Building the Modern Research Data Portal using the Globus Platform

Rachana Ananthakrishnan

rachana@globus.org

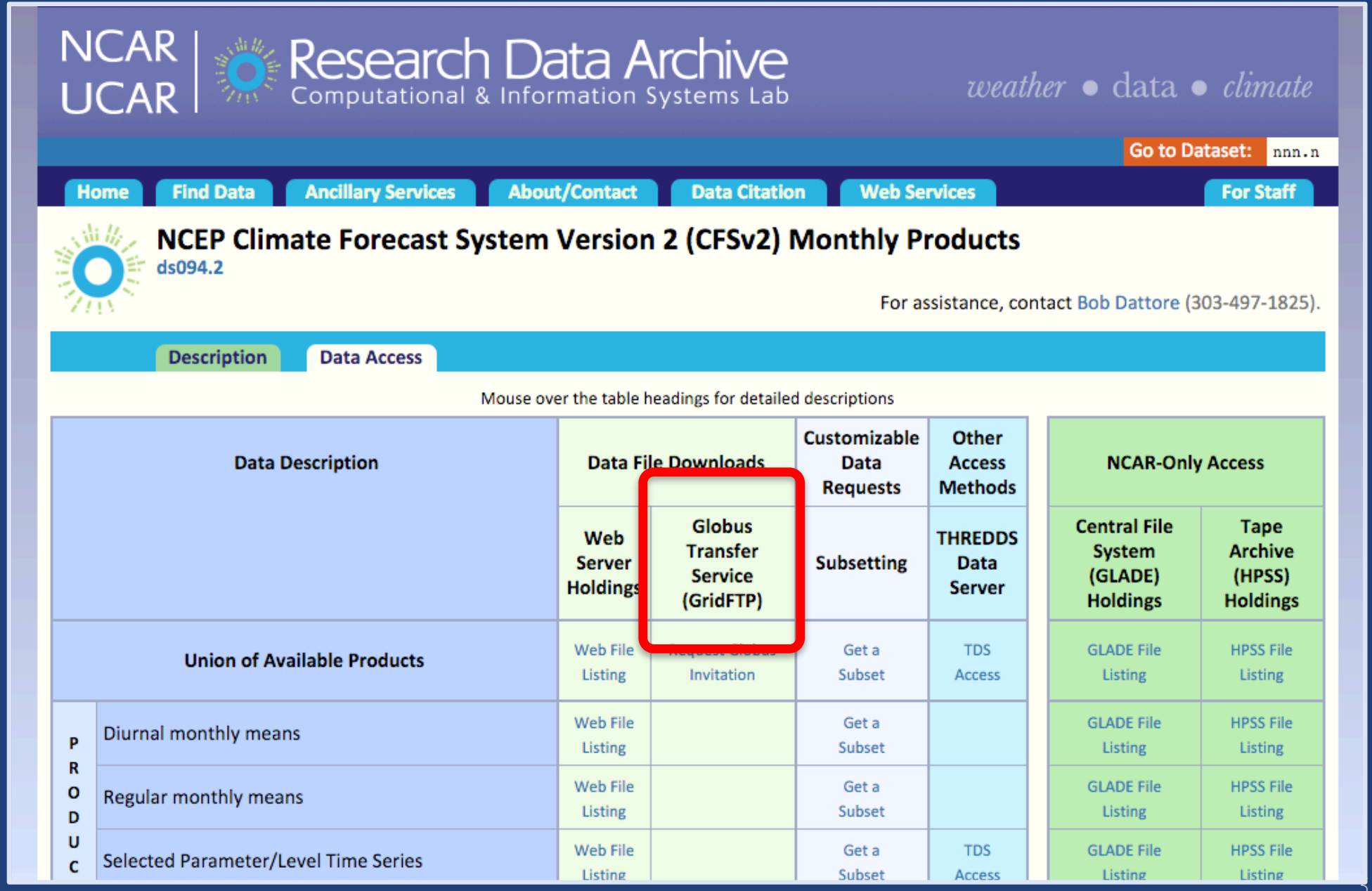
GlobusWorld 2017

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



NCAR UCAR | Research Data Archive
Computational & Information Systems Lab
weather • data • climate

Go to Dataset: nnn.n

Home | Find Data | Ancillary Services | About/Contact | Data Citation | Web Services | For Staff

NCEP Climate Forecast System Version 2 (CFSv2) Monthly Products
ds094.2

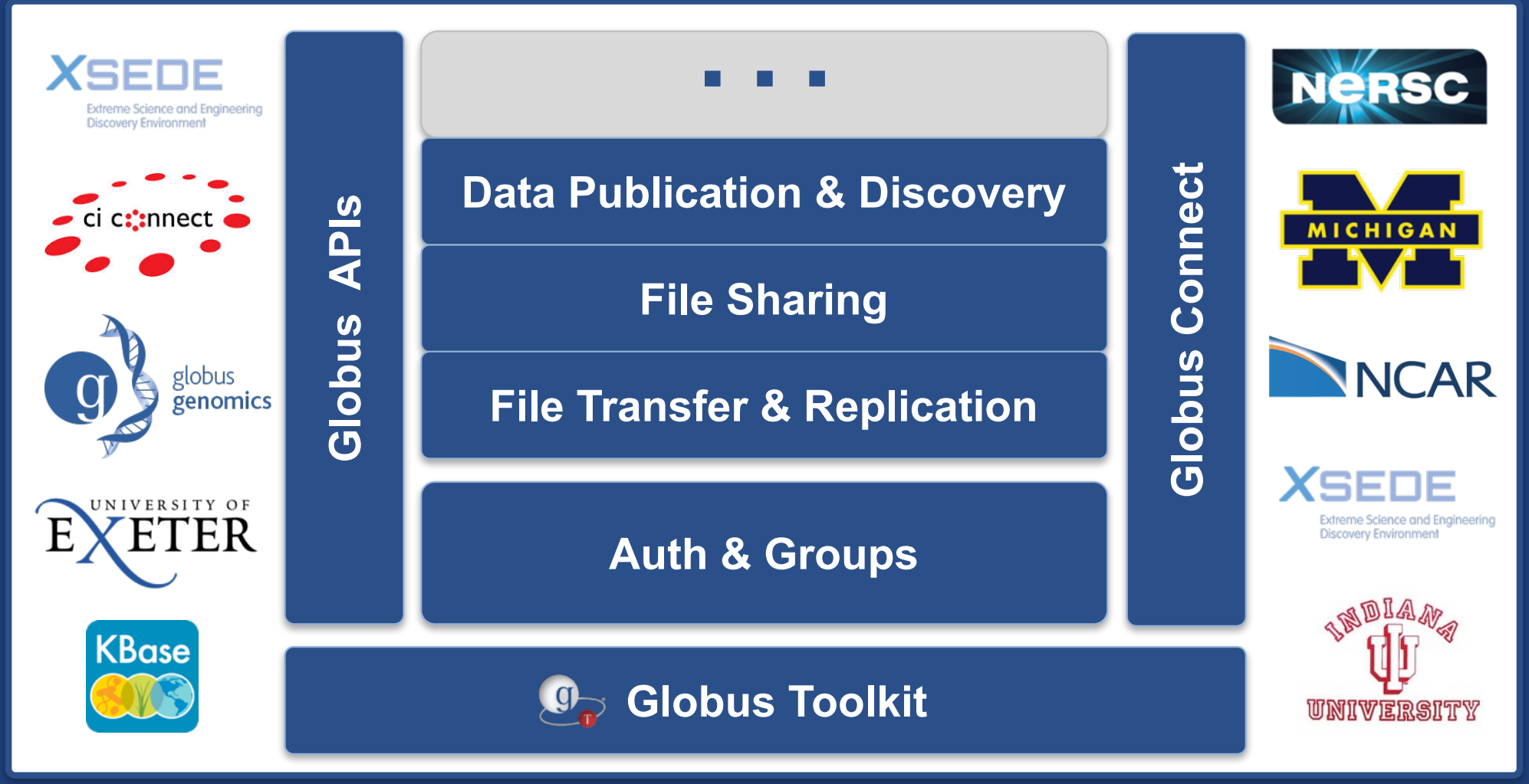
For assistance, contact Bob Dattore (303-497-1825).

Description | Data Access

Mouse over the table headings for detailed descriptions

Data Description		Data File Downloads		Customizable Data Requests	Other Access Methods	NCAR-Only Access	
		Web Server Holdings	Globus Transfer Service (GridFTP)	Subsetting	THREDDS Data Server	Central File System (GLADE) Holdings	Tape Archive (HPSS) Holdings
Union of Available Products		Web File Listing	Invitation	Get a Subset	TDS Access	GLADE File Listing	HPSS File Listing
PRODUC	Diurnal monthly means	Web File Listing		Get a Subset		GLADE File Listing	HPSS File Listing
	Regular monthly means	Web File Listing		Get a Subset		GLADE File Listing	HPSS File Listing
	Selected Parameter/Level Time Series	Web File Listing		Get a Subset	TDS Access	GLADE File Listing	HPSS File Listing

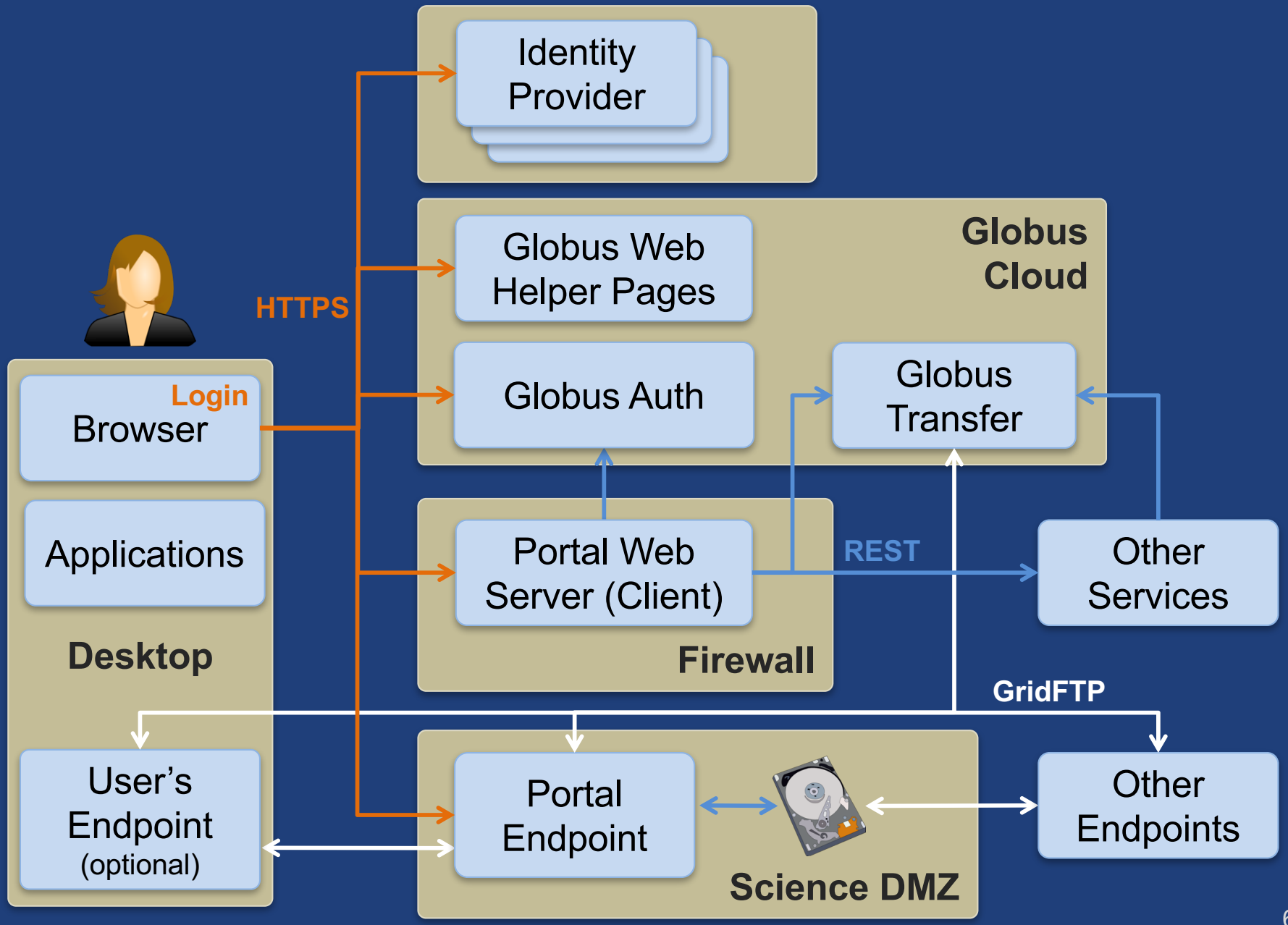
Globus PaaS


XSEDE
Extreme Science and Engineering Discovery Environment
ci connect
globus genomics
UNIVERSITY OF EXETER
KBase
Globus APIs
. . .
Data Publication & Discovery
File Sharing
File Transfer & Replication
Auth & Groups
Globus Toolkit
Globus Connect
NERSC
MICHIGAN
NCAR
XSEDE
Extreme Science and Engineering Discovery Environment
INDIANA UNIVERSITY

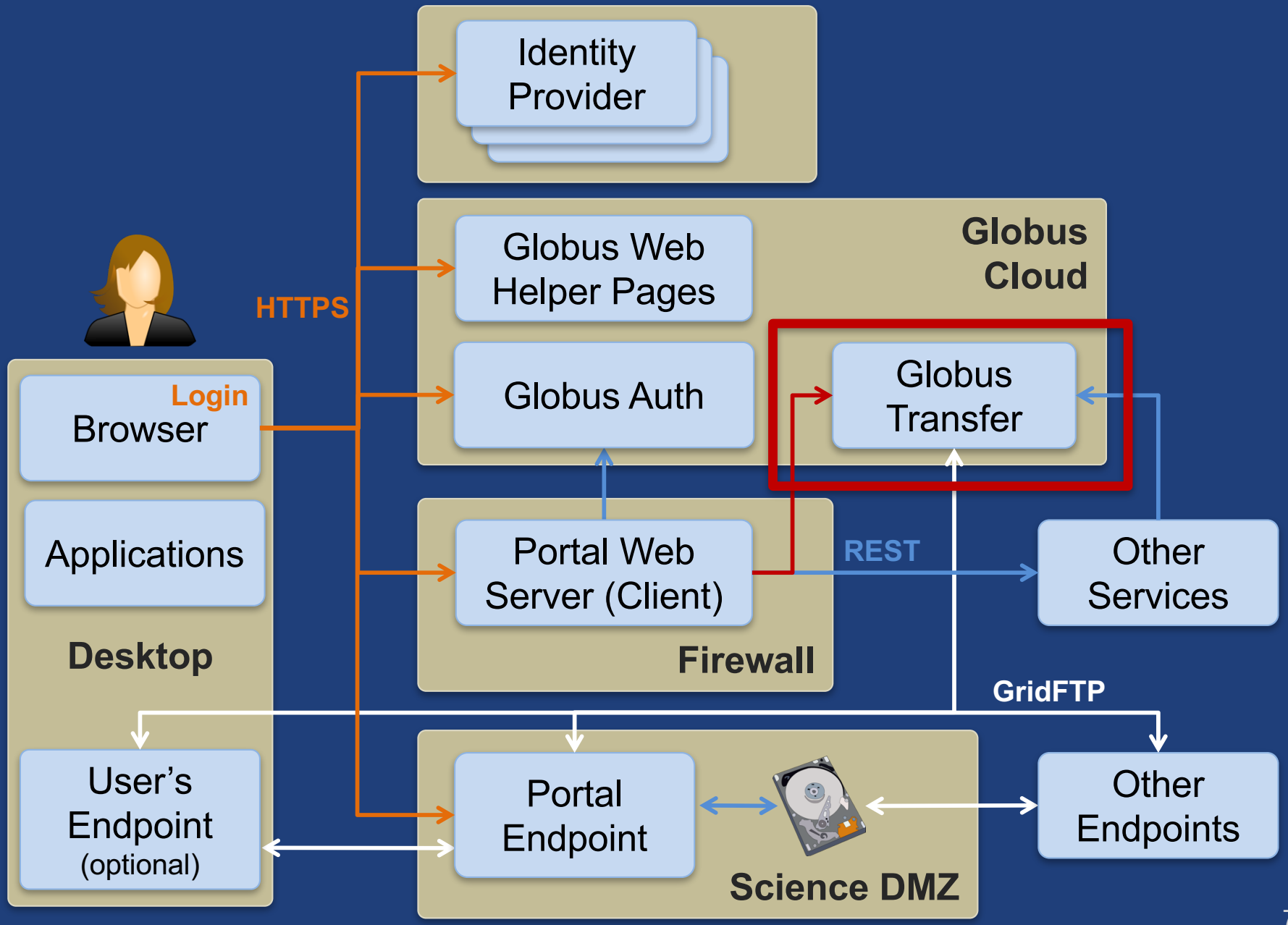
Demo

Sample Research Data Portal

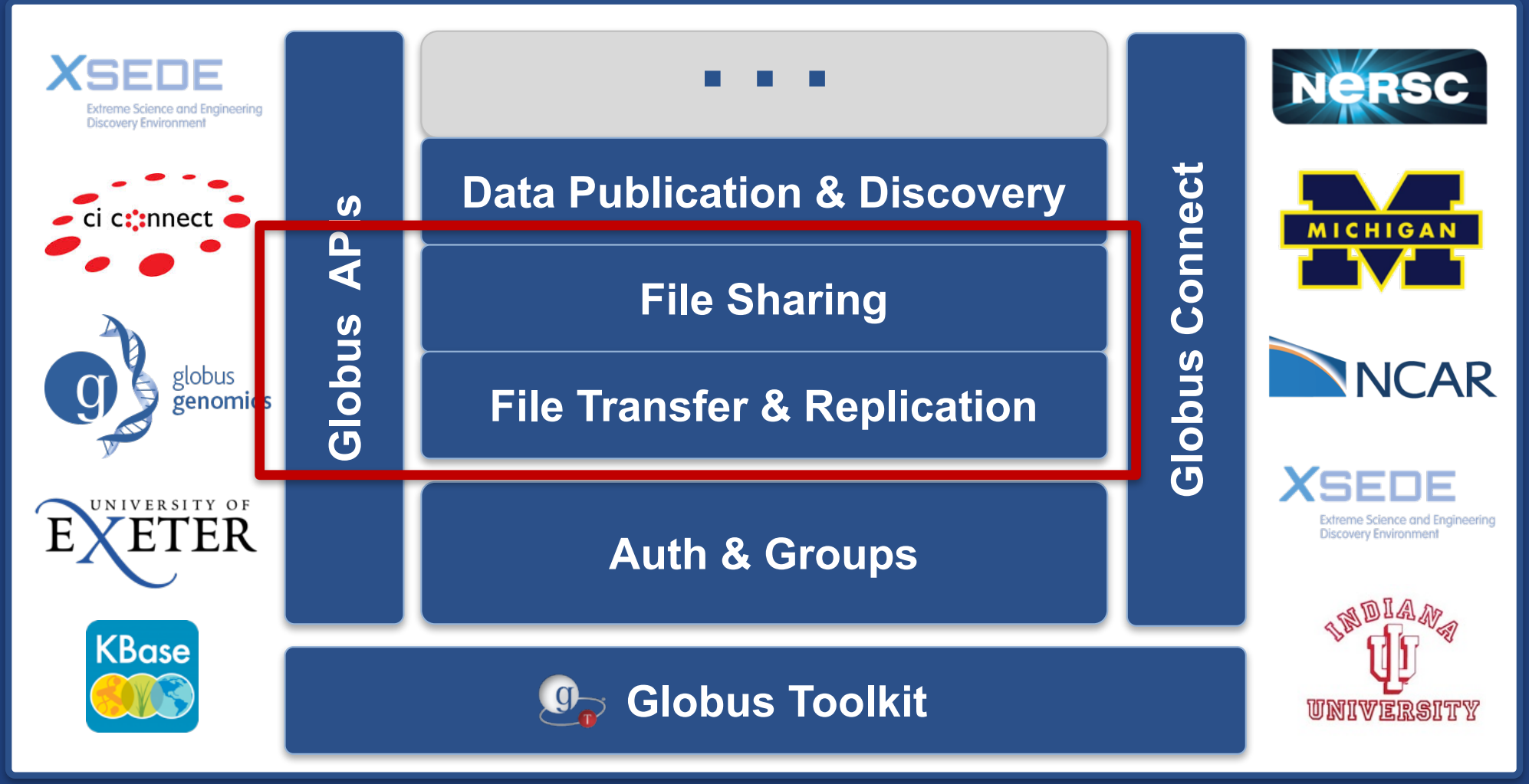
Prototypical research data portal



Prototypical research data portal



Globus PaaS


XSEDE
Extreme Science and Engineering Discovery Environment
ci connect
globus genomics
UNIVERSITY OF EXETER
KBase
Globus APIs
. . .
Data Publication & Discovery
File Sharing
File Transfer & Replication
Auth & Groups
Globus Connect
Globus Toolkit
NERSC
MICHIGAN
NCAR
XSEDE
Extreme Science and Engineering Discovery Environment
INDIANA UNIVERSITY

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 asdflkqhafsdafeawk

Globus Transfer API

- **Nearly all Globus Web App functionality implemented via public Transfer API**

docs.globus.org/api/transfer

- **Fairly stable**
- **Deprecation policy**

Globus Python SDK

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

globus.github.io/globus-sdk-python

`TransferClient` class

- `globus_sdk.TransferClient` **class**

```
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

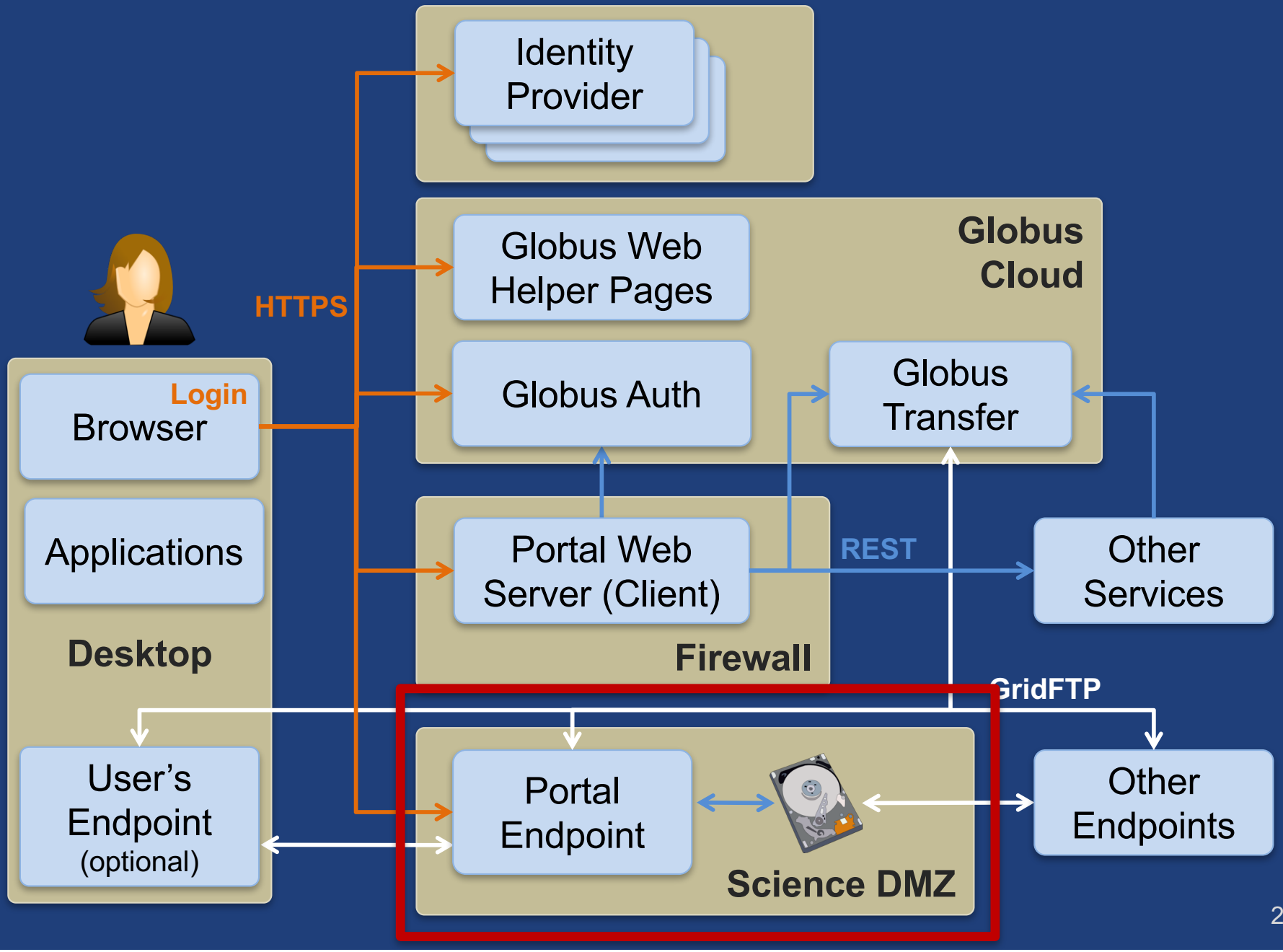
Install Jupyter notebook either locally or on EC2 instance

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

Modify Jupyter notebook to:

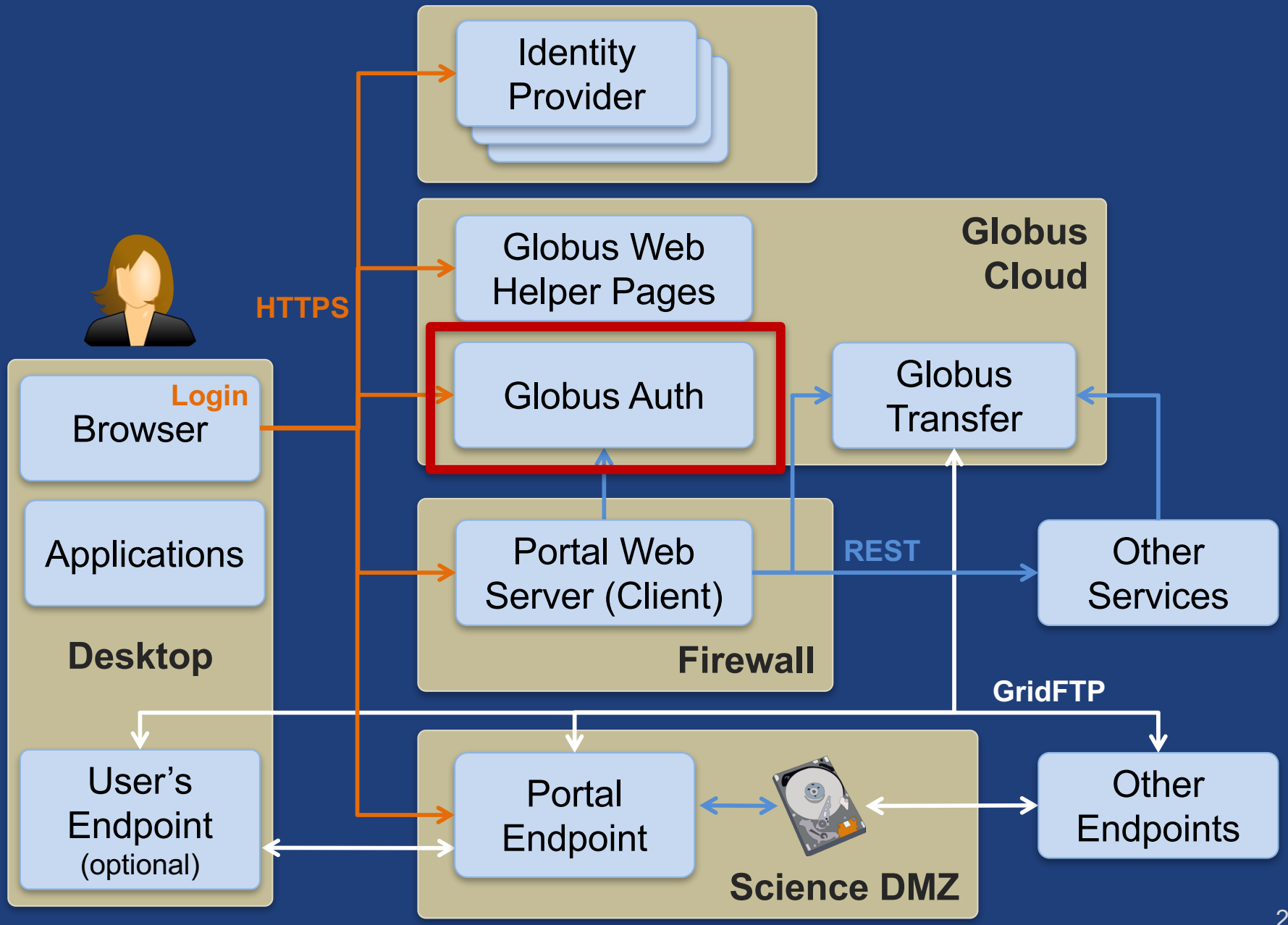
1. **Find the endpoint id for XSEDE Comet**
2. **Set all the metadata fields on your shared endpoint**
3. **Set permissions to allow your neighbor to access your shared endpoint**
4. **Transfer all files *.txt from the tourexercise directory on the Globus Vault endpoint to any other endpoint.**
5. **Monitor 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

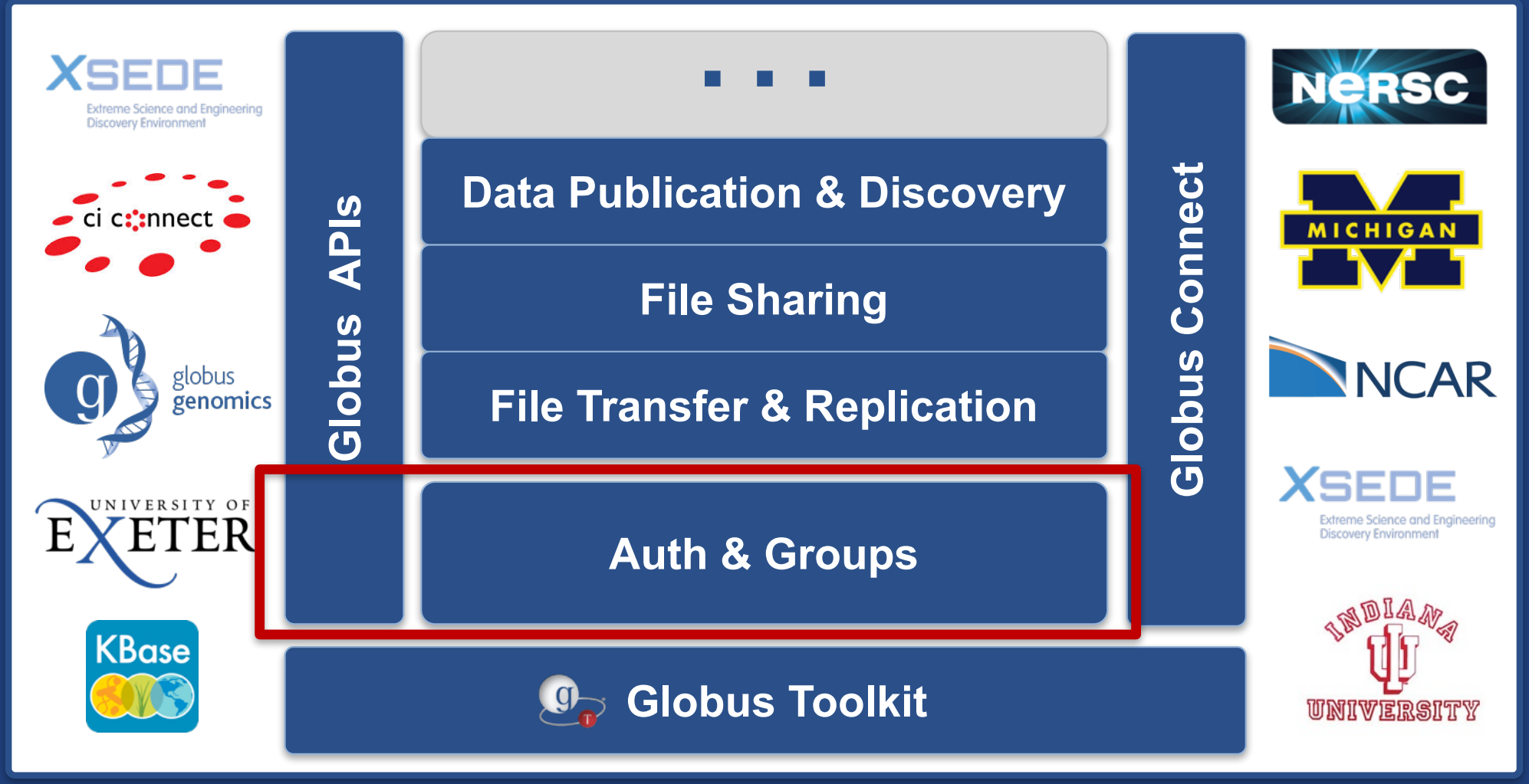


Maximizing the value of the Science DMZ

Prototypical research data portal



Globus PaaS


XSEDE
Extreme Science and Engineering Discovery Environment
ci connect
globus genomics
UNIVERSITY OF EXETER
KBase
Globus APIs
. . .
Data Publication & Discovery
File Sharing
File Transfer & Replication
Auth & Groups
Globus Connect
Globus Toolkit
NERSC
MICHIGAN
NCAR
XSEDE
Extreme Science and Engineering Discovery Environment
INDIANA UNIVERSITY

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

- **Specification**
- **Developer Guide**
- **API Reference**

Based on widely used web standards

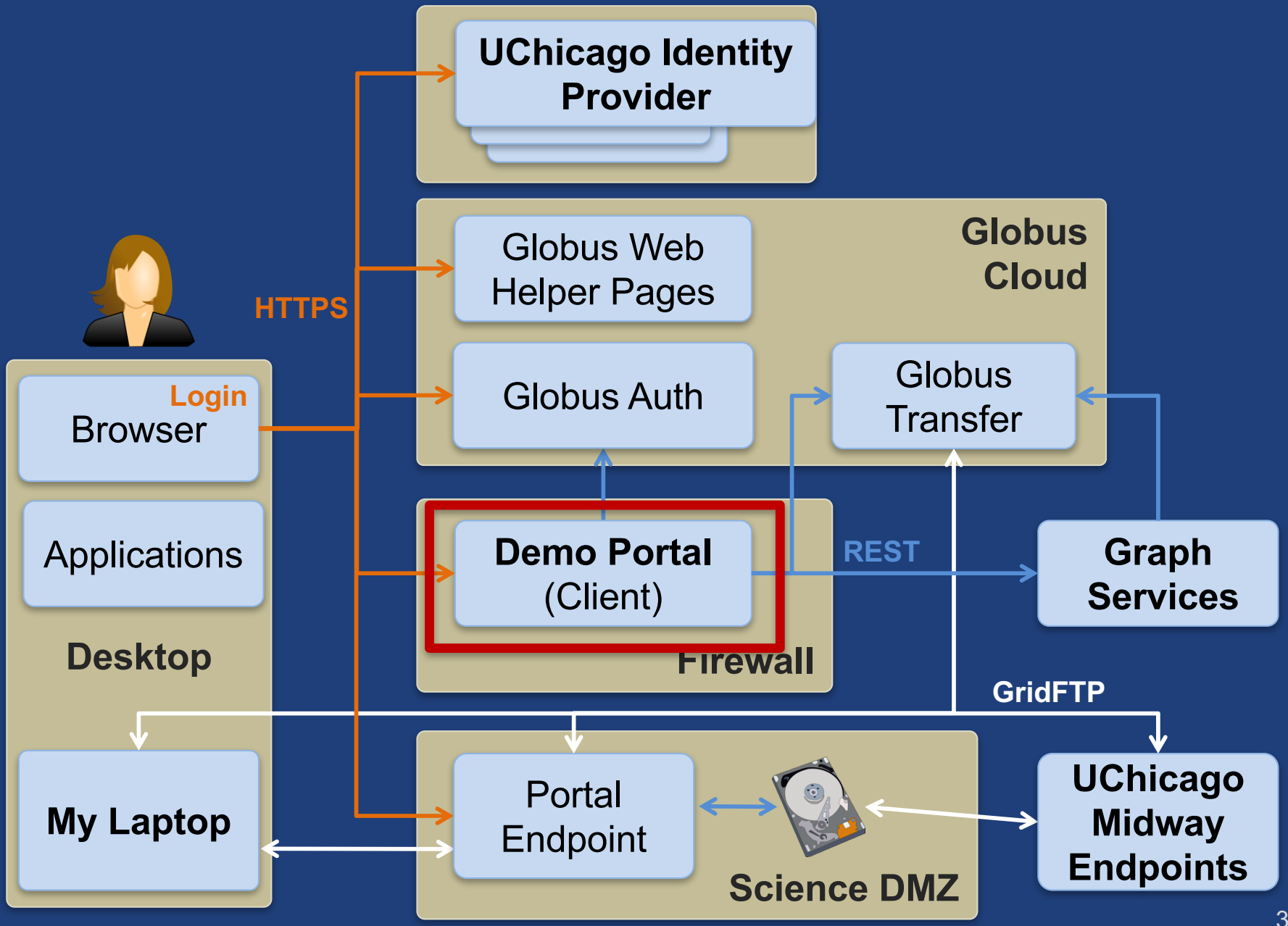
- **OAuth 2.0 Authorization Framework**
  - aka OAuth2
- **OpenID Connect Core 1.0**
  - aka OIDC

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

Globus account

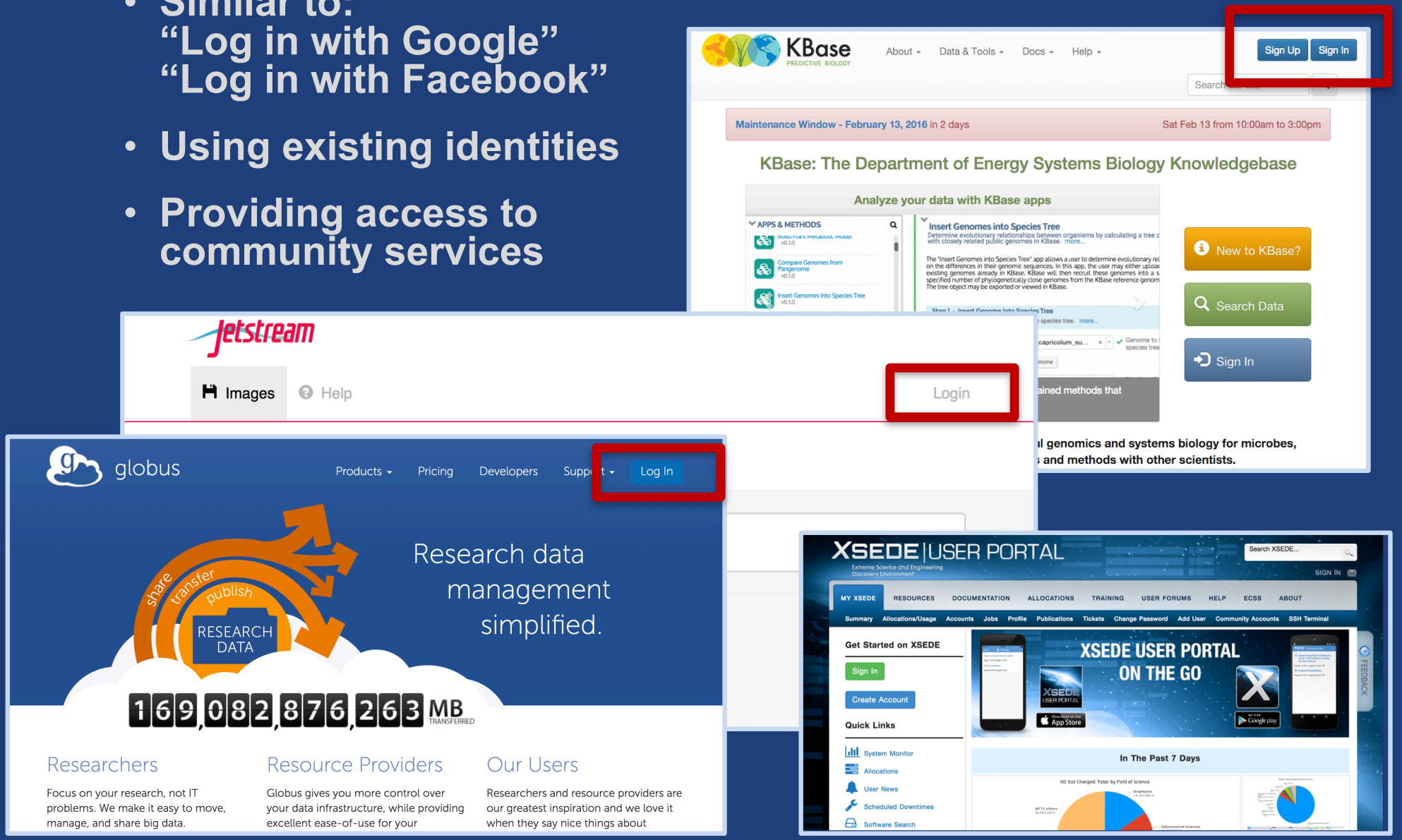
- **A Globus account is a set of identities**
  - A *primary identity*
    - Identity can be primary of only one account
  - One or more *linked identities*
    - Identity can (currently) be linked to only one account
- **Account does not have own identifier**
  - An account is uniquely identified using its primary identity

Sample Research Data Portal



Use case: Log in with Globus

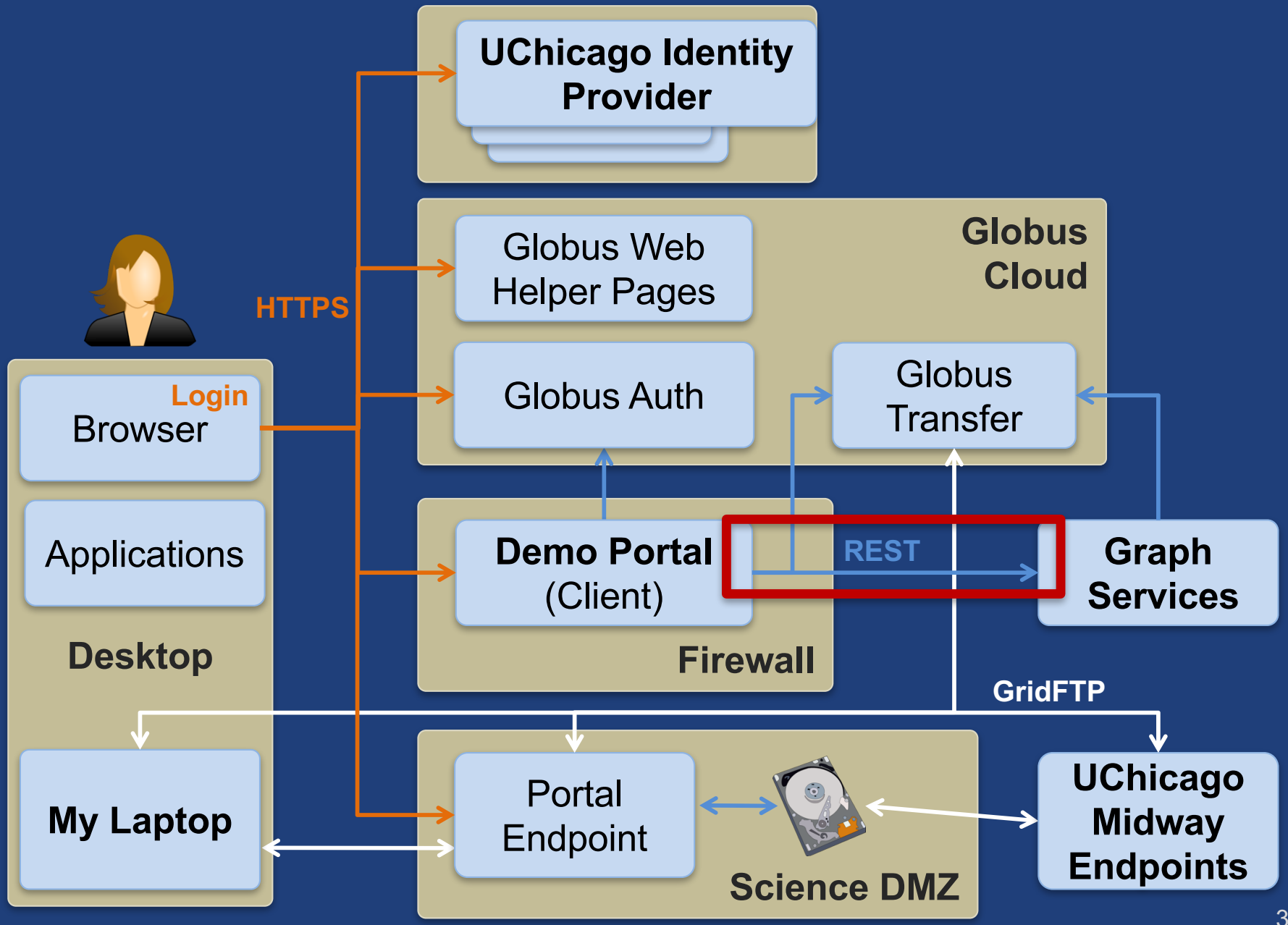
- **Similar to:**
  “Log in with Google”
  “Log in with Facebook”
- **Using existing identities**
- **Providing access to community services**



Demo

Jetstream App use of Globus Auth

Sample Research Data Portal



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

Authorization Code Grant

Browser (User)

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

Modern Research Data Portal

Portal (Client)

Globus Auth (Authorization Server)

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

Globus Transfer (Resource Server)

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
    - 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**

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

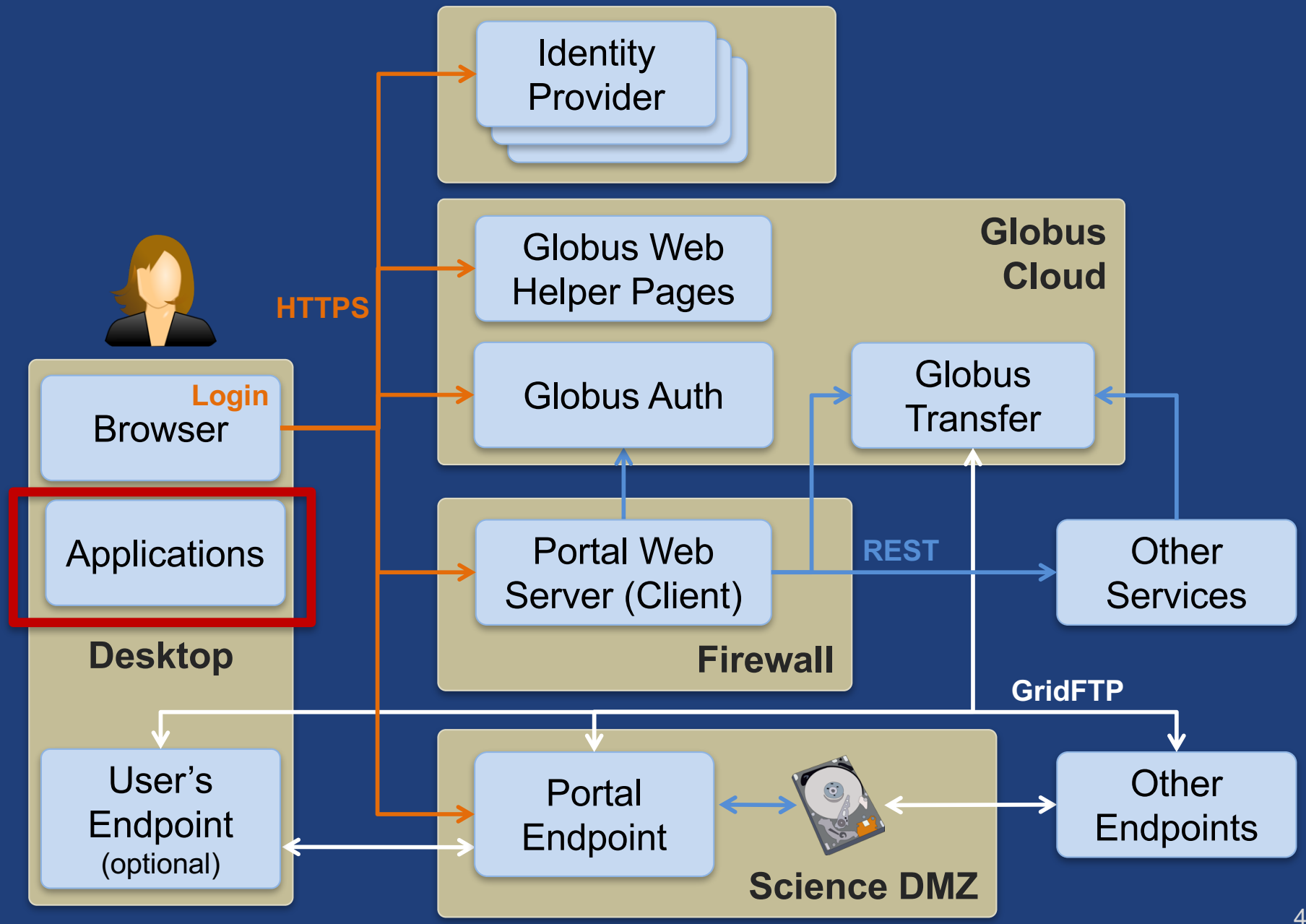
github.com/globus/globus-sample-data-portal.git

- **Register your application at:**

developers.globus.org

- **Instructions in the README file**

Prototypical research data portal



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

Native App grant

1. Run application

Browser

3. Authenticate and consent

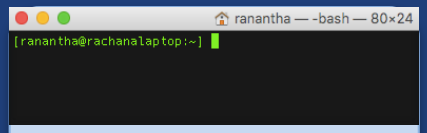
4. Auth code

2. URL to authenticate

5. Register auth code

6. Exchange code

7. Access tokens



Native App
(Client)

Globus Auth (Authorization Server)

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

Globus Transfer (Resource Server)

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

Refresh tokens

1. Run application

Browser

3. Authenticate and consent

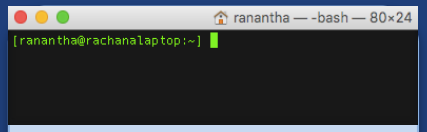
4. Auth code

2. URL to authenticate

5. Register auth code

6. Exchange code, request refresh tokens

7. Access tokens and refresh tokens



Native App (Client)

9. Exchange refresh token for new access tokens

10. Access tokens

8. Store refresh tokens

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

Globus Auth (Authorization Server)

Globus Transfer (Resource Server)

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**

Sample Research Data Portal

Demo: Native Apps and Refresh tokens

Exercise: Native App

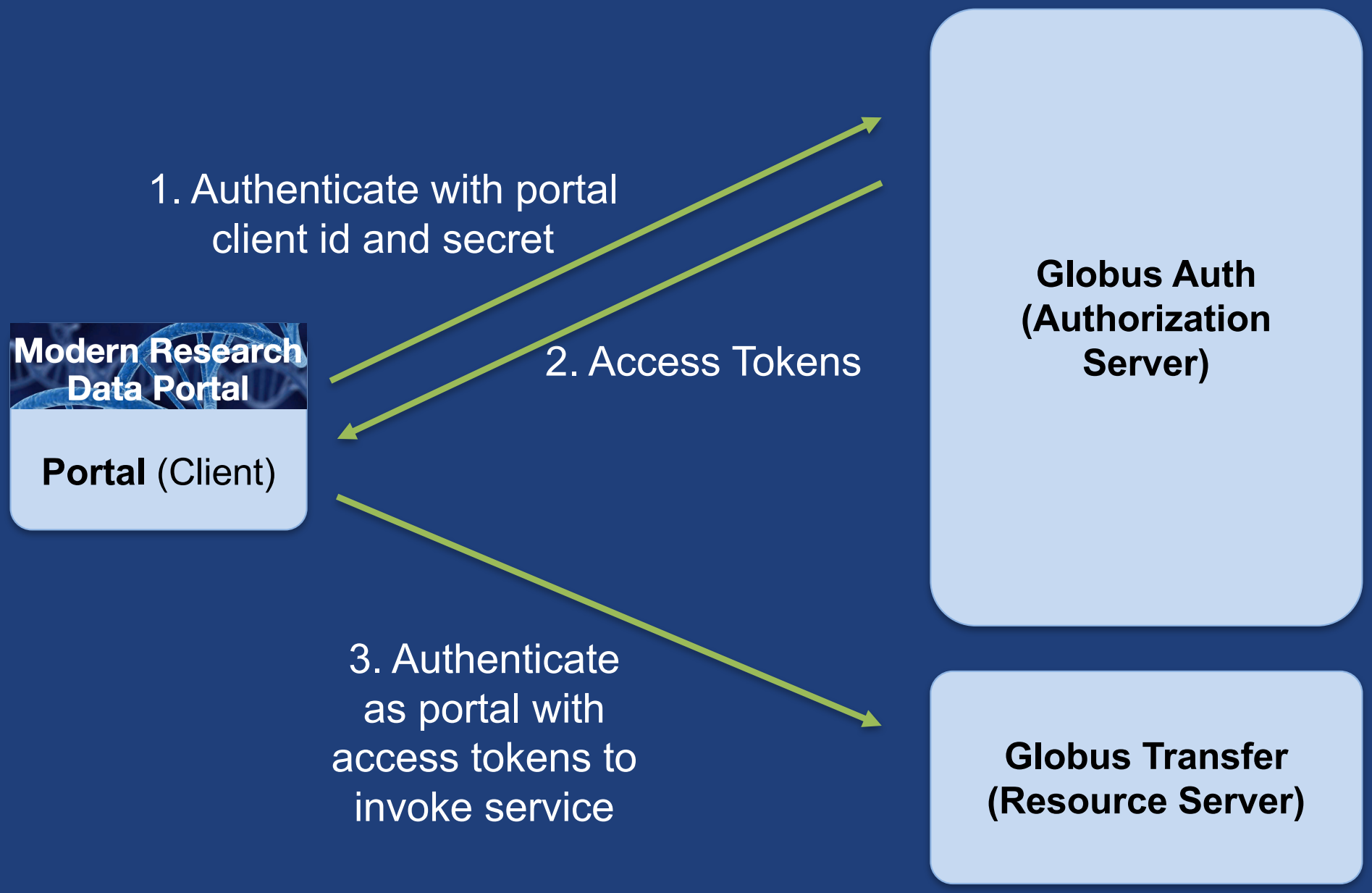
https://github.com/globus/native-app-examples

- **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

Client credential grant



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

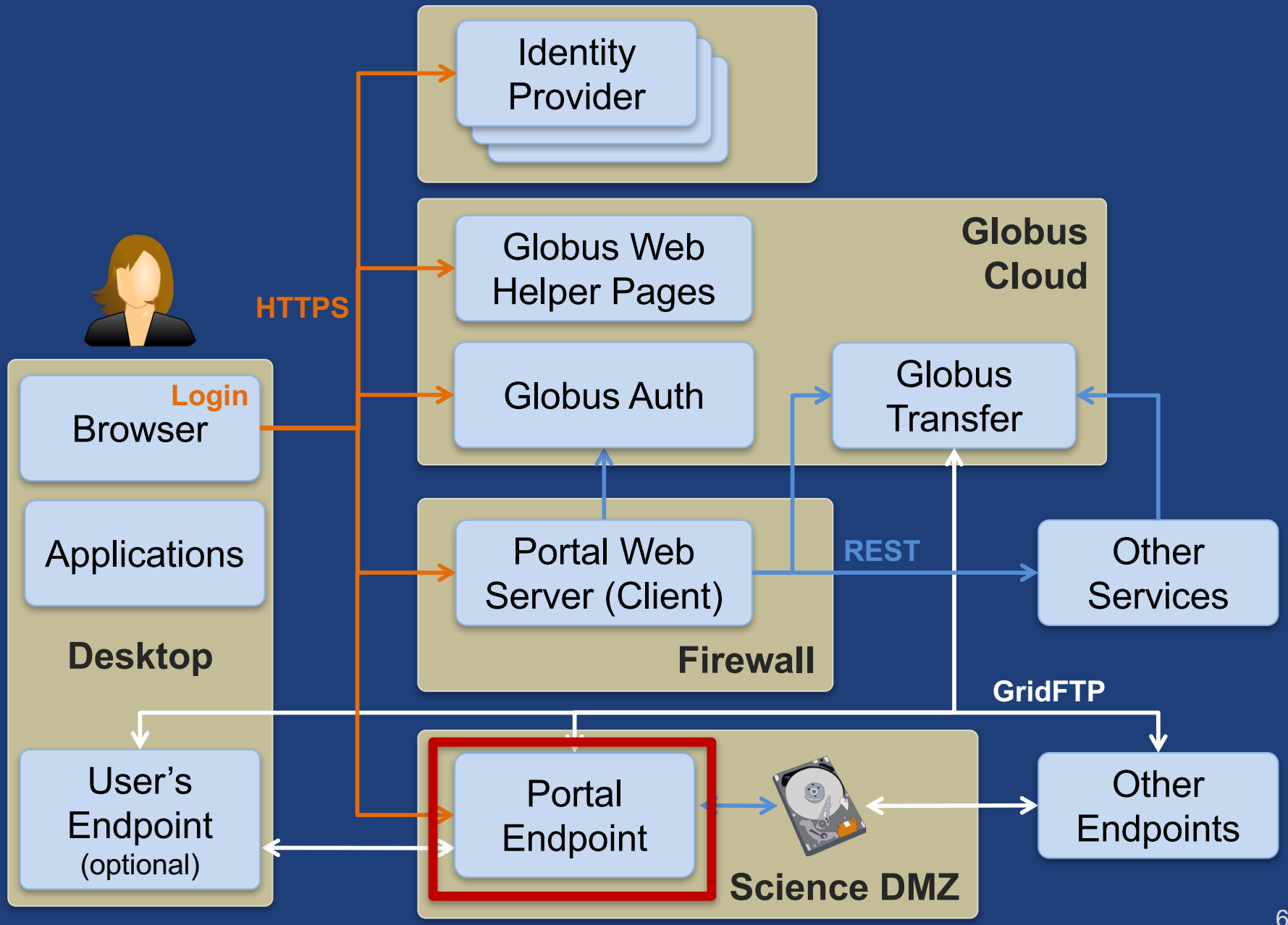
Client identity

- **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**

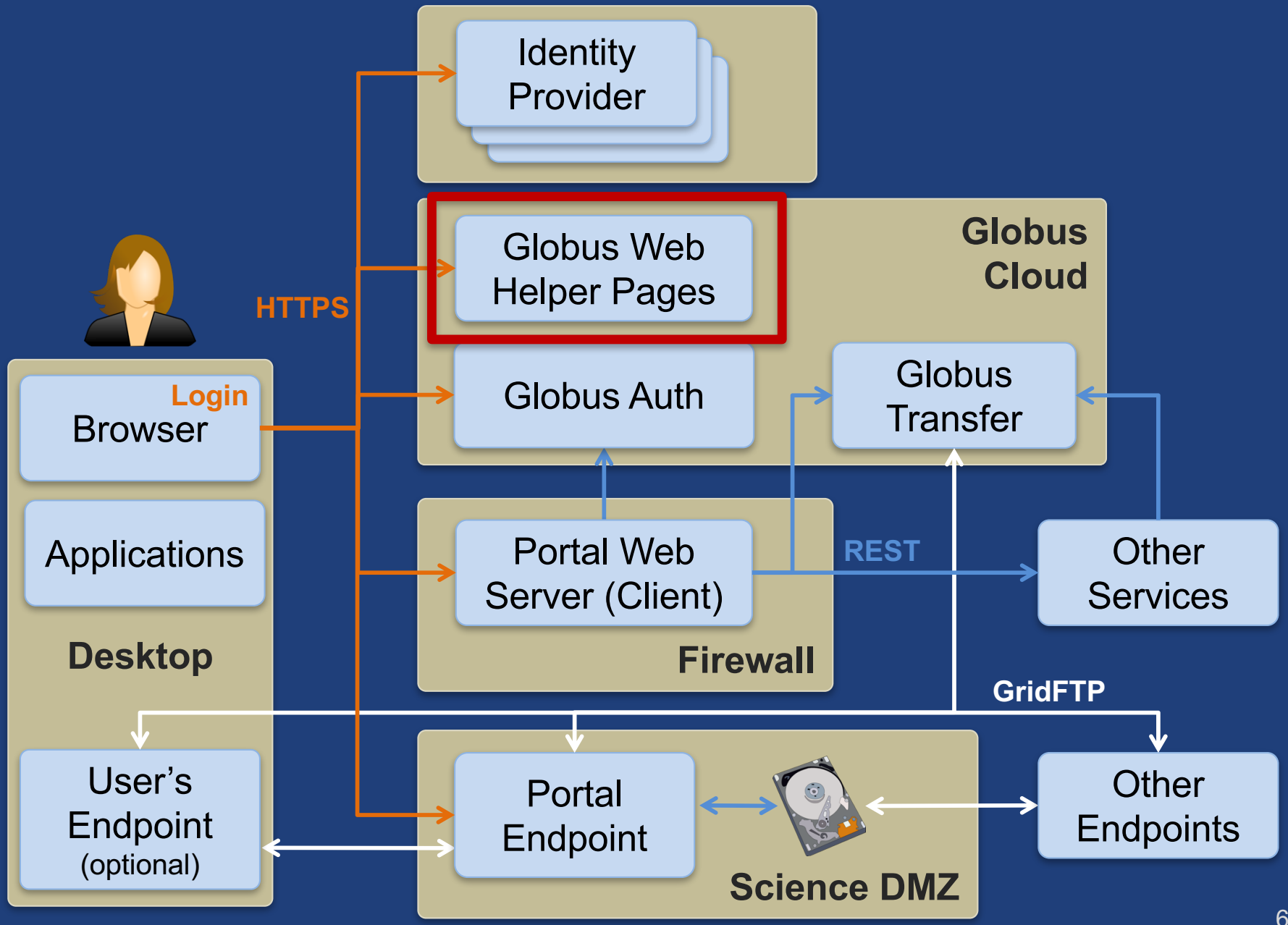
Prototypical research data portal



HTTPS to Endpoints

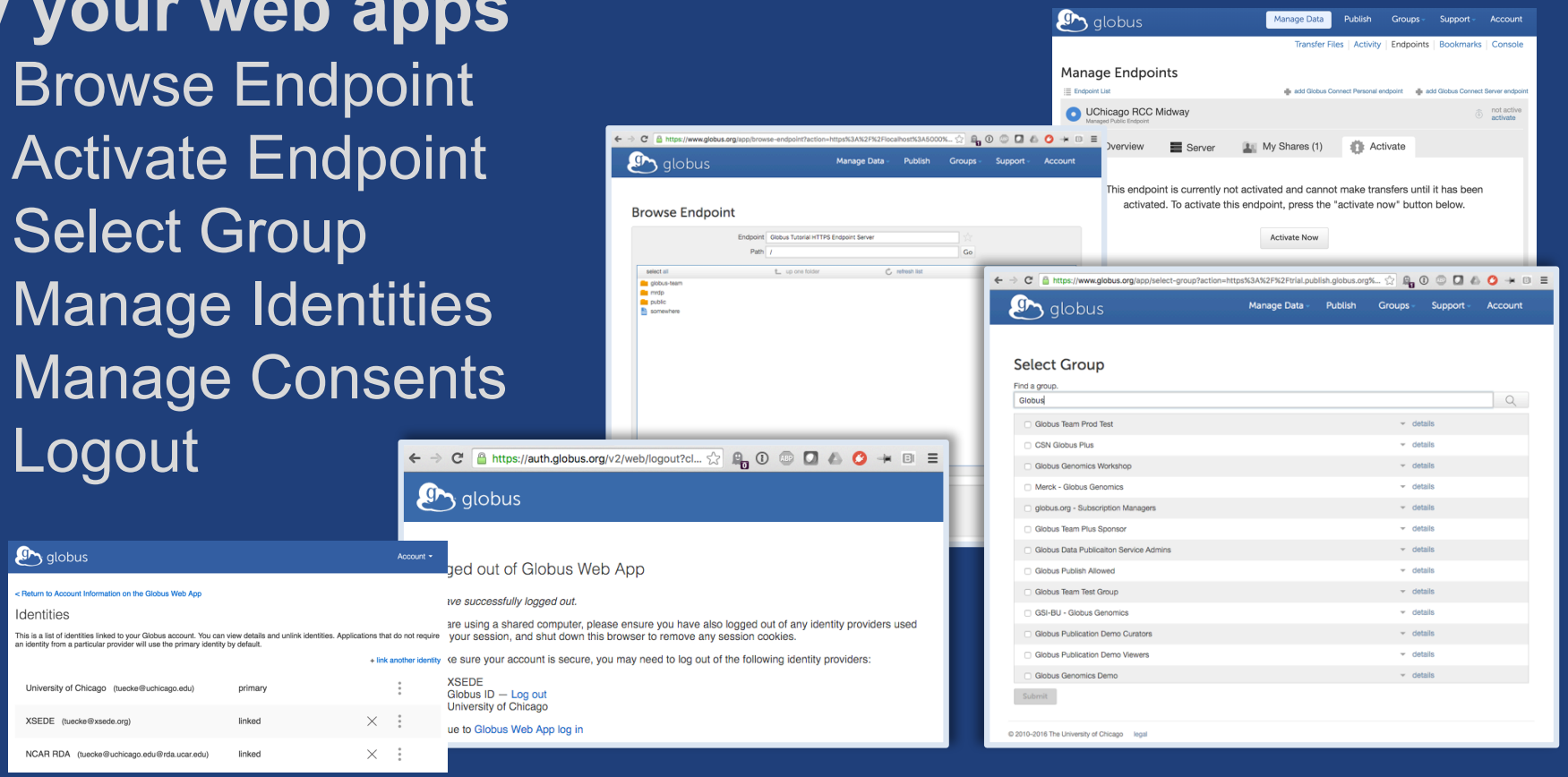
- **Each endpoint HTTPS server is a Globus Auth service (resource server)**
- **Web page can link to file on server**
  - Browser GET will cause HTTPS server to authorize request via Globus Auth (note SSO)
- **Portal (client) can request scope for endpoint resource server**
  - Use access token in requests

Prototypical research data portal



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

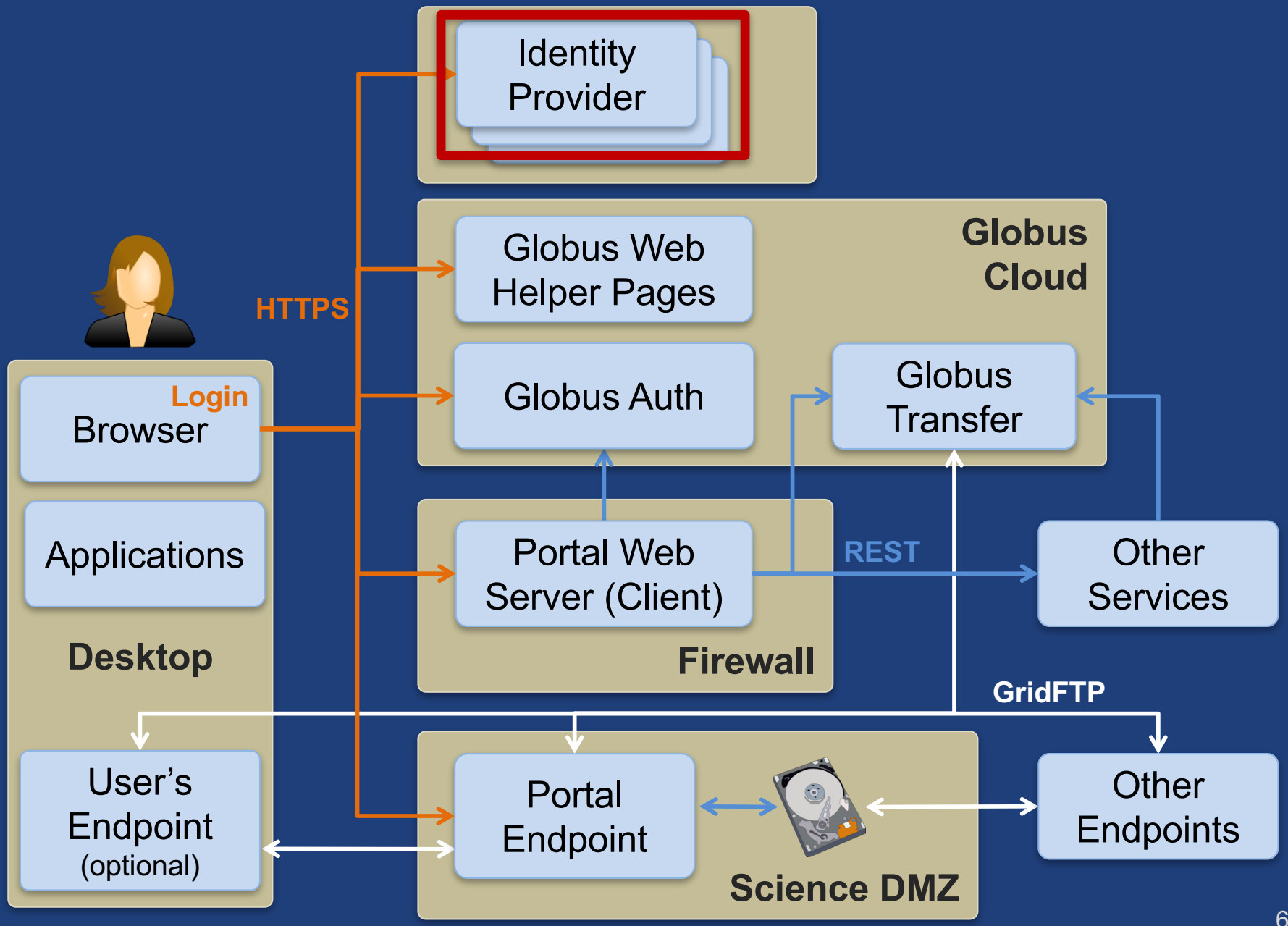


docs.globus.org/api/helper-pages

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



Adding your identity provider

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

- **Any other OpenID Connect identity provider** ***(subscription)***

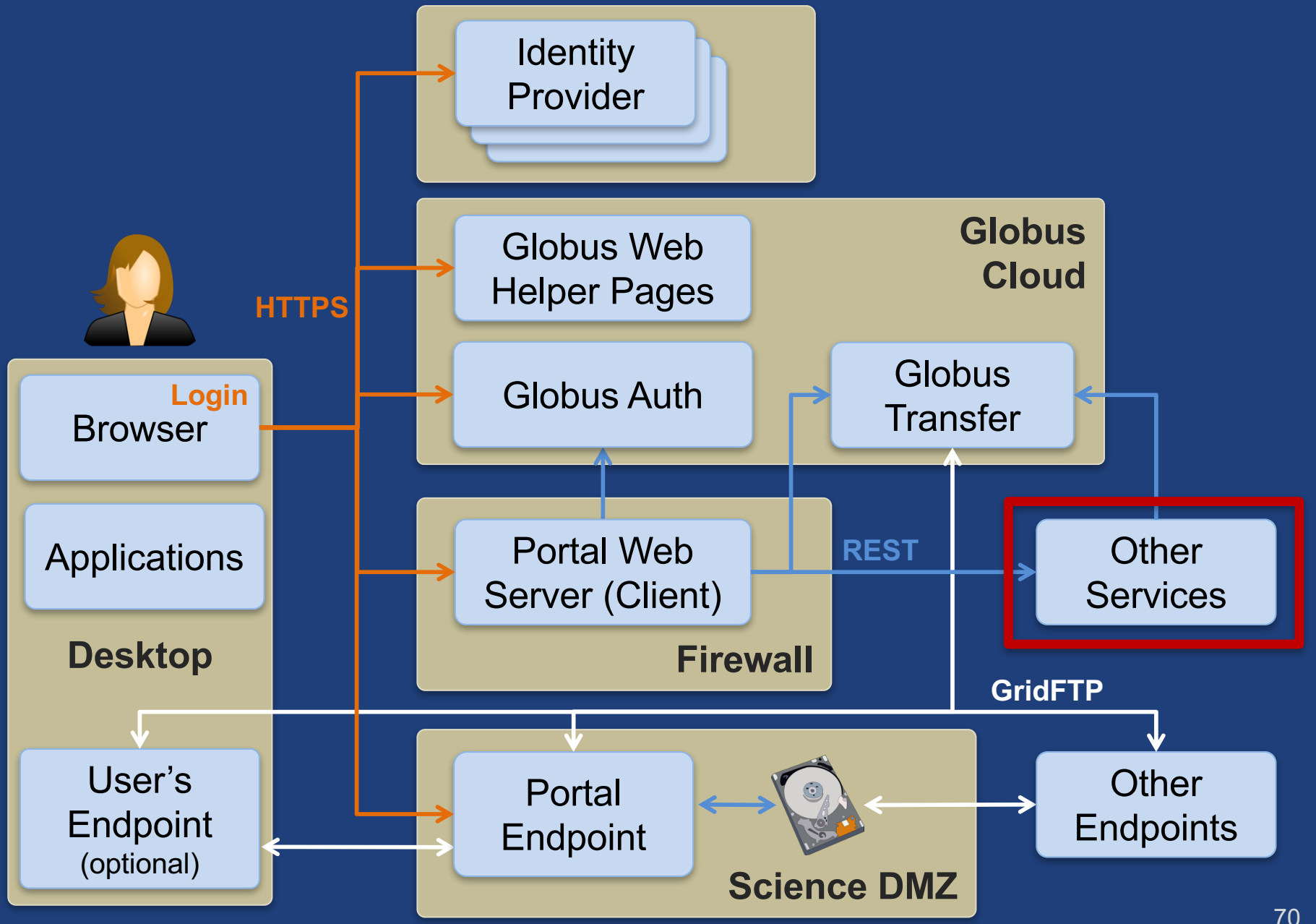
Adding an identity provider

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

Portal accounts for users

- **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



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

Portal to Graph service interaction

1. Login and consent for portal and use of graph & transfer service.

2. Client credential grant to get access tokens

Globus Auth (Authorization Server)

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

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

Modern Research Data Portal

Portal (Client)

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

Graph Service (Resource Server)

6. Verifies token, authorization checks

7. Graph service response

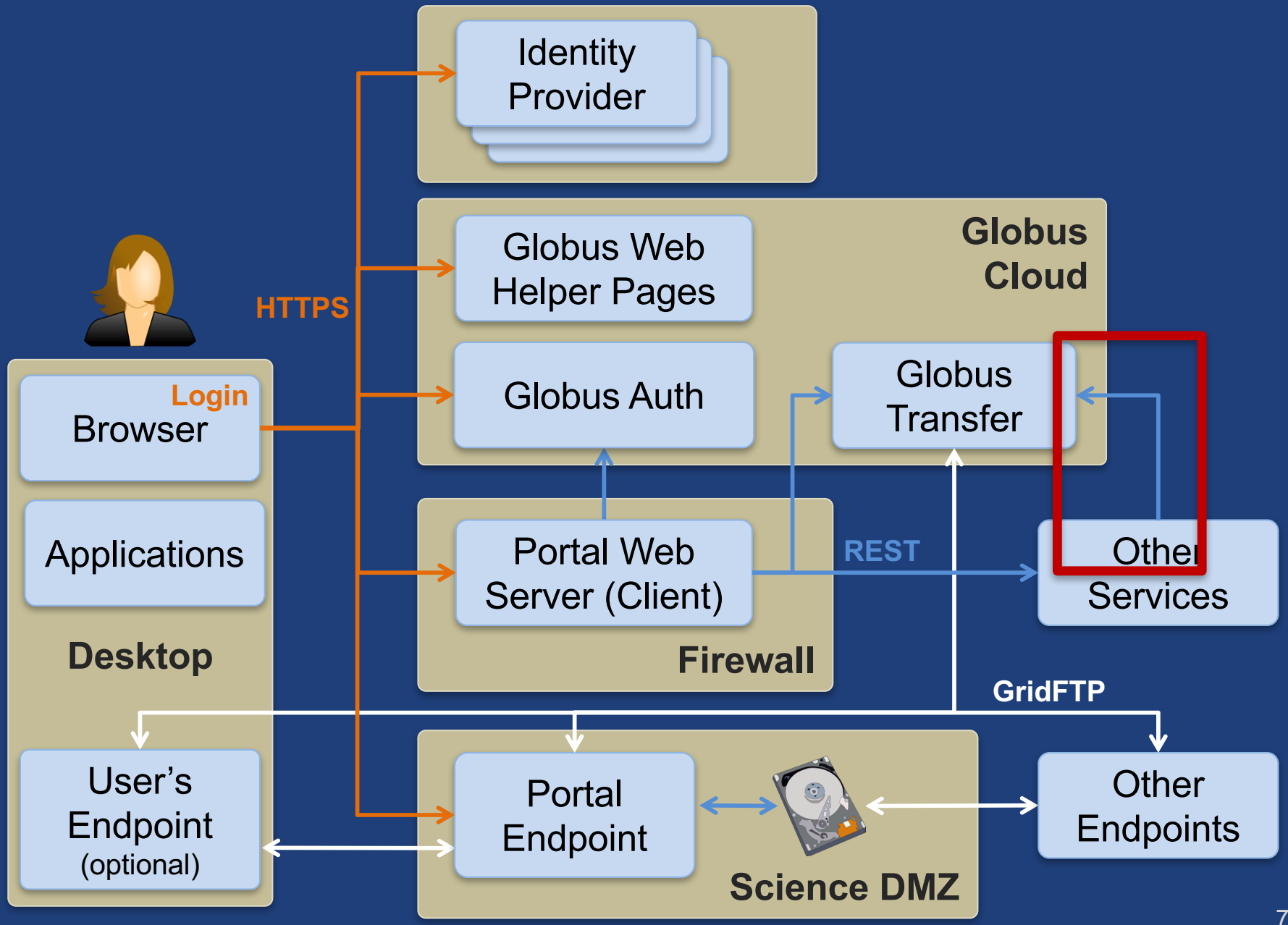
Service registration

- **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
    - Returns a token for each dependent service
  - Uses correct dependent token for downstream REST call
- **Responds to client HTTPS request as appropriate**

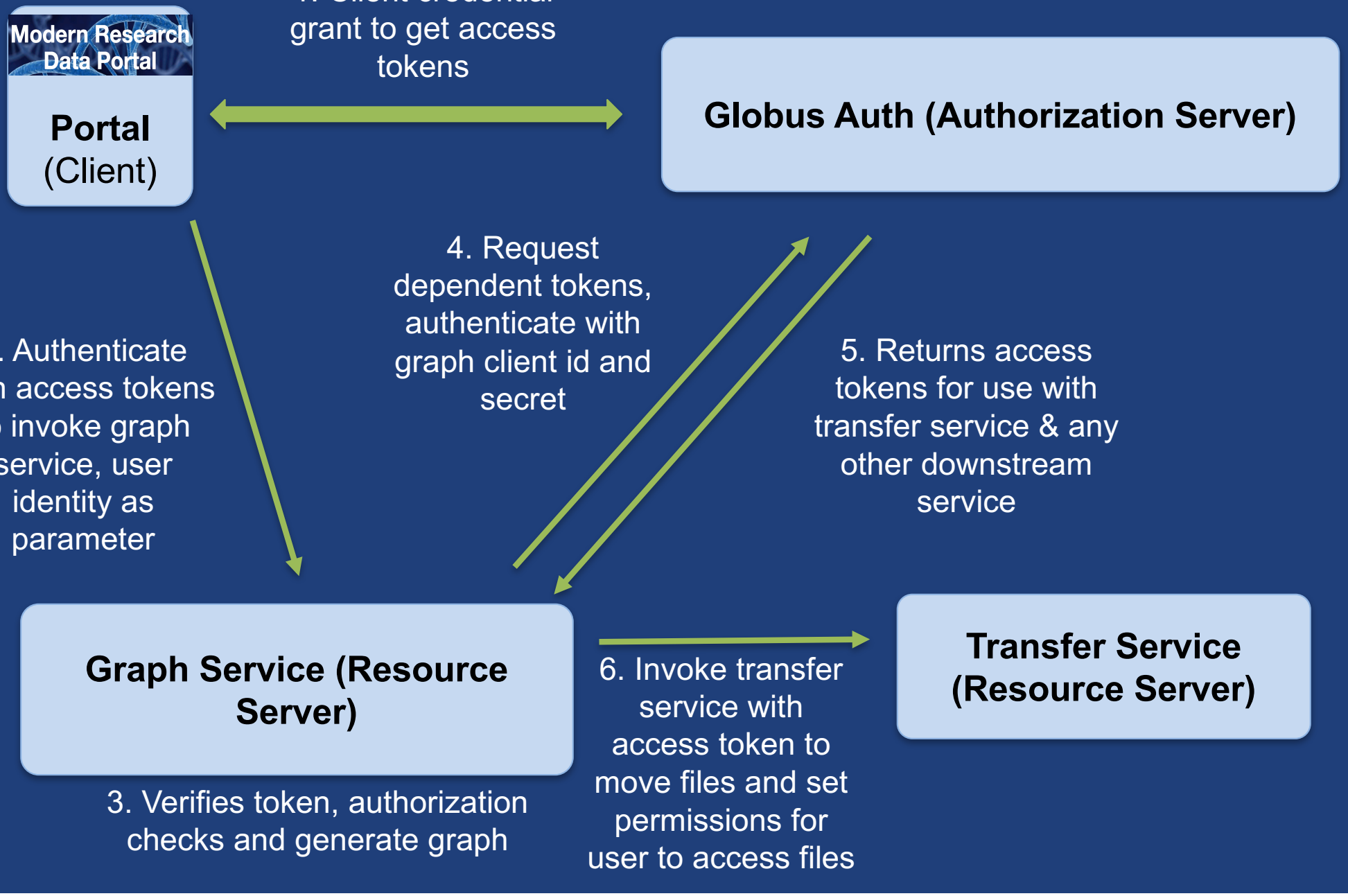
Prototypical research data portal



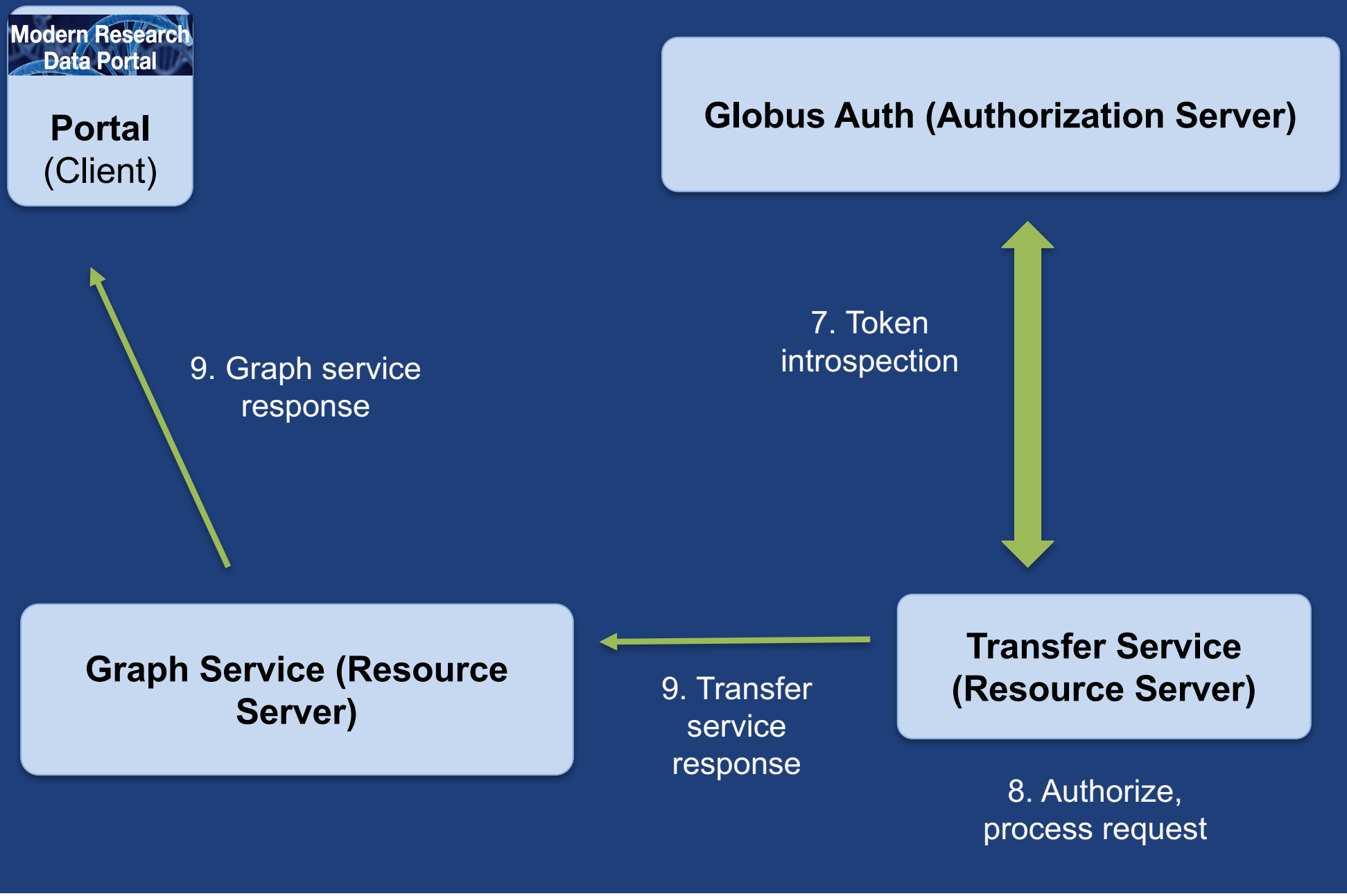
Dependent tokens

- **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



Graph service to transfer interaction



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

    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**

Additional Features for Service Developers

Effective identity

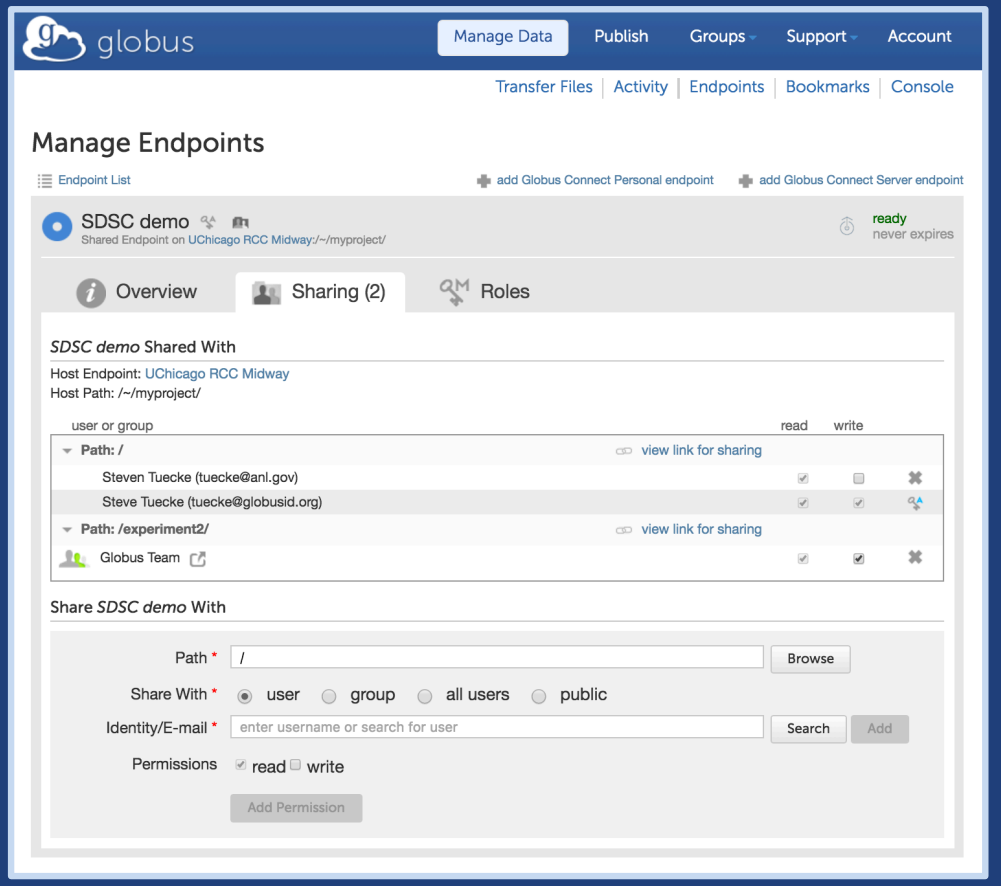
- **App or service can choose to operate only with identities from a particular identity provider**
  - Globus Auth login will require an identity from that provider to be linked to user's account
  - OIDC id_token uses this "effective identity"
- **If app or service does not set an effective identity policy, then the primary identity of the account is used as the effective identity for that app**

Authorization based on identity set

- **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**



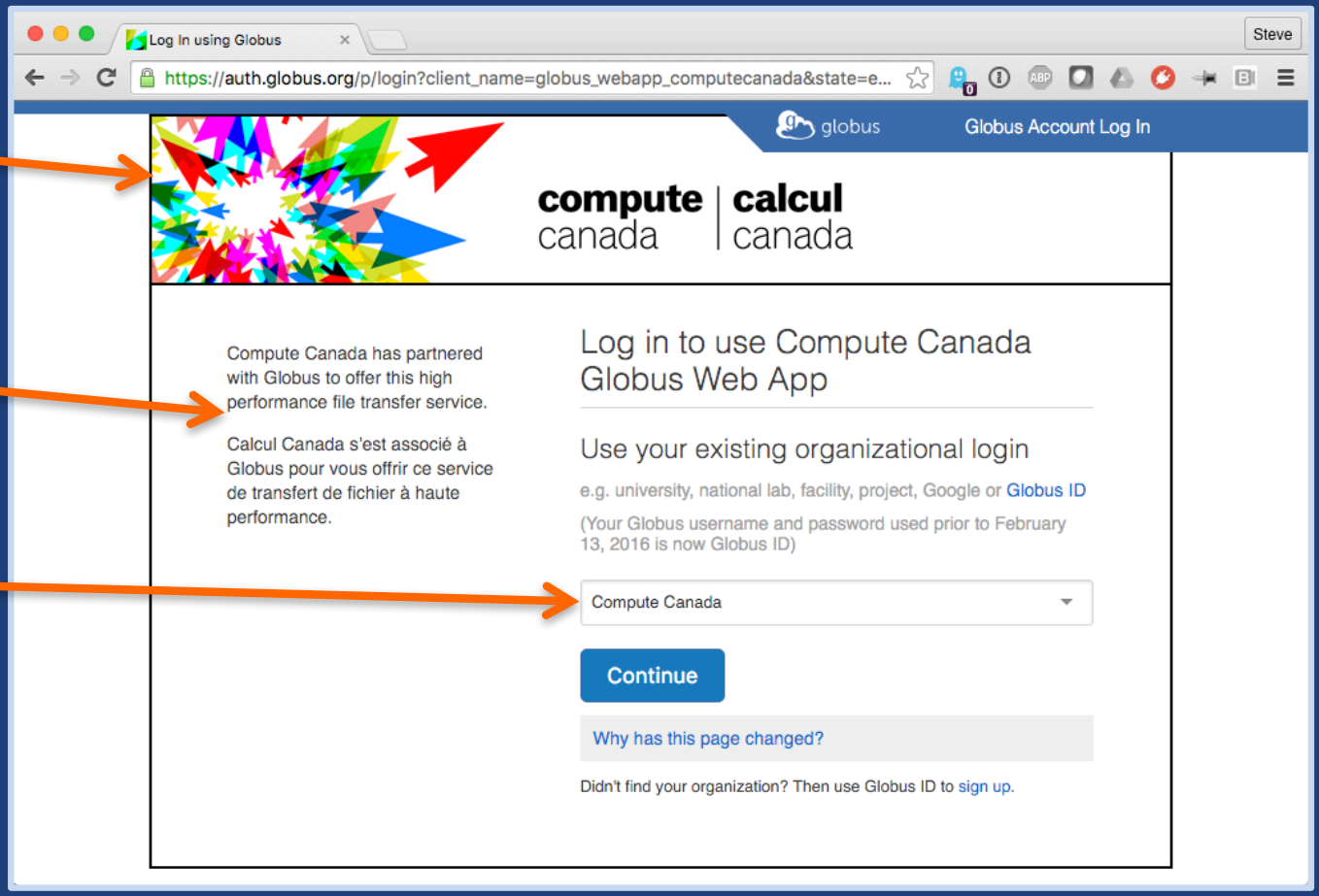
Branding

- **Can skin Globus Auth pages**

Header

Text

Default IdP



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.

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**