Useful developer links

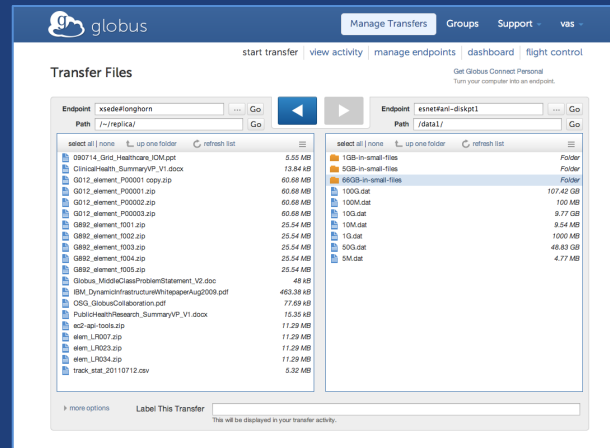> **github.com/globus**

> **docs.globus.org**

Globus serves as...

A platform for building science gateways, portals and other web applications in support of research and education

Use(r)-appropriate interfaces

Web App

CLI

```
(globus-cli) jupiter:~ vas$ globus
Usage: globus [OPTIONS] COMMAND [ARGS]...

Options:
  -v, --verbose              Control level of output
  -h, --help                 Show this message and exit.
  -F, --format [json|text]   Output format for stdout. Defaults to text
  --map-http-status TEXT     Map HTTP statuses to any of these exit codes:
                             0,1,50-99. e.g. "404=50,403=51"

Commands:
  bookmark       Manage Endpoint Bookmarks
  config         Modify, view, and manage your Globus CLI config.
```
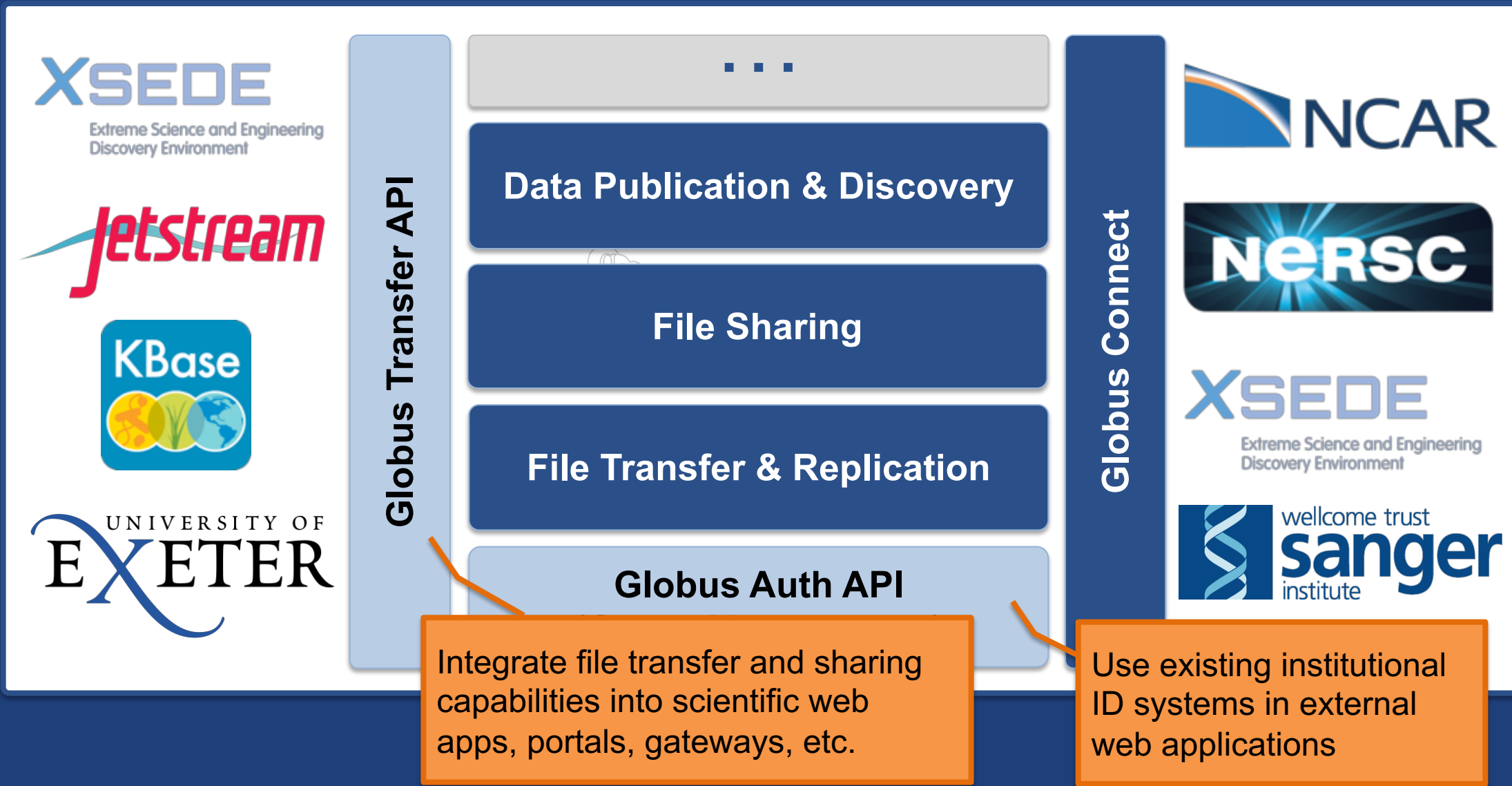
```
GET /endpoint/7026c6d4-0c84-11e8-a763-0a448319c2f8
PUT /endpoint/48640ab2-0c84-22b9-b571-bb4589ac34d4
200 OK
X-Transfer-API-Version: 0.10
Content-Type: application/json
...
```

Rest API

Globus Platform-as-a-Service

XSEDE
Extreme Science and Engineering
Discovery Environment

Jetstream

KBase

UNIVERSITY OF
EXETER

Globus Transfer API

. . .

Data Publication & Discovery

File Sharing

File Transfer & Replication

Globus Auth API

Globus Connect

NCAR

NeRSC

XSEDE
Extreme Science and Engineering
Discovery Environment

wellcome trust
sanger
institute

Integrate file transfer and sharing capabilities into scientific web apps, portals, gateways, etc.

Use existing institutional ID systems in external web applications

Globus Platform
Transfer API

Globus Transfer API

- **Globus Web App consumes public Transfer API**

- **Resource named by URL (standard REST approach)**
  - Query parameters allow refinement (e.g., subset of fields)

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

- **Requests authorized via OAuth2 access token**
  - Authorization: Bearer asdflkqhafsdafeawk

docs.globus.org/api/transfer

Globus Python SDK

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

- `globus_sdk.TransferClient` **class handles connection management, security, framing, marshaling**

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

globus.github.io/globus-sdk-python

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 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)
```

Walkthrough
Jupyter Notebook
jupyter.demo.globus.org

github.com/globus/globus-jupyter-notebooks

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

- **Activation: Binding a credential to an endpoint for login**
- **Globus Connect Server endpoints that use MyProxy or MyProxy OAuth 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**
- **If credentials exist, OK; otherwise response tells you how to activate**

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 (with optional sync level option)
  - Delete

- **Get** `submission_id`**, followed by submit**
  - Ensures 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: Transfer API - Data distribution

Modify Jupyter notebook to…

1. Find the endpoint ID for the "NYSERNet Share" endpoint
2. Bonus points: how many endpoints are currently associated with "NYSERNet"?
3. Make a directory for your files: /<your_name>
4. Get endpoint ID: "ESnet Read-Only Test DTN at Sunnyvale"
5. Transfer "/data1/50GB-in-medium-files" to your directory
6. Optional: Check if successfully transfered and delete files

Solution: Transfer API

- **Find ID for "NYSERNet Share" endpoint**

```
r = tc.get("/endpoint_search",
    params=dict(filter_fulltext="NYSERNet Share", limit=1))
```

- **Make a directory for your files: /<your_name>**

```
endpoint_id = nysernet_ep_id
endpoint_path = "/vas"
```

- **Transfer ESnet files to your directory**

```
source_endpoint_id = "db57ddde-6d04-11e5-ba46-22000b92c6ec"
source_path = "/data1/50GB-in-small-files/"
dest_endpoint_id = nysernet_ep_id
dest_path = "/vas/50GB-in-small-files/"
```

Solution: Transfer API

- **Check if successfully transfered and delete files**

```
If status == "SUCCEEDED":
    source_endpoint_id = "db57ddde-6d04-11e5-ba46-22000b92c6ec"
    endpoint_id = nysernet_ep_id
    path = "/vas/50GB-in-small-files/"
    r = globus_sdk.DeleteData(tc, endpoint_id, recursive=True)
    ddata.add_item(path)
    tc.endpoint_autoactivate(endpoint_id)
    submit_result = tc.submit_delete(ddata)
    print("Task ID:", submit_result["task_id"])
```
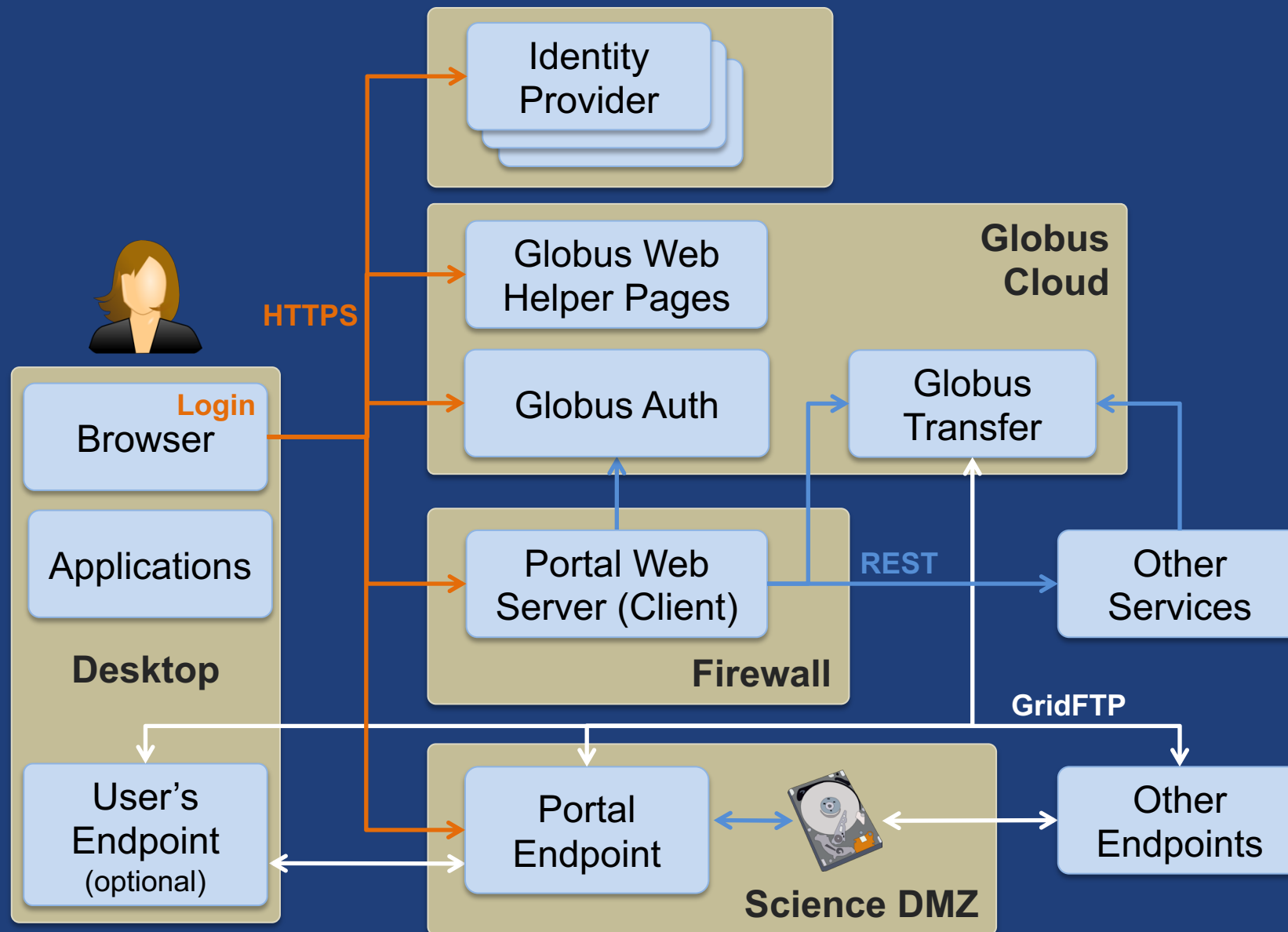
How can I do this in my [science gateway, data portal, web app, …]?

Prototypical research data portal

Maximizing the value of the Science DMZ

Legacy data portal/science gateway design

Border Router perfSONAR **Firewall** **Enterprise**

perfSONAR

Browsing path
Query path
Data path

10GE

Portal server applications:
- *web server*
- *search*
- *database*
- *authentication*
- *data service*

10GE **Portal Server**

Filesystem (data store)

WAN
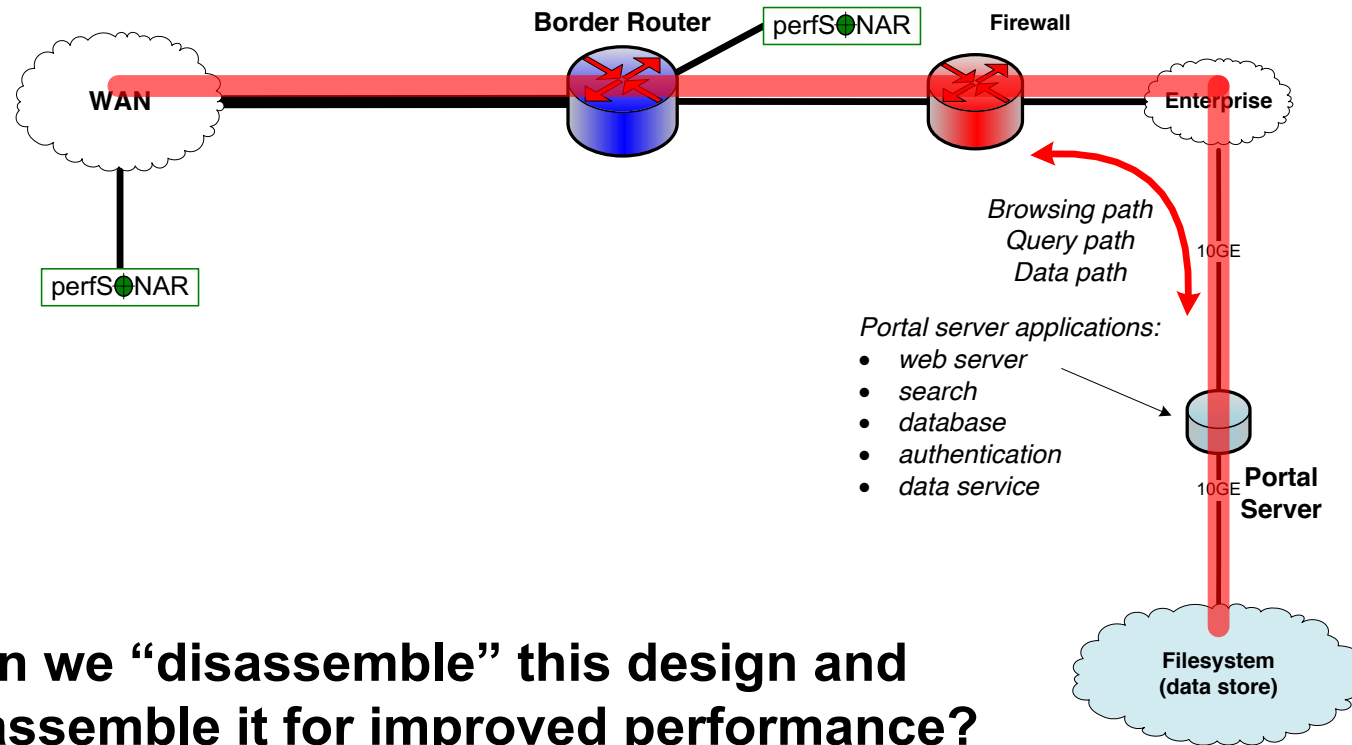
Can we "disassemble" this design and reassemble it for improved performance?

Modern data apps leverage the Science DMZ
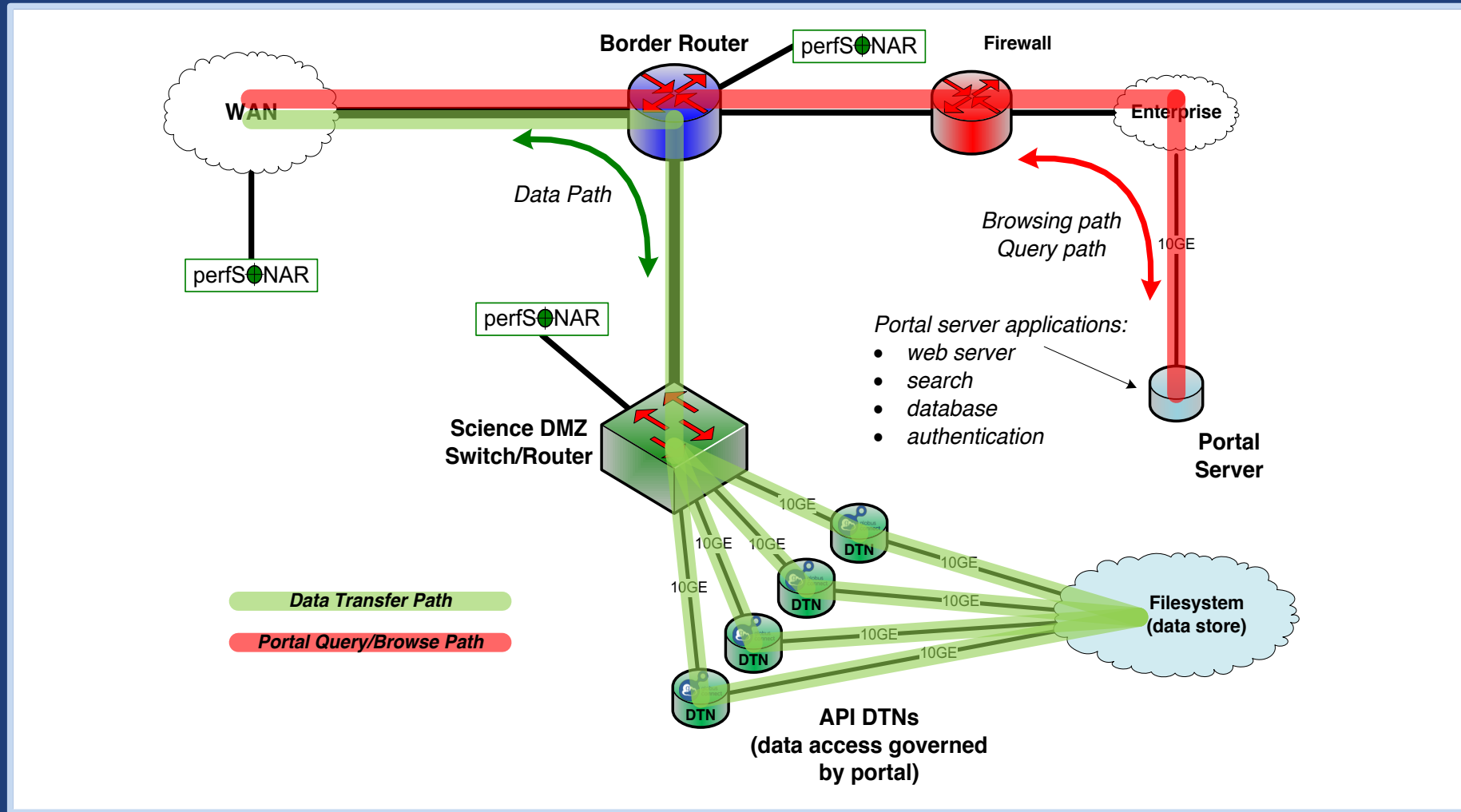
fasterdata.es.net/

Data Distribution: ARM Climate Research Facility

Analysis Workflow Integration: Wellcome Sanger

Globus Platform

Auth API

Globus Auth: Foundational IAM service

- **Enables login for diverse app ecosystem**
- **Simplifies creation/integration of apps, services**
- **Outsources mundane feature development**
- **Brokers authentication and authorization interactions**
- **Protects REST API communications**
- **No new identity required**
- **Employs least privileges security model**
- **Programming language and framework agnostic**

Based on widely used web standards

- **OAuth 2.0 Authorization Framework (a.k.a. OAuth2)**

- **OpenID Connect Core 1.0 (a.k.a. OIDC)**

- **Access via OAuth2 and OIDC libraries of your choice**
  - Google OAuth Client Libraries (Java, Python, etc.), Apache mod_auth_openidc, etc.
  - Globus Python SDK

docs.globus.org/api/auth

Fundamental Concepts

- **Scopes: APIs that client is requesting access to**
  - Scope syntax: OpenID Connect: openid, email, profile
  - urn:globus:auth:scope:<service-name>:<scope-name>

- **Consents: authorization client to access a service, within limited scope, on the resource owner's behalf**

Globus account

- **Globus Account = Primary identity + Linked Identities**
  - An identity can be primary on only one account
  - Identities can be linked to only one account

- **Account does not have own identifier**
  - An account is uniquely identified using its primary identity

Identity *id* vs. *username*

- **Identity *id***
  - Unique among all Globus Auth identities; 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

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)

Modern Research Data Portal

Portal (Client)

Globus Auth (Authorization Server)

Globus Transfer (Resource Server)

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
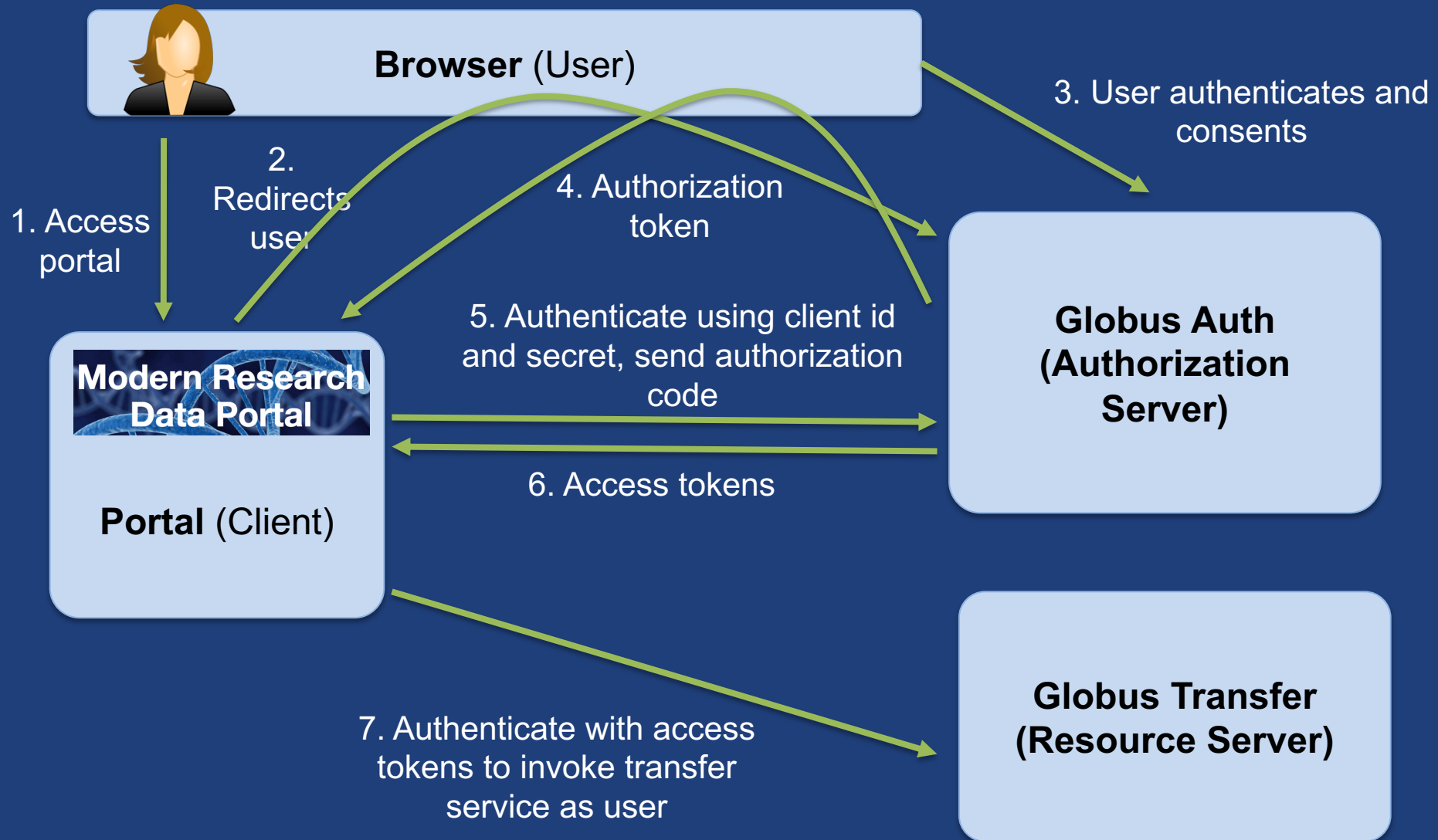
Use case: Native apps

- **Examples (any client that cannot keep a secret)**
  - Command line, desktop apps
  - Mobile apps
  - Jupyter notebooks

- **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

2. URL to authenticate

3. Authenticate and consent

4. Auth code

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)
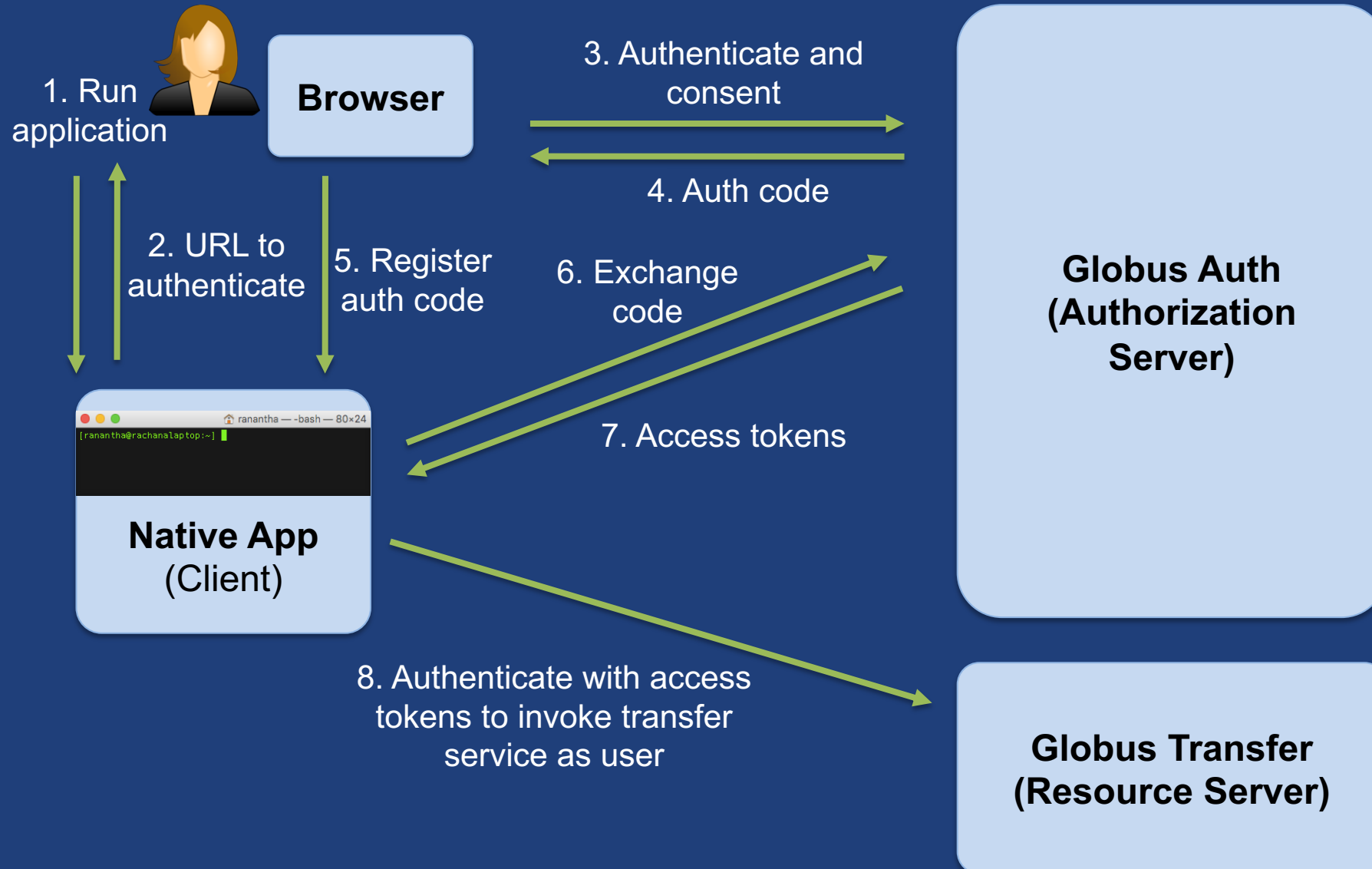
Use case: Apps that need long-lived access tokens

- **Examples**
  - Portal checking transfer status when user is not logged in
  - Running command line app from script

- **App requests refresh tokens**

- **Globus SDK:**
  - To get token: ConfidentialAppClient or NativeAppClient
  - To use tokens: RefreshTokenAuthorizer

Automation Example: Repeated replication

Recurring transfers
with sync option

Copy /ingest
Daily @ 3:30am

- **Using Globus CLI or SDK**

- **Meant to be run via cron or other task manager**

- **Native app grant**

Refresh tokens

- **For "offline services"**
  - e.g., transfer working on your behalf even when you are offline
- **Refresh tokens issued to client, in 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**

Refresh tokens

1. Run application

2. URL to authenticate

5. Register auth code

Browser

3. Authenticate and consent

4. Auth code

6. Exchange code, request refresh tokens

7. Access tokens and refresh tokens

Globus Auth
(Authorization Server)

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 service as user

App/Service
(Resource Server)

Demo: Native App/Refresh Tokens

github.com/globus/native-app-examples

- **See README for installation**

- **`./example_copy_paste.py`**
  - Copy paste code to the app

- **`./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

- **Every app is/has an identity in Globus Auth (<client_id>@clients.auth.globus.org)**

- **App registers with Globus to get client id/secret**
  - Native app cannot do this (no client_secret)

- **Uses Client Credential Grant**

- **Can use the client_id just like any other identity_id**
  - Sharing access manager role, permissions, group membership, etc.

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

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, e.g. set an ACL on a shared endpoint

Automation Example: Data distribution

Data distribution

Shared Endpoint

.../my_share
--/cohort045
--/cohort096
--/cohort127

- **Uses Auth and Transfer API via SDK**

- **Native app grant**

- **Client credential grant**
  - Portal or service
  - Permission for the client id

Client credential grant

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)
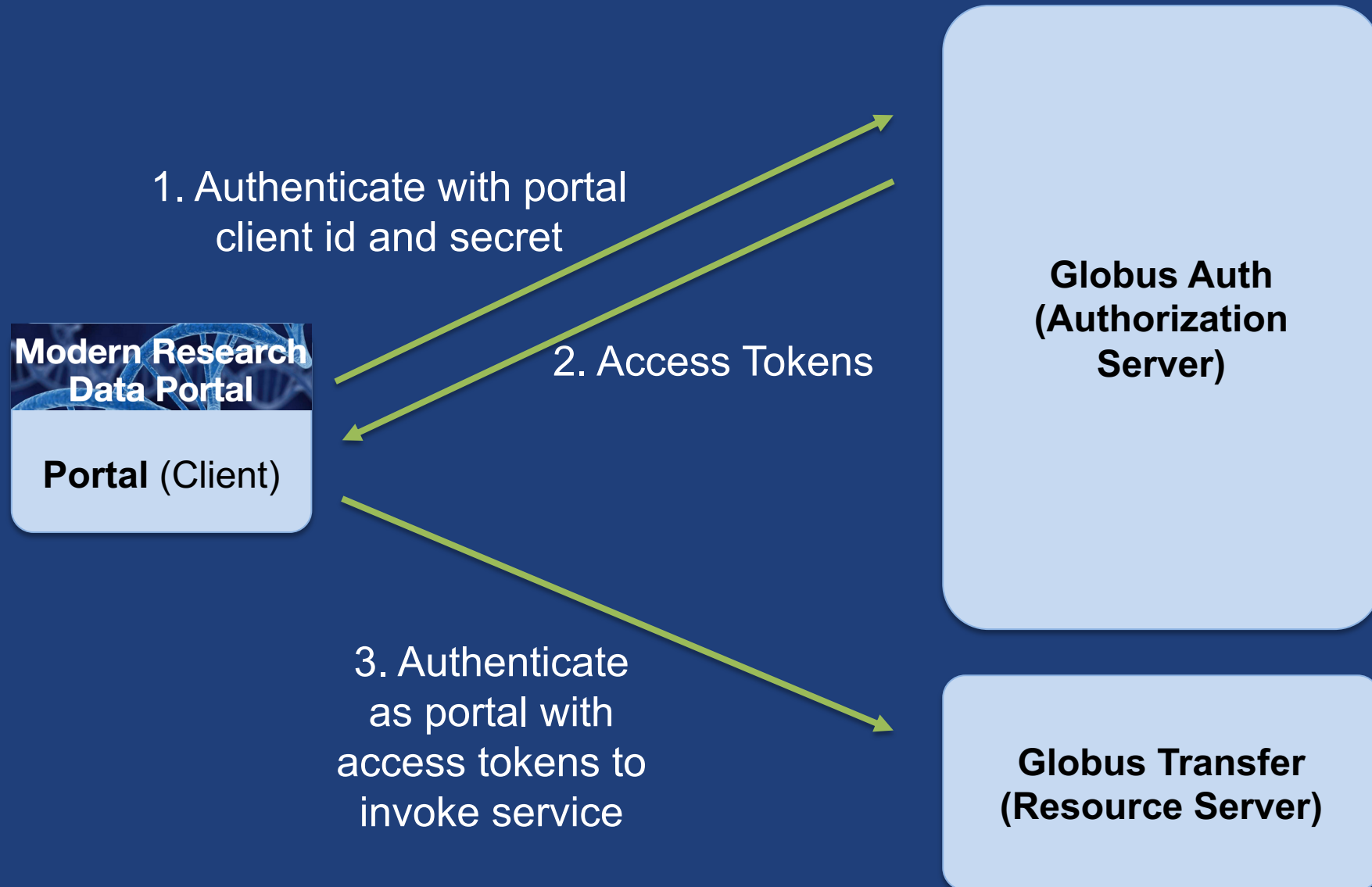
Exercise: Automation using the REST API

- **Use the share_data Python script to share files with your neighbor**
  - Modify the script to:
    - Create a shared endpoint
    - Add access permissions to the shared endpoint for your neighbor
  - Move data to the shared endpoint
  - Check that your neighbor received the sharing notification

Globus Helper Pages

- **Globus 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

Globus PaaS developer resources

Python SDK

globus.github.io/globus-sdk-python/

globus-sdk-python 0.2.5 documentation »   next | modules | index

Table Of Contents

Globus SDK for Python (Beta)
Installation
Basic Usage
API Documentation
Lice...

Next

High...

This Page

Show Source

Quick search

[   ] Go

Globus SDK for Python (Beta)

This SDK provides a convenient Pythonic interface to Globus REST APIs, including the Transfer API and the Globus Auth API. Documentation for the REST APIs is available at https://docs.globus.org.

...ce, supporting only GET, PUT, POST, and DELETE operations, and a high level interface providing helper

...obus/globus-sdk-python.

Installation

The Globus SDK requires Python 2.6+ or 3.2+. If a su...

The simplest way to install the Globus SDK is using th...
installations:

```
pip install globus-sdk
```

This will install the Globus SDK and it's dependencies...

Bleeding edge versions of the Globus SDK can be in...

```
git checkout https://github.com/globus/glob...
cd globus-sdk-python
python setup.py install
```

Basic Usage

Modern Research Data Portal

LOGIN | SIGN UP

Modern Research Data Portal

It's how research data management is done!

Sample Application

Jupyter Notebook

Requirements

- You need to be in the tutorial users group for sharing: https://www.globus.org/app/groups/50b6a29c-63a...
- Installed Globus Python SDK

```
In [15]:  from __future__ import print_
          tutorial_endpoint_1 = "ddb59aei-6d04-11e5-ba46-22000b92c6ec"      # endpoint "Glo
          tutorial_endpoint_2 = "ddb59af0-6d04-11e5-ba46-22000b92c6ec"      # endpoint "Glo
          tutorial_users_group = "50b6a29c-63ac-11e4-8062-22000ab68755"     # group "Tutori
```

Configuration

First you will need to configure the client with an OAuth2 access token. For the purpose of this tutorial, you ca...
website. Click the "Jupyter Notebook" option and copy the resulting text below, or click on "Globus CLI" and...

```
In [16]:  transfer_token = None # if None, tries to get token from ~/.globus.cfg file
```

docs.globus.org/api   **github.com/globus**

Support resources

- **Globus documentation: docs.globus.org**

- **Helpdesk and issue escalation: support@globus.org**

- **Customer engagement team**

- **Globus professional services team**
  - Assist with portal/gateway/app architecture and design
  - Develop custom applications that leverage the Globus platform
  - Advise on customized deployment and integration scenarios

Join the Globus community

- Access the service: **globus.org/login**

- Create a personal endpoint: **globus.org/app/endpoints/create-gcp**

- Documentation: **docs.globus.org**

- Engage: **globus.org/mailing-lists**

- Subscribe: **globus.org/subscriptions**

- Need help? **support@globus.org**

- Follow us: **@globusonline**