Globus Auth: A Research Identity and Access Management Platform

Steven Tuecke, Rachana Ananthakrishnan, Kyle Chard, Mattias Lidman,
Brendan McCollam, Stephen Rosen, and Ian Foster
Computation Institute
The University of Chicago and Argonne National Laboratory, Chicago, IL 60637, USA

***Abstract*—Globus Auth is a foundational identity and access management platform service designed to address unique needs of the science and engineering community. It serves to broker authentication and authorization interactions between end-users, identity providers, resource servers (services), and clients (including web, mobile, desktop, and command line applications, and other services). Globus Auth thus makes it easy, for example, for a researcher to authenticate with one credential, connect to a specific remote storage resource with another identity, and share data with colleagues based on another identity. By eliminating friction associated with the frequent need for multiple accounts, identities, credentials, and groups when using distributed cyberinfrastructure, Globus Auth streamlines the creation, integration, and use of advanced research applications and services. Globus Auth builds upon the OAuth 2 and OpenID Connect specifications to enable standards-compliant integration using existing client libraries. It supports identity federation models that enable diverse identities to be linked together, while also providing delegated access tokens via which client services can obtain short term delegated tokens to access other services. We describe the design and implementation of Globus Auth, and report on experiences integrating it with a range of research resources and services, including the JetStream cloud, XSEDE, NCAR's Research Data Archive, and FaceBase.**

I. Introduction

Developers of research applications and services face two major infrastructure challenges: (1) providing secure and reliable identity and access management (IAM) functionality; and (2) integrating, in a manner convenient for users, with other services that have been developed by independent parties. The difficulties of addressing these challenges has resulted in a fragmented ecosystem of research services. For example, few scientific web applications and science gateways leverage federated identity systems such as InCommon [1]. Instead, each service provider commonly cobbles together its own identity management solution. The result is often applications with limited functionality (due to the high cost and expertise of implementing sophisticated IAM functionality), little integration (due to different IAM approaches), high development and maintenance costs (due to each group creating their own partial solutions), poor user experience (due to inconsistent and incompatible IAM functionality), and even poor security (due to buggy implementations).

Globus Auth is platform-as-a-service (PaaS) that addresses these challenges, with the goal of streamlining the creation, integration, and use of advanced research services [2]. In brief, it allows research service providers to outsource identity, authentication, credential, and authorization management functions to a cloud-hosted, professionally managed service. In so doing, providers gain five major benefits. First, they gain access to sophisticated IAM functionality that would be difficult for them to implement themselves. Second, they gain integration with other system and services, based on standards such as OAuth 2 [3], OpenID Connect [4], SAML [5], and X.509. Third, they reduce implementation and operation costs: complex in-house code can be replaced with simple REST API calls to a professionally operated service. Fourth, they improve user experience by delivering high-quality, consistently presented IAM capabilities and interfaces. And fifth, they improve the security of their system by using IAM functionality implemented and operated by dedicated security professionals.

The rest of this paper is as follows. We describe the use cases that motivate the Globus Auth design in Section II. In Section III we present the Globus Auth model and its use of authentication and authorization standards. In Section IV we review how Globus Auth is implemented, deployed, and operated. In Section V we present five research services that build upon Globus Auth. Finally, in Section VI we describe related work before summarizing our contributions in Section VII.

II. Motivating Use Cases

Globus Auth addresses important use cases that arise in scientific settings. Specifically, it brokers authentication and authorization interactions, enables identity linking, supports single sign on across scientific services, and enables delegated access to external services. We focus on the advantages that Globus Auth brings to developers and users of scientific services [6]—an increasingly common model for delivering scientific capabilities to a broad community via well defined, Internet accessible services.

Identity broker: Developers of scientific services want to allow users to authenticate using existing identities (e.g., campus accounts). However, implementing such support requires registering clients with a multitude of identity providers, developing support for a number of different authentication protocols (e.g., SAML, OAuth 2, OpenID), addressing subtle differences that exist between implementations due to non-prescriptive specifications, and supporting these implementations over time as changes are inevitably made. Ideally,

developers would like to leverage a reliable and secure service that can broker different identity providers while offering a single stable API from which these identities can be used.

Identity federation: Researchers are accustomed to requiring an ever-growing number of identities to perform their daily tasks. For example, using institution identities to access local storage resources, Google accounts to access Google Docs, and national cyberinfrastructure credentials to access HPC resources. Instead, users want to be able to link their identities, proving ownership once (or infrequently), and then being authorized to perform actions based on this set of identities. For example, consider a common authorization model that uses group membership to manage access to resources. A user should, irrespective of which identity they have used for authentication, be able to perform the roles granted to any identity in that set.

Single sign on: Researchers now have access to many domain-specific and general scientific services. However, these services typically operate in silos in which identities, groups, data, analyses, and other state are not shared between services. Thus, artificial barriers exist between services. Instead we envision a global scientific ecosystem in which many different scientific services build upon a common platform [7]. As a first step towards this goal, methods are required that enable state to be easily shared between services, so that users can use a single identity across services (i.e., single sign on), and service developers can unambiguously refer to users.

Delegated access: Scientific services are increasingly built upon a suite of external services: i.e., as "mash ups." For example, a climate modeling service may outsource to external services tasks such as managing users, groups, data storage, and computation; the climate modeling service implementation then needs to address only domain-specific issues. This approach has the benefit of allowing developers to deliver advanced capabilities at a fraction of the cost of developing them entirely from scratch. However, for such applications to work, they must be able to make requests to external services on behalf of users, with user information passed down so that user authorizations can be enforced at remote services. We thus require sophisticated authorization models via which users can allow services to transfer to other services the authority to access their state or to perform actions on their behalf. That is, they require secure authorization delegation mechanisms.

III. Globus Auth

Globus Auth provides a set of features that enable identities to be asserted from a range of identity providers, offers various standard interfaces for integration in third-party applications and services, and enables linking of identities to facilitate federated login. Here, we describe the unique features of Globus Auth.

A. General model

Globus Auth builds upon the OAuth 2 and OpenID Connect specifications to deliver identity and access management as a platform. The high level model is depicted in Fig. 1. Here we outline the general Globus Auth model and describe how it relates to the OAuth 2 and OpenID Connect specifications. Terms defined in these specifications are denoted with italics.

A *protected resource* (or simply *resource* in this paper) is something that can be addressed via a URL, and is accessible to authorized clients via HTTPS methods (e.g., a REST API).

Globus Auth is an *authorization server*. It issues an *access token* to a *client* after successfully authenticating the *resource owner* and obtaining authorization for the client to access *resources* provided by a *resource server*. The resource owner is typically an end-user, who authenticates to a Globus Auth-managed Globus account using an identity issued by one of an extensible set of (federated) identity providers supported by Globus Auth. The resource owner authorizes (i.e., *consents*) that the client can request access to the resource server on the resource owner's behalf within a limited *scope*. The client might be an application (e.g., web, mobile, desktop, command line), or it may be another service, as described below.

When a client makes a request to a resource server, it presents the access token as part of the request (in the HTTP Authorization header), to demonstrate that it is authorized to make the request.

Globus Auth can act as the authorization server to an extensible set of resource servers. All Globus [8] services, such as the Globus data management [9] and groups services [10], are resource servers that use the Globus Auth authorization server. Third parties can also create their own resource servers that rely on the Globus Auth authorization server in exactly the same way as Globus services. This broad applicability is why we call Globus Auth a foundational service: it provides a platform for an extensible, integrated ecosystem of resource servers and their clients.

The OAuth 2 specification states that "[t]he interaction between the authorization server and resource server is beyond the scope of [the OAuth 2] specification." Globus Auth fills this gap by defining a REST API that allows a resource server, upon receiving a request with an access token from a client, to verify that the access token is valid and intended for use with this resource server, and to query for additional information related to that access token such as the client identity, the scope, the resource owner's identity, and other identities linked to that resource owner's identity, which the resource server can use to make authorization decisions for the request. Globus Auth leverages the OAuth 2.0 Token Introspection specification (RFC 7662) [11] for this interaction.

Globus Auth also plays the role of an OAuth 2 resource server, allowing clients to access Globus Auth-managed resources, such as identities, and access tokens. For example, clients acting on behalf of end-user resource owners, can be clients to the Globus Auth resource server, to access and manage the end-user's Globus account-related information. When a resource server receives a request from a client with an access token, that resource server assumes the role of a (different) client to the Globus Auth resource server, in order to validate the access token. In this situation, the resource server uses its own client id and client secret when making requests

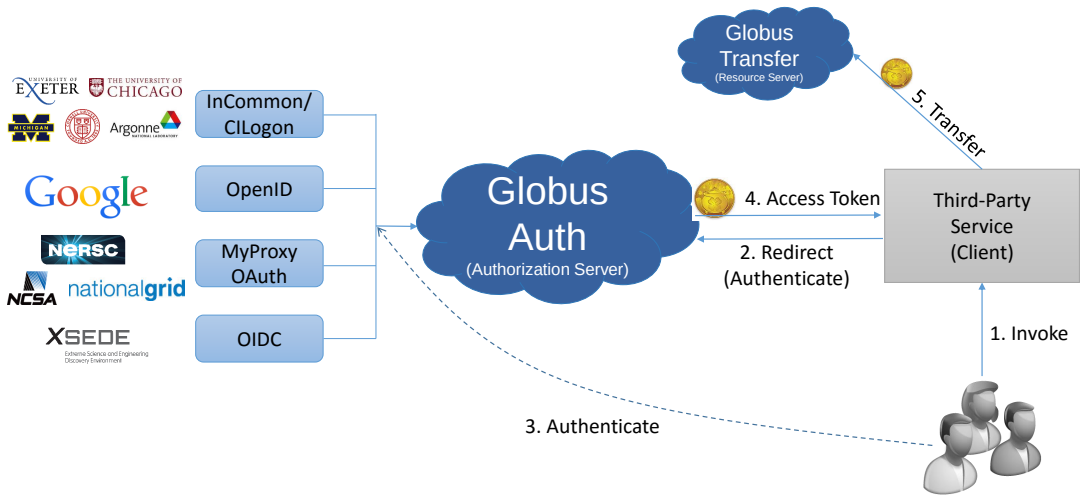

Fig. 1: The Globus Auth model. A user authenticates with a third-party service using one of the supported Globus Auth IDPs. Globus Auth presents a page allowing the user to consent the third-party service to access other resources. Globus Auth presents the third-party service a delegated access token that can be used to both authenticate the user and also access other approved services as that user.

to the Globus Auth resource server. An identity provider can act as a client to the Globus Auth resource server to provision and manage identities within the identity provider's namespace. In this situation, the identity provider uses its own client id and client secret (established previously through a registration process with Globus Auth) when making requests to the Globus Auth resource server.

B. Identities

A Globus Auth identity is a unique name (e.g., `user@example.org`), issued by an identity provider (e.g., an institution or Google), for which a user or client can prove possession via an authentication process (e.g., presenting a password to the identity provider). Globus Auth manages the use of identities (e.g., to login to clients and services), their properties (e.g., associated contact information), and relationships between identities (e.g., allowing login to an identity by using another linked, "federated" identity).

Globus Auth neither defines its own identity usernames nor verifies authentication (e.g., via passwords) with identities. Rather, it acts as an intermediary between external identity providers and clients and services that want to leverage identities issued by those providers.

Globus Auth assigns each identity that it encounters an identifier (`id`): a universally unique identifier (UUID) that is guaranteed to be unique among all Globus Auth identities, and that will never be reused. This identifier is what resource servers and clients should use as the canonical identifier for a Globus Auth identity. Associated with each id is an identity provider (`identity_provider`), a name given to the identity by the provider, and other provider-supplied information such as display name and contact email address (see Listing 1).

The identity `username` is a (somewhat) human friendly string, such as an email address or InCommon `eduPersonPrincipleName`, which is guaranteed by Globus Auth to be unique at any point in time. However, because some identity providers (e.g., InCommon) reuse identity usernames (typically with a hiatus between uses), a given identity username may map to different identity ids over time. In such cases, Globus Auth uses a unique identifier (`provider_specific_id`) provided by the identity provider (e.g., InCommon `eduPersonTargetedID`) to disambiguate, and ensure that at any given time there is a one-to-one mapping between an identity `username` and an identity `id`.

Listing 1: Globus Auth Identity

```
username: user@institution.edu
id: ab2312e23-8a34-bd230bca023120ab
identity_provider: institution.edu
display_name: User Name
email: user@institution.edu
```

If Globus Auth encounters an identity username that has been reused (i.e., same identity `username`, different `provider_specific_id`), it will invalidate the old identity and create a new Globus Auth identity uniquely associated with that identity username. Conversely, if Globus Auth encounters an existing identity where the identity `username` has changed for a given `provider_specific_id` (e.g., the user changes their name), it will update the identity `username` while retaining the same Globus identity `id`. Thus, at any point in time, the relationship between identity `username` and Globus Auth identity is unique, and a Globus Auth identity `id` can be relied on to always refer to the same identity.

C. Identity providers

Globus Auth supports an extensible set of identity providers that may employ a variety of identity naming and authentication approaches.

1) Registration with Globus Auth: Each identity provider supported by Globus Auth must register with Globus Auth. (Currently this registration is an out-of-band process, but in the future it can be automated via the Globus Auth API.) At time of registration, Globus Auth establishes a client id and client secret for the identity provider, to be used to allow the identity provider to authenticate as a client to Globus Auth. Identity provider clients can use certain Globus Auth interfaces, such as identity provisioning.

Each identity provider must register either a web browser based authentication protocol (e.g., OpenID Connect, SAML), or a non-browser based protocol (e.g., LDAP, Kerberos, SAML ECP), or both. If an identity provider registers only a non-browser based protocol, Globus Auth will provide a browser based interface for this identity provider. If an identity provider registers only a browser based protocol, some Globus Auth OAuth 2 grant types will not be possible with this identity provider (e.g., Resource Owner Password Credentials Grant), limiting the use of this provider's identities to only browser-based applications.

2) Identity provider namespaces: Each identity provider has one or more namespaces in which it can exclusively issue identity usernames.

An identity provider that issues "user@provider" names is constrained to issuing identities with one or more specific domain names. For example, The University of Chicago's identity provider, is the only provider that can issue identity usernames with a provider domain of "@uchicago.edu" (e.g., "johndoe@uchicago.edu"). Note that subdomains are distinct namespaces from their parent domain. For example, "@uchicago.edu" and "@ci.uchicago.edu" are distinct namespaces, from potentially different providers.

Some identity providers use email addresses as their usernames. For example, an identity provider restricted to issuing identities with names of "*@provider.org" may issue an identity with the name "johndoe@uchicago.edu@provider.org," but not "johndoe@uchicago.edu."

3) Identity and account provisioning: An identity provider, acting as a client to Globus Auth, may explicitly provision its own identities into Globus Auth through the API. The identity must include an identity username, and may include various other fields such as email address, display name, etc.

When a user logs into Globus Auth using an identity that is not associated with a Globus account (i.e., it is not a primary identity or linked identity of any account), either a Globus account must be created with this identity as the account's primary identity, or this identity must be linked to an existing account's primary identity. For some identity providers, when an unlinked identity authenticates to Globus Auth, an account will automatically be created with this identity as the primary. For other identity providers, Globus Auth will prompt the user to create an account or link the identity with another account.

4) Supported identity providers: Globus Auth currently supports a range of identity providers including GlobusID, OpenID Connect and Google identity providers.

GlobusID (Globus legacy usernames): Globus previously required that users create an explicit Globus account with a unique Globus username and password. This is no longer required with Globus Auth. Rather, Globus usernames are now managed and issued by a separate service: the GlobusID identity provider. GlobusID identities are issued under the identity provider domain namespace of "@globusid.org." This identity provider has no special status with Globus Auth: it is just another identity provider.

OpenID Connect: Globus Auth can act as a client to any standard OpenID Connect identity provider. The Globus Auth identity username for OpenID Connect identities will be the `sub` claim from the ID token issued by that identity provider, suffixed with DNS name of the OpenID Connect server as the provider domain. For example, if an OpenID Connect server running at "example.org" issues an ID token with a `sub` claim of "joeuser," the Globus Auth identity username is "joeuser@example.org." OpenID Connect identity providers can optionally register to follow Google's conventions on use of the `sub`, `email`, and `email_verified` claims, which Globus Auth uses as described below.

Google: While Google uses OpenID Connect (with some extensions), it is handled as a special case by Globus Auth. The Google identity provider can issue identities for any email address, and by default, such identities will have a Globus Auth identity username of the email address (i.e., the value of the Google-issued OpenID Connect ID token `email` claim), with a "@accounts.google.com" provider domain: for example, "user@uchicago.edu@accounts.google.com." Globus Auth only accepts Google-issued identities for email addresses that it has verified (i.e., Google-issued ID token has an `email_verified` claim with the value "true"). Globus Auth uses the value of the Google-issued ID token `sub` claim, as a provider-specific unique identifier for the identity. However, Google is also the exclusive issuer of identities for certain domains, such as "@gmail.com" and certain app domains registered with Globus Auth. For these pre-defined domains, Globus Auth does not add "@accounts.google.com" to the identity username.

Email addresses: Globus Auth treats email addresses as a special type of identity, where the identity's username is the email address (without an additional provider domain), and authentication of that username is done using the common email verification technique of sending an email to the address containing a secret that the user must validate via a web authentication/verification form. Note that due to identity provider namespacing, as described above, Globus Auth will never allow an email address identity with a domain name issued by a registered identity provider. For example, if the University of Chicago identity provider owns the "@uchicago.edu" namespace, "user@uchicago.edu" must be authenticated using the University of Chicago identity provider, and not simply via email address verification. If a

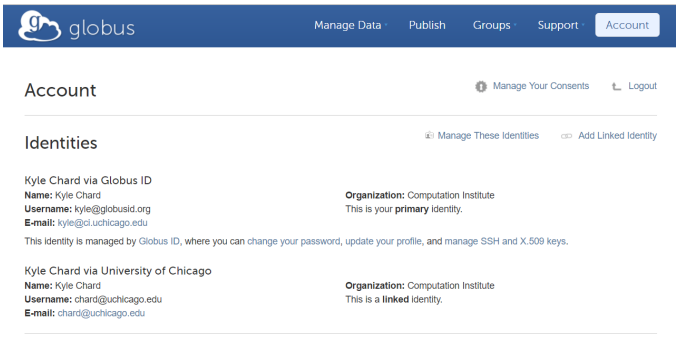

Fig. 2: Linked Globus Auth identities.

new identity provider is registered with an exclusive provider domain for which email address identities were previously issued, then Globus Auth will automatically change the provider of such identities to the new identity provider.

D. Linking identities

Globus Auth enables the creation of a "Globus account" using any identity. A Globus account is not an identity, rather it represents a set of identities that belong to the same individual. Fig. 2 shows an example of a Globus account with several linked identities.

A Globus account is a set of identities comprising a primary identity and a number of other identities linked to that primary identity. An identity can be the primary identity of at most one Globus account. Identity linking allows for authentication via one identity to imply login to a Globus account with a different primary identity (i.e., federated identity login). Note that a Globus account is not an identity itself. An account does not have its own name. Rather, a Globus account is identified by its primary identity. Similarly, profile information and other metadata is tied to identities, not to accounts.

Clients and services should grant access to resources on the basis of identities (specifically, identity ids) and their associated attributes (e.g., group memberships, organization affiliations), not accounts. Login to a Globus account, via its primary identity or one of its linked identities, implies login to the account's primary identity and all identities linked to that account's primary identity. In other words, login to a Globus account potentially grants access to all resources accessible via all identities linked to that Globus account's primary identity. In future work, Globus Auth will support "level of assurance" policies to further constrain the access(es) that are allowed by the set of linked identities.

E. Resource server registration

A resource server must register with Globus Auth before it can use Globus Auth as an authorization server. During registration, Globus Auth establishes a client identifier and client secret for the resource server. These credentials allow the resource server to authenticate to Globus Auth in order to obtain and validate access tokens.

A resource server, during registration, can set a number of configurable properties such as a unique name, a restricted set of allowable identity providers, its scopes, and scopes it uses from other resource servers. The resource server must register a unique resource server name, a DNS name that is uniquely identifiable. The resource server name is used as part of the scope URNs for its resources. Resource servers may optionally restrict the set of permitted identity providers. In this case, users must have an identity issued by the selected identity providers, to access the resource server. During the authentication process, the resource server will request a specific *effective identity* associated with the access token. It will then be given the user's identity from this provider, even if the user has a different primary identity.

A resource server defines a set of scopes for itself, each corresponding to a subset of that resource server's resources or functionality. For example, a service could offer separate scopes for *starting*, *viewing*, and *managing* tasks. Each scope has a Globus Auth-issued URN that is unique across all scopes on all resource servers, and is never reused. For example:

```
urn:globus:auth:scope:example.com:tasks:start
urn:globus:auth:scope:example.com:tasks:view
urn:globus:auth:scope:example.com:tasks:manage
```

Clients request an access token that authorizes use of a specific set of scopes (and thus resource servers). A resource server may choose to offer just a single scope that grants full access; more limited scopes allow the resource server to protect resources better by offering more limited rights.

Finally, as described in the following section, a resource server may also define a set of scopes that it will use as a client to other resource servers.

F. Access delegation

The OAuth 2 specification defines how to obtain and use access tokens for interactions between a client and a resource server, within a specified scope. However, it does not prescribe an approach to allow delegated token usage. For example, consider a resource server (RS1) that receives a request from a client (C1) using a request access token (AT1), and that then wants to act as a client (C2) to another resource server (RS2), in order to help fulfill the request. The OAuth 2 specification does not specify what access token should be used in the request from C2 to RS2.

This scenario arises frequently within research IT scenarios. For example, a user of a web application client wants to submit a request to a workflow management service to run a workflow. The workflow resource server, in turn, wants to submit a request to the Globus data sharing service [12] to access data from a shared endpoint for use in the workflow. In order to satisfy the request, the Globus data sharing resource server must, in turn, make a request to the Globus groups service to determine the groups of which the user is a member, based on that user's linked identities, in order to determine the user's permissions on that shared endpoint. In this scenario, we call the Globus groups service a *dependent resource server* to the Globus data sharing resource server, and the Globus data

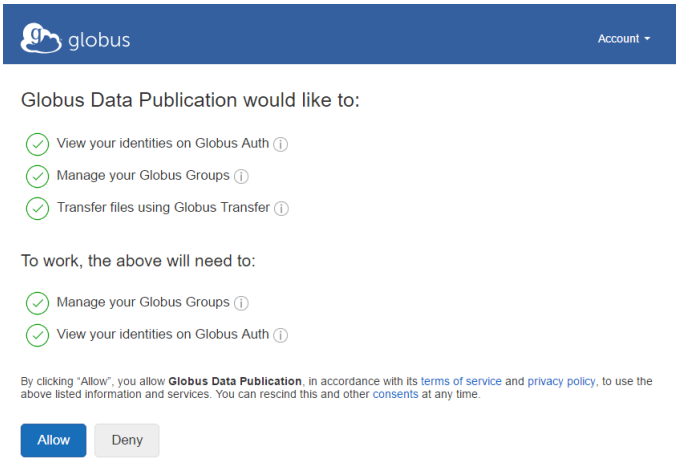

Fig. 3: The Globus Auth consent web interface. In this case the client (Globus Data Publication) is requesting delegated access tokens to retrieve identities managed by Globus Auth, manage groups using Globus groups, and perform transfers using Globus Transfer.

sharing resource server is a *dependent resource server* to the workflow service.

The Globus Auth authorization server provides an API for its resource servers. This API allows a resource server to request new *dependent access tokens* based on the access token that it received from its client. These dependent access tokens can be used to access downstream dependent resource server scopes. The Dependent Token Grant API supports access token delegation for such service invocation chains. When first accessing the client via the Globus Auth authentication workflow, the user is asked to consent the sharing of tokens with the requesting client, as shown in Fig. 3.

G. Groups

Groups are an important component of any authorization model and are therefore used in conjunction with Globus Auth by several external services. While groups management is beyond the scope of Globus Auth, we have extended Globus groups [10] to support linked Globus Auth identities. In this case, linked identities enable a user to perform actions permitted by the union of memberships that their set of identities dictate. Applications can leverage Globus groups to define course-grained authorizations by assessing users' memberships. Globus groups is offered like any other Globus Auth resource server and delegated access to a user's groups can be obtained via a requested scope.

IV. Implementation and Operations

The Globus Auth service is comprised of standards compliant OAuth 2 endpoints including custom extensions, API endpoints for querying identity and identity set information, and a user-facing web interface for managing clients, consents, and identity linkages.

The service is written in Python using the Pyramid web framework. It uses an Amazon Web Services (AWS) Reliable Database Service (RDS) PostgreSQL database for storing state. RDS was chosen for its proven performance, scalability and reliability. Globus Auth has been used in production since February 2016, servicing more than 40,000 Globus users with no downtime to date.

Globus Auth has extremely high requirements on security, availability and data integrity. This section describes important practices and patterns used during development and operations to ensure those requirements are met.

A. Implementation

The Globus Auth implementation comprises three major components: a web interface and API, the application, and a stateful database.

The Globus Auth web interface provides the ability for users to authenticate (by selecting an identity provider), consent to client access, manage an account, and add and remove linked identities. A subset of these capabilities are also available via REST APIs for integration in third-party applications.

The application implements client functionality to integrate with various identity providers, for example acting as an OpenID Connect client to Google. It provides the logic to securely register and manage identity providers and a modular framework via which new identity provider types can be added. The application manages all resources in the system such as identities, accounts, and clients. It provides support to register and configure clients including defining scopes, identity restrictions, and customizing the authentication workflows. Finally, the application also implements server OAuth 2 and OpenID Connect APIs for integration in third-party applications.

The database stores all Globus Auth state including identities, accounts, clients, consents, and identity providers. Database constraints and triggers are used extensively in the data model. For example, a database trigger ensures that if an identity is unlinked, consents that depend on that identity are automatically revoked. This mechanism reduces the risk of application-level bugs and also eliminates the risk that administrator tools that bypass the user application interfaces neglect to implement important side effects.

B. Standards compliance

We have worked hard to ensure that Globus Auth is compliant with the OAuth 2 standard and related specifications. This compliance ensures that external developers can use any one of a number of off-the-shelf libraries to integrate their services with Globus Auth. Custom extensions in Globus Auth, such as returning multiple tokens in a single response, have been designed so that they do not violate existing standards.

A challenge when working with OAuth 2 is that the specification is, in places, vague, and that existing implementations either do not support every aspect of the specification or implement some aspects incorrectly. Thus we have submitted patches to OAuth 2 libraries to implement specification behaviors on which we rely. In addition, we have adapted Globus Auth to support commonly used but non-standard behavior: for example, we support both scope lists that are space-separated

(as mandated by the specification) and comma-separated (the behavior of several widely used implementations).

C. Security

All Globus Auth API endpoints follow a pattern whereby anything that is not explicitly allowed is forbidden. The first part of this model is *ingress security*. Every incoming request must be authenticated with valid credentials before being allowed into a lower level of the application. Business logic permission checks are then performed. Finally, before an object (e.g., an identity) is rendered into a response, an *egress security check* is performed to ensure that the object has been explicitly marked as viewable by the authenticated party.

We have invested considerable effort into ensuring that Globus Auth is protected from both general and targeted attacks. We take care to manage private data and limit both internal and external access to Globus Auth resources. All credentials that Globus Auth is responsible for validating, such as access tokens and client secrets, are only ever stored in a hashed state. Further, any potentially sensitive data stored in the database are encrypted with a secret shared amongst the application servers. Where possible, Globus Auth stores signatures and message digests for the credentials that it issues, rather than the credentials themselves. Secrets are generated using entropy sources suitable for cryptographic use, and signatures are generated using SHA256 or SHA512. Finally, all sensitive data used in deployment are encrypted in our configuration management system with private keys belonging to the application servers.

Globus Auth access tokens are HMAC-signed and contain an expiration timestamp encoded within the token itself. Thus, Globus Auth can reject invalid or expired access tokens without querying the database. Globus Auth also supports an extensible set of token versions, allowing seamless key rotation and token format changes.

D. Testing and development model

Globus Auth features an extensive automated test suite, covering not only expected behavior but also a wide number of error cases such as misbehaving client applications and reliant services being unavailable. The full test suite is run on every single build of the system, and no build can be promoted beyond a "sandbox" environment without every test passing.

Globus Auth follows a rigorous development and testing model, where code changes progress through various environments before being deployed in production. Every code change is first reviewed by another member of the team. Next it is deployed in a sandbox environment where the development team can perform integration testing. Releases are then deployed on a staging environment in which a team of test engineers conduct a range of manual test cases. Finally, the fully tested code is deployed to production servers.

E. Migration and legacy support

Globus Nexus [10] has long provided identity management capabilities for Globus services. With the creation of Globus Auth it was necessary that we continue to support existing Globus identities. To do so, we registered a new identity provider in Globus Auth, called GlobusID. The transition to GlobusID was further complicated by the fact that Globus Nexus already supported an identity linking model. To retain these linkages we migrated identities from Globus Nexus to Globus Auth. We used a script-based approach to migrate existing identities, along with the identity provider that issued the linked identity, to Globus Auth. We then assigned each identity a unique Globus Auth identifier (i.e., UUIDs) and linked the identities together into a Globus account.

Globus Nexus has offered an OAuth 2 interface (called GOAuth) for several years. While this interface enables integration with third party applications, it neither provides the flexibility of Globus Auth nor supports integration with standard client libraries. Thus, we have deprecated this interface in favor of Globus Auth. To avoid the many applications and services that previously used GOAuth having to migrate to Globus Auth at the time of release, we developed a proxy model via which GOAuth requests can be mapped to Globus Auth requests. We created Globus Auth clients for all existing applications. Requests to the GOAuth authorization endpoint are redirected to Globus Auth, where a Python middleware layer translates the `scope` and `client_id parameters` into forms accepted by Globus Auth. Globus Auth handles the authorization request normally, prompting for user consent if needed and returning an authorization code. The legacy client POSTs the authorization code to the GOAuth token endpoint, which acts as a proxy, passing the code to Globus Auth for validation, and returning the access token response issued by Globus Auth in place of its native GOAuth token response.

F. Operations

The Globus Auth service implementation comprises two primary components: the Database and the Application. The Database component is a replicated PostgreSQL database hosted on RDS. The Application servers are a cluster of stateless web servers which use the database for all data persistence requirements. This lets us scale the main load-sensitive component—the web servers—trivially without any impact on service availability. Additionally, we autoscale web worker processes with load, allowing for resilience against failure and spikes in the rate of requests. In addition to the core infrastructure we operate a number of additional services to monitor the system, record and aggregate logs, and detect intrusions. These services enable our operations team to be alerted to service degradation immediately and to review the events leading up to an error or failure.

Deployments and software updates are handled largely by the Chef configuration management system, allowing us to safely assume that all application servers are identical and interchangeable. Chef allows us to develop automated and reproducible scripts for completely configuring distributed infrastructure including all dependencies, application software, and configurations. Using a reproducible and automated model enables us to maintain high availability and reduce errors,

allows versioned infrastructure configurations, supports reproducible deployments, enables deployment of production-like test and staging environments, and facilitates rapid deployment required for updates, infrastructure scaling, and failure. To eliminate downtime caused by updates, all application updates and deployments are conducted as rolling upgrades on the application servers. These are the targets of a continuous integration pipeline that additionally lets us expand or replace the set of application servers at any time.

V. Third-party integration

External services and applications can use Globus Auth IAM capabilities directly, via the Globus Auth API. In addition, the Globus Auth delegated authorization model allows integrated services to access any other Globus Auth-compliant service using delegated access tokens. We describe here how five services built upon Globus Auth.

A. Globus Services

Globus provides a suite of capabilities for managing research data. Globus Transfer [9] provides for reliable, high-performance, third-party, unattended file transfers between Globus-accessible storage servers. Globus Data Publication [13] supports self-service publication of research data with user-configured submission and curation workflows, metadata association, persistent identifier creation, and flexible search mechanisms. Globus groups [10] provides a user-oriented groups model with web and email-based workflows, role-based management, and configurable visibility and membership policies.

All Globus services use Globus Auth for identity management, authentication, and authorization. Each service has a registered Globus Auth client, implements standard OAuth 2 authentication workflows, and uses Globus Auth APIs to retrieve identity information. Each service is *linked identity aware*, meaning that it derives user state from the collection of linked identities. Thus, a Globus Transfer user can access endpoints shared with any of their linked identities; a Globus Data Publication user can perform the superset of roles assigned to their linked identities; and Globus group membership is derived from the user's linked identities.

As an example of the enhanced capabilities obtained from using Globus Auth, we have recently developed a secure Globus HTTPS endpoint server [14] that enables users to access Globus endpoint-accessible data via HTTPS. This server is an extension of the Globus Connect Server installed on a Globus endpoint, and acts as an OAuth2 resource server for itself and all shared endpoints that it hosts. A user's web client is a "client" to the HTTPS resource server. The HTTPS resource server is a client to Globus Auth and Globus Transfer resource servers. Globus Auth serves as the authorization server. When a user attempts to access a file on a Globus endpoint via HTTPS, the client is redirected to Globus Auth to authenticate using a supported identity provider. The Globus Auth API creates access tokens that match the approved consents (transfer and auth) for this user, and presents these tokens to the HTTPS resource server. The HTTPS resource server can then validate the token and use the delegated transfer access token to validate the user's access permissions on that endpoint.

B. Research Data Archive

The National Center for Atmospheric Research (NCAR) maintains the Web-based Research Data Archive (RDA), which contains more than 600 data collections. These collections, which range in size from gigabytes to tens of terabytes, include meteorological and oceanographic observations, operational and reanalysis model outputs, and remote sensing datasets to support atmospheric and geosciences research, along with ancillary datasets, such as topography/bathymetry, vegetation, and land use datasets. RDA users are primarily researchers at federal and academic research laboratories. In 2014 alone, more than 11,000 people downloaded more than 1.1 petabytes. Until recently, all downloads were over HTTP, either via Web browser, or via scripts that use wget or cURL.

In order to provide its users with an easy to use, reliable, high performance delivery service, NCAR recently added the ability to download data via Globus. Globus provides simple web interfaces for setting up and monitoring downloads, and implements the downloads themselves by specialized software and protocols that usually outperform HTTP and that can resume a download even if the system being downloaded to (or from) is temporarily turned off or temporarily loses its network connection. The Globus Transfer service thus ensures that downloads complete, regardless of how many times they are interrupted along the way.

When NCAR added Globus data services to RDA, they also integrated support for Globus identities and authentication. The original integration used Globus Nexus and thus required that each RDA user have a Globus username and password (i.e., a GlobusID identity). NCAR is now migrating RDA to use Globus Auth, which means that RDA users will no longer need to have a GlobusID identity. Instead, users can leverage their existing RDA identity either individually or linked with other identities in a Globus account. This enhancement significantly improves user experience and decreases the complexity of the RDA-Globus integration.

C. XSEDE

The Extreme Science and Engineering Discovery Environment (XSEDE) [15] is the national scientific cyberinfrastructure federation in the US. XSEDE supports 16 supercomputers and high-end visualization and data analysis resources across the US. The XSEDE ecosystem also includes administration and scientific services such as the XSEDE User Portal (XUP) and XSEDE Resource Allocation Service (XRAS). These services allow users to manage accounts, publications, and allocations associated with their use of XSEDE resources.

XSEDE is currently integrating Globus Auth into their user-facing services and APIs. XSEDE service APIs are being updated to support Globus Auth access tokens and web interfaces are being extended to operate with Globus Auth supported

identities. To translate XSEDE identities into a supported Globus Auth protocol we have developed an OpenID server that translates XSEDE identities obtained using Kerberos into an OpenID Connect interface. All XSEDE clients (e.g., XUP) use Globus Auth clients that are set with an "effective identity" policy that requires that all users have an XSEDE identity linked to their Globus account.

D. Jetstream

Jetstream [16] is an NSF-funded cloud resource designed to support general science and engineering research. Jetstream offers on-demand access to virtualized resources and services. Its implementation is based on OpenStack and uses the Atmosphere cloud computing environment to expose an interface to users. Atmosphere offers both web and REST interfaces that allow users to instantiate and manage virtual machines, including all of the complexities involved with storage, network, and security configurations. It provides a repository of community virtual machines which are prepopulated with standard software, this allows users to quickly stand up environments to conduct their research.

The Jetstream team have integrated Atmosphere with Globus Auth to provide seamless authentication and authorization experience. Atmosphere acts as a resource server enabling user access to its resources. To support a delegated access model in which other services may use Atmosphere functionality we have registered a new Globus Auth scope for accessing Atmosphere resources. As Jetstream is an XSEDE resource, Atmosphere also requires that each authenticated user have a linked XSEDE identity. The authentication flow redirects users to Globus Auth, which in turn allows the user to authenticate using a supported identity. The resulting access token is returned to Atmosphere and validated via the Globus Auth APIs. The user is then either granted or denied access based on inspection of the token and linked identities.

E. FaceBase

The FaceBase consortium generates data and develops tools to support research into craniofacial development and malformation. Ten *spoke* projects are tasked with generating data and tools, and a single *hub* is responsible for aggregating these data and tools and making them available to the craniofacial research community. FaceBase includes genetic, molecular, biological, imaging, and other data for zebrafish, mouse, human, and other organisms.

The FaceBase architecture is multi-faceted. Data is stored in a proprietary object store and metadata is stored in a relational entity management system [17]. A flexible data search and browsing interface is provided by a dynamic web application. Rather than manage identities and groups (for access control), FaceBase instead uses Globus Auth and Globus groups. The FaceBase team have developed a modular authentication and authorization plugin to their services. This plugin uses a standard OpenID Connect client library to implement the OpenID Connect workflow provided by Globus Auth. To abstract the complexities involved with using multiple services, FaceBase relies on a branded Globus web instance to enable identity and group management for their users. In this way, FaceBase users are able to authenticate using any of the Globus Auth supported identity providers, manage their Globus account (e.g., linking various identities) and manage groups.

VI. Related Work

Social and commercial applications have long foregone built-in identity management for the use of external identity management solutions. For example, many websites and mobile applications use social network identities, such as Google and Facebook, for authentication. In each case, the third-party application or service implements a standard OAuth 2 client to the Google or Facebook authorization server. However, unlike Globus Auth, these systems do not provide identity brokering capabilities and instead support only a single identity provider.

InCommon is a framework that provides trusted access to online resources and identity management federation across US academic institutions. Global Authentication INfrastructure for education (eduGAIN) [18] extends this notion through a global confederation that connects regional federations, including InCommon. Both InCommon and eduGAIN are built upon SAML and allow service providers to enable user authentication using federated identities. Neither is capable of supporting various IDP protocols, linking identities, or providing scoped or delegated tokens. The Agave [19] and Apache Airavata [20] platforms provide user management, authentication and authorization, job submission, and data management capabilities via REST APIs. Agave supports OAuth 2-based authentication workflows that allow third-party applications to leverage its capabilities. However, unlike Globus Auth, it provides no brokering, account linking, or delegation capabilities. Apache Airavata does not support identity and group management.

Several commercial entities provide identity brokering services. For example, the Google identity platform supports identities from SAML and OpenID Connect identity providers, while also offering OAuth 2 and OpenID Connect interfaces for application integration. Atlassian Crowd [21] is a service-based identity management service for web applications. It enables user identities to be sourced from external directories and exposes different authentication interfaces, such as OpenID, that can be embedded in external applications.

Amazon Web Services Identity and Access Management (IAM) [22] provides identity federation support enabling users to authenticate using their local identity provider. This integration is built on SAML. IAM also provides Web identity federation, a mechanism that allows developers to use different identity providers and to trade an authentication token from these providers for temporary AWS credentials. It supports Amazon, Facebook, and Google identity providers. The related Amazon Cognito [23] associates a unique identifier with each identity that can be referenced across devices and applications. It supports the creation of temporary, limited-privilege credentials such that a third-party application can access AWS resources. Its primary purpose is simplified synchronization of

application state across devices and thus it lacks capabilities such as identity linking and delegated access tokens.

Another commercial service, Auth0 [24], addresses the challenge of mapping from many identities to many applications. It supports a large number of identity providers, including Facebook, Google, LinkedIn, Github, and Amazon. Like Globus Auth, it provides APIs for accessing and managing profiles, and allows users to link identities, use linked identities in federated login scenarios, restrict the range of identity providers that can be used to authenticate with a given client, and use OAuth 2 and OpenID Connect interfaces for integration into third-party applications.

Despite these similarities, Auth0's focus on identity mapping leads to important differences. For example, Auth0 provides little support for enforcing limited access to managed resources. At present, it only provides limited control over who can access the profile information that it manages, offering just three pre-defined scopes for profile information access. In contrast, Globus Auth implements a flexible and extensible consent-based scope model in which many different scopes can be defined for each resource server. Similarly, the Auth0 delegation model is limited: clients must be preconfigured with a group of add-on (or external) services for which delegated tokens can then be obtained. In contrast, Globus Auth provides a rich delegation model in which tokens with different scopes can be obtained by a given client to access other services on behalf of the user. Globus Auth is also differentiated by its support for primarily research identity providers.

VII. Summary

Globus Auth provides a flexible identity and access management platform for the research community. Its unique characteristics, including identity brokering, identity linking, and delegated access model, directly address many frictions associated with creating and operating research services. Globus Auth already supports a range of research identity providers and can be integrated with external research services via standard OAuth 2 interfaces. Globus Auth can thus form the basis for a new generation research platform on which researchers can rapidly develop new services that leverage other research services. Moreover, Globus Auth provides a fabric for integrating the currently fragmented and siloed ecosystem of research services. In the three months since deployment Globus Auth has been adopted by a number of large research projects. This encouraging uptake highlights the potentially transformative effect that Globus Auth can have on the research service landscape.

Acknowledgments

We thank Globus subscribers for supporting the operation and development of Globus. We also thank users of Globus services for their continued support. This research was supported in part by NSF grant ACI-1053575 (XSEDE) and US Department of Energy contract DE-AC02-06CH11357.

References

[1] V. Welch, A. Walsh, W. Barnett *et al.*, "A roadmap for using NSF cyberinfrastructure with InCommon," in *TeraGrid Conference: Extreme Digital Discovery (TG)*, 2011, pp. 28:1–28:2. [Online]. Available: http://doi.acm.org/10.1145/2016741.2016771

[2] R. Ananthakrishnan, K. Chard, I. Foster *et al.*, "Globus platform-as-a-service for collaborative science applications," *Concurrency - Practice and Experience*, vol. 27, pp. 290–305, 2014.

[3] D. Hardt, *The OAuth 2.0 Authorization Framework*, http://www.rfc-editor.org/info/rfc6749 [accessed May 2016], RFC 6749 Std., October 2012.

[4] N. Sakimura, J. Bradley, M. Jones *et al.*, "OpenID connect core 1.0," http://http://openid.net/specs/openid-connect-core-1_0.html [accessed May 2016], OpenID Foundation, 2014.

[5] S. Cantor, J. Kemp, R. Philpott *et al.*, "Security assertion markup language (SAML) v2.0," https://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf [accessed May 2016], OASIS, 2014.

[6] I. Foster, "Service-oriented science," *Science*, vol. 308, no. 5723, pp. 814–817, 2005. [Online]. Available: http://science.sciencemag.org/content/308/5723/814

[7] I. Foster, K. Chard, and S. Tuecke, "The discovery cloud: Accelerating and democratizing research on a global scale," in *IEEE International Conference on Cloud Engineering (IC2E)*, April 2016, pp. 68–77.

[8] I. Foster, "Globus Online: Accelerating and democratizing science through cloud-based services," *IEEE Internet Computing*, vol. 15, no. 3, pp. 70–73, May 2011.

[9] B. Allen, J. Bresnahan, L. Childers *et al.*, "Software as a service for data scientists," *Communications of the ACM*, vol. 55, no. 2, pp. 81–88, Feb. 2012. [Online]. Available: http://doi.acm.org/10.1145/2076450.2076468

[10] K. Chard, M. Lidman, B. McCollam *et al.*, "Globus Nexus: A platform-as-a-service provider of research identity, profile, and group management," *Future Generation Computer Systems*, vol. 56, pp. 571–583, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X1500285X

[11] J. Richer, "OAuth 2.0 token introspection," https://tools.ietf.org/html/rfc7662 [accessed May 2016], Internet Engineering Task Force (IETF), 2014.

[12] K. Chard, S. Tuecke, and I. Foster, "Efficient and secure transfer, synchronization, and sharing of big data," *IEEE Cloud Computing*, vol. 1, no. 3, pp. 46–55, Sept 2014.

[13] K. Chard, J. Pruyne, B. Blaiszik *et al.*, "Globus data publication as a service: Lowering barriers to reproducible science," in *11th IEEE International Conference on e-Science*, Aug 2015, pp. 401–410.

[14] K. Chard, S. Tuecke, I. Foster *et al.*, "Globus: Recent enhancements and future plans," in *the Annual Annual Conference on Extreme Science and Engineering Discovery Environment (XSEDE)*, 2016.

[15] J. Towns, T. Cockerill, M. Dahan *et al.*, "XSEDE: Accelerating scientific discovery," *Computing in Science Engineering*, vol. 16, no. 5, pp. 62–74, Sept 2014.

[16] C. A. Stewart, T. M. Cockerill, I. Foster *et al.*, "Jetstream: A self-provisioned, scalable science and engineering cloud environment," in *XSEDE Conference: Scientific Advancements Enabled by Enhanced Cyberinfrastructure*, ser. XSEDE '15, 2015, pp. 29:1–29:8. [Online]. Available: http://doi.acm.org/10.1145/2792745.2792774

[17] R. Schuler, C. Kesselman, and K. Czajkowski, "Digital asset management for heterogeneous biomedical data in an era of data-intensive science," in *IEEE International Conference on Bioinformatics and Biomedicine*, Nov 2014, pp. 588–592.

[18] "eduGAIN," http://services.geant.net/edugain/ [accessed May 2016].

[19] R. Dooley, M. Vaughn, D. Stanzione *et al.*, "Software-as-a-service: The iPlant Foundation API," in *5th IEEE Workshop on Many-Task Computing on Grids and Supercomputers (MTAGS)*, 2012.

[20] M. Pierce, S. Marru, L. Gunathilake *et al.*, "Apache Airavata: Design and directions of a science gateway framework," in *6th International Workshop on Science Gateways*, June 2014, pp. 48–54.

[21] "Atlassian Crowd," http://atlassian.com/softwar/crowd/overview [accessed May 2016].

[22] "Amazon identity and access management (IAM)," http://aws.amazon.com/iam [accessed May 2016].

[23] "Amazon Cognito," https://aws.amazon.com/cognito/ [accessed May 2016].

[24] "Auth0," http://auth0.com [accessed May 2016].