthorizer
1. Authenticate with portal client id and secret

2. Access Tokens

3. Authenticate as portal with access tokens to invoke service

Modern Research Data Portal

Portal (Client)

Globus Auth (Authorization Server)

Globus Transfer (Resource Server)
User identity vs. portal identity

• User logging into portal results in portal having user’s identity and access token
  – Used to make requests on the user’s behalf

• Portal may also need its own identity
  – Access and refresh tokens for this identity
  – Used to make requests on its own behalf
Client identity

- Portal App has client_id & client_secret
- Globus Auth client_id is an identity_id
  - <client_id>@clients.auth.globus.org
- Use OAuth2 Client Credentials Grant to authenticate the client identity
  - Using client_id and client_secret
- Can use the client_id just like any other identity_id
  - Sharing access manager role, permissions, group membership, etc.
Exercise: Using Client credential grant

- Start with native app examples
- Register a new app to get client id and secret
- **Globus SDK:**
  - ConfidentialClientApp
  - AccessTokenAuthorizer
- **Using the Globus webapp:**
  - Create a shared endpoint
  - Set Access Manager role for the new client id
- **List files on the shared endpoint as the client identity**
- **Change permissions on the shared endpoint as the client identity**
- **Hint:** Look at Jupyter notebook for SDK calls for the transfer operations
Prototypical research data portal

Browser

Applications

Desktop

Login

Globus Web Helper Pages

Identity Provider

Globus Auth

Portal Web Server (Client)

Portal Endpoint

User’s Endpoint (optional)

Firewall

Identity Provider

Globus Web Helper Pages

Globus Auth

Portal Web Server (Client)

Other Endpoints

Globus Cloud

Globus Transfer

Other Services

Globus Cloud

Globus Transfer

Other Services

GridFTP

REST

HTTPS

Science DMZ
Globus Helper Pages

- Globus provided web pages designed for use by your web apps
  - Browse Endpoint
  - Activate Endpoint
  - Select Group
  - Manage Identities
  - Manage Consents
  - Logout

[docs.globus.org/api/helper-pages]
Client Logout

• Call token revocation on access tokens
  – https://auth.globus.org/v2/oauth2/token/revoke
  – Doc: docs.globus.org/api/auth/reference
  – Note: Does not revoke dependent tokens

• Delete access tokens

• Redirect to logout helper page
  – https://auth.globus.org/v2/web/logout
  – Doc: docs.globus.org/api/helper-pages
Prototypical research data portal

- Identity Provider
  - Globus Web Helper Pages
  - Globus Auth
  - Portal Web Server (Client)
    - Portal Endpoint
      - Other Endpoints
  - Other Services
    - GridFTP
        - Other
- Science DMZ
  - Desktop
    - User’s Endpoint (optional)
  - Firewall
    - HTTPS
- Browser
  - Applications
    - Login
      - User’s Endpoint (optional)
Adding your identity provider

• InCommon identity providers that release research & scholarship attributes to CILogon *(free)*

• Any other OpenID Connect identity provider *(subscription)*
Adding an identity provider

• If your portal has identities already:
  – Deploy OIDC server in front of it
    o Globus Python OIDC (coming soon)
    o Any standard OIDC server should work
    o Requires claim that can map to username
    o Optional claims: name, email, organization
  – Can register apps and services with an effective identity policy
    o Requires account to have identity from your identity provider when logging into your app
Portal accounts

• Your app portal can still have portal accounts for users
• Tie portal account to Globus account identity, rather than username/password
• Associate your profile with this account
• Globus Auth handles authentication of that identity, in order to log user into your portal account
Prototypical research data portal

- Identity Provider
- Globus Web Helper Pages
- Globus Auth
- Portal Web Server (Client)
- Portal Endpoint
- User’s Endpoint (optional)

Connections:
- HTTPS
- REST
- GridFTP

Endpoints:
- Firewall
- Science DMZ
- Other Endpoints
- Other Services

Components:
- Browser
- Applications
- Desktop
- Login

Services:
- Globus Cloud
- Other
- GridFTP
- REST
- HTTPS

Network Zones:
- Prototypical research data portal
- Science DMZ
- Other

Syntax Elements:
- Identity Provider
- Firewall
- Globus Cloud
- Prototypical research data portal
- HTTPS
- GridFTP
- REST
- Login
- Applications
- Portal Web Server (Client)
- Portal Endpoint
- User’s Endpoint (optional)

System Configurations:
- Prototypical research data portal
- Science DMZ
- Other Endpoints
- Other Services
- Other
- HTTPS
- GridFTP
- REST

Network Architecture:
- Prototypical research data portal
- Science DMZ
- Other Endpoints
- Other Services
- Globus Cloud
- Identity Provider
- Firewall
Why create your own services?

• Front-end / back-end within your portal
  – Remote backend for portal
  – Backend for pure Javascript browser apps

• Extend your portal with a public REST API, so that other app and service developers can integrate with and extend your portal
Why Globus Auth for your service?

- Outsource all identity management and authentication
  - Federated identity with InCommon, Google, etc.

- Outsource your REST API security
  - Consent, token issuance, validation, revocation
  - You provide service-specific authorization

- Apps use your service like all others
  - Its standard OAuth2 and OIDC

- Your service can seamlessly leverage other services

- Other services can leverage your service

- Implement your service using any language and framework

*Add your service to the science cyberinfrastructure platform*
1. Login and consent for portal and use of graph & transfer service.

2. Client credential grant to get access tokens

3. Authenticate with access tokens to invoke graph service: HTTPS with access token as header

4. Authenticate with graph service client id and secret to introspect token

5. Return validity, client, scope, effective identity, identity set (for the portal)

6. Verifies token, authorization checks

7. Graph service response

---

**Portal to Graph service interaction**

**Portal (Client)**

**Globus Auth (Authorization Server)**

**Graph Service (Resource Server)**
Service registration

- Client_id and client_secret for service
- Service display name
- Validated DNS name for service
- One or more scopes
  - Authorize clients to use each scope
    - All clients (public API), or specific clients
  - Declare dependent scopes
    - Need long-term, offline refresh tokens?
    - May require authorization from scope admin
- Links for terms of service & privacy policy
- Effective identity policy (optional)
- Email: support@globus.org
Typical service interactions

• Service receives HTTPS request with header
  – Authorization: Bearer <request-access-token>

• Introspects the request access token
  – Auth API: POST /v2/oauth2/token/introspect
  – Authorized by client_id and client_secret
  – Returns: validity, client, scope, effective_identity, identities_set

• Verifies token info

• Authorizes request

• If service needs to act as client to other services:
  – Calls Globus Auth Dependent Token Grant
    o Returns a token for each dependent service
  – Uses correct dependent token for downstream REST call

• Responds to client HTTPS request as appropriate
• Use identities_set when authorizing a request based on the resource owner associated with an access token
  – E.g., ACLs on Globus shared endpoints

• Authorizing based on set of identities is same complexity as authorizing based on group membership set
Groups

- **Globus group service is identity set aware**
  - “Tell me all groups for all identities of the logged in user”

- **Services can leverage this for authorization**
Token caching

• Service should cache tokens and related information
  – Improves performance of service
  – Reduces load on Globus Auth

• Access token -> introspect response
  – Cache timeout: 1-30 seconds recommended
  – To improve performance and load related to bursty use of REST API
  – Validity: Timeout duration determines responsiveness to token revocation and rescinding consent
  – client, scope, effective_identity: These will never change for an access token
  – identities_set: This may change at any time, due to identity (un)linking. May affect authorization. Timeout duration affect responsiveness to linking changes.
  – Future: add group membership to this, which is dependent on identities_set

• Access token -> dependent access tokens
  – Cache timeout: lifetime of access token
  – To avoid costly dependent token re-issuance
  – Rescinding consent will invalidate everything

• Refresh tokens
  – For however long they are needed for specific operations.
Prototypical research data portal

- Identity Provider
- Globus Web Helper Pages
- Globus Auth
- Portal Web Server (Client)
- Portal Endpoint
- Other Endpoints
- Other Services
- Science DMZ
- GridFTP
- REST
- HTTPS
Dependent tokens

• Your service can act as client to other services (scopes)
  – Globus Transfer and Auth
  – XSEDE (e.g., Jetstream, XUP)
  – Other community services
  – Future: Commercial services (e.g., Google Drive)

• Entire service call tree consented by user and service owners
  – Rescinding consent revokes all dependent tokens

• Dependent tokens are restricted to a particular client, calling a particular scope, on behalf of a particular resource owner (e.g., user)
  – Restricted delegation!
Graph service to transfer interaction

1. Client credential grant to get access tokens

2. Authenticate with access tokens to invoke graph service, user identity as parameter

3. Verifies token, authorization checks and generate graph

4. Request dependent tokens, authenticate with graph client id and secret

5. Returns access tokens for use with transfer service & any other downstream service

6. Invoke transfer service with access token to move files and set permissions for user to access files
Graph service to transfer interaction

Portal (Client)

Graph Service (Resource Server)

Globus Auth (Authorization Server)

Transfer Service (Resource Server)

1. Portal requests
2. Token introspection
3. Authorize service request
4. Transfer service response
5. Graph service response

7. Token introspection
8. Authorize, process request
9. Graph service response

Modern Research Data Portal
Walk-through

Graph Service Code
Exercise: Graph service

• Either locally or on EC2 instance

• Modify service/service.conf
  – PORTAL_CLIENT_ID should be set to your portal’s client id from portal/portal.conf

  [link](github.com/globus/globus-sample-data-portal.git)

• Find and print to console:
  – Expiration time of each of dependent tokens
  – The complete ACL rule added to the folder for the user
  – The full response from token introspection

• Modify cleanup to wait for files to be deleted before returning
Join the Globus developer community

• Join developer-discuss@globus.org mailing lists: globus.org/mailing-lists

• Python SDK is open source
  – github.com/globus/globus-sdk-python
  – Submit issues, pull requests
  – Discussions on developer-discuss@globus.org

• Jupyter notebook, sample data portal and native applications are open source on github

• Documentation: docs.globus.org

• We’re hiring: globus.org/jobs