XSEDE Web SSO for Application Developers

Version 1.3

The Web SSO service

The XSEDE Web SSO service offers XSEDE users **a uniform and consistent process to sign on to applications**. It is intended to be used by the XSEDE User Portal, science gateways, training websites, the Community Software Repository, XSEDE staff applications, and any application that is part of the XSEDE user experience.

When a user wishes to sign on to an application, the following happens.

1. The application directs the user to the Web SSO service.
2. The Web SSO service allows the user to securely authenticate using an identity provider of the user's choice (this may or may not be XSEDE).
3. The Web SSO service returns an XSEDE identity to the calling application.
4. The application can then use the XSEDE identity to provide a personalized user experience, including access control (authorization) decisions.
5. Meanwhile, the Web SSO service maintains a sign-on session for the user that is used when the user signs on to other Web SSO applications, until the user explicitly signs off.

Beyond the sequence above, the Web SSO service provides the following significant features.

- Users are able to use their XSEDE username and password to sign on with applications.
- Users are able to use their identities from InCommon and eduGAIN member institutions and other academic/research organizations to sign on with applications.
- Applications always receive the user's XSEDE identity regardless of the identity used for authentication. Users who haven't already registered with XSEDE or linked an XSEDE identity are directed to do so as they sign on.
- If a user has recently signed on to a Web SSO-enabled application and hasn't signed off, the user may sign on with other Web SSO applications without re-authenticating.
- Users can link their own identities from multiple institutions, enabling applications to make authorization decisions based on a user's full set of identities.
- Users' private credentials (passwords, one-time tokens, etc.) are never exposed to the Web SSO service or to applications that use the Web SSO service.

Figure 1 shows two different people using an application that uses the XSEDE Web SSO service. One person uses the University of Washington to authenticate, while the other uses ORCID. Because both users previously linked their XSEDE identities in the Web SSO service, the application receives their XSEDE identities. (The application can access data from the user's other identity providers with the user's permission.)

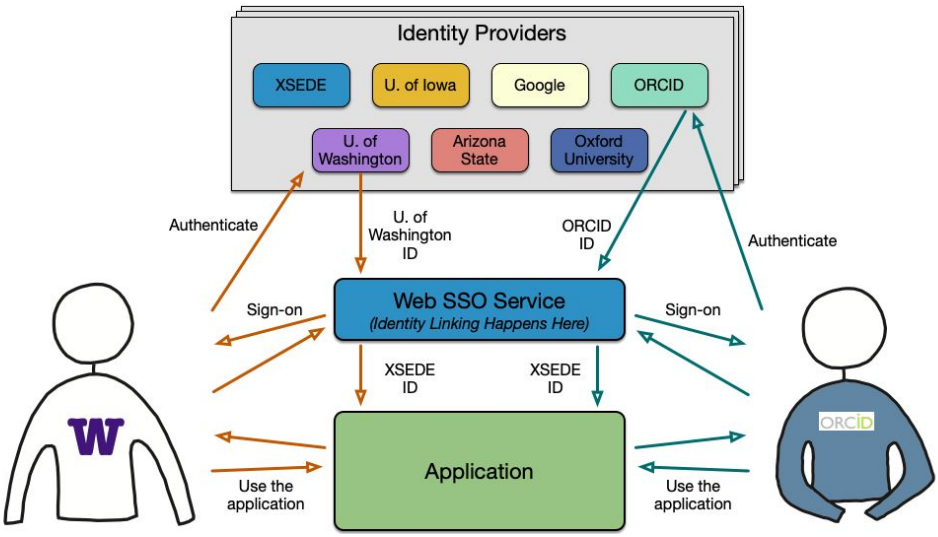

Figure 1. Overview of XSEDE's Web SSO service showing two users and one application.

The XSEDE Web SSO service relies heavily on an existing Web SSO system, Globus Auth. Globus Auth is a service of the University of Chicago. Because XSEDE has selected Globus Auth to be its Web SSO service provider, applications that use the XSEDE Web SSO service, in effect, use Globus Auth. However, XSEDE adds several elements and procedures that customize the Globus Auth experience for application developers and for users.

Globus is responsible for managing the user experience for XSEDE's Web SSO service for both application developers and application users. As described in the section, "How to get help," Globus participates closely in XSEDE's operations and user support activities and provides user and developer support for the Web SSO service.

The "Login with XSEDE" interface element

If your application uses the XSEDE Web SSO service, XSEDE strongly recommends that you use the "Login with XSEDE" or the "Login to XSEDE" button shown in Figure 2 as the interface to initiate sign-on. (Downloadable versions are available at https://software.xsede.org/development/xsede-web-sso/ .) The button uses the XSEDE 'X' icon and the XSEDE white-on-navy color scheme. You may scale the button to fit your application's interface.

Figure 2. The "Login with XSEDE" and the "Login to XSEDE" buttons

The “Login to XSEDE” button is best for applications provided by XSEDE, such as the XSEDE User Portal (XUP). These applications should be on hosts whose names end in “.xsede.org.”

The “Login with XSEDE” button is best for applications that aren’t provided directly by XSEDE. Any application that uses the XSEDE Web SSO service as described in this document may use this button. It is specifically meant to be used by Service Provider (SP) web interfaces, science gateways, and campus services that use XSEDE.

Using a “Login to/with XSEDE” button is important for two reasons.

1. It offers a uniform user experience across applications that use the service.
2. When users see the same design element in multiple applications, it signals that these applications are using a common sign-on mechanism, so signing on (or off) with one application may affect the sign-on status in other applications that display the same element.

The second point is important: when your users sign on to your application (or sign off!), it will affect the behavior of other applications that use the Web SSO service. Giving the user a hint via a common user interface element that a common service is being used will help avoid confusion.

The two examples below show how this button can be used in the XSEDE user portal and other XSEDE applications.

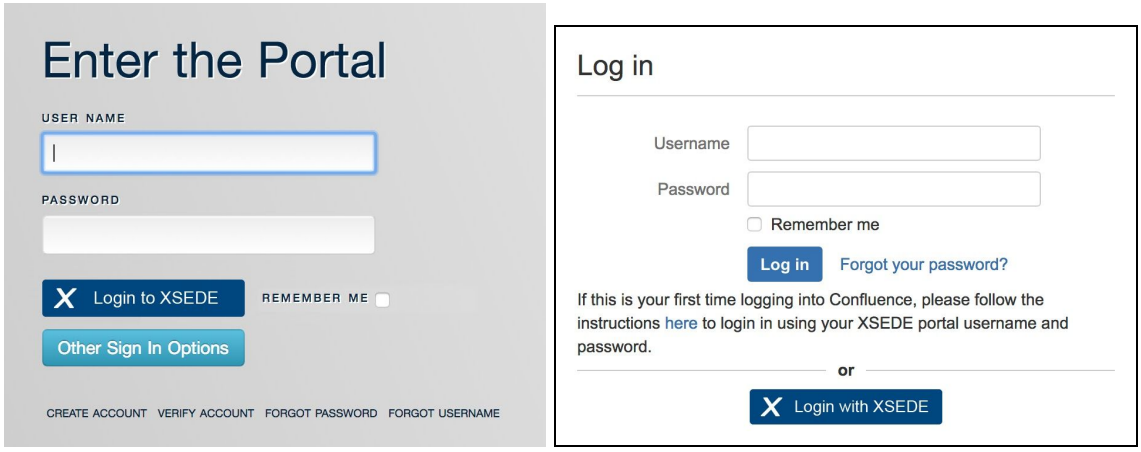

Figure 3. Two examples of the “Login to/with XSEDE” button

If possible, your application should also indicate the user’s signed-on status by displaying the username, especially if your application uses the XSEDE username as described below in “Elements of an XSEDE identity.” This will also help users understand that their sign on session is related to XSEDE.

How it works

To adapt an application to use the Web SSO service, you'll follow these high-level steps.

1. Choose an OIDC plugin, adapter, module, or SDK.
2. Register your application.
3. Configure your OIDC plugin, adapter, module, or SDK to use the XSEDE Web SSO service.
4. If you're using an SDK, write code to obtain and use the user's XSEDE identity.
5. If you're using a plugin, adapter, or module, configure it to use XSEDE identities.

Choosing an OIDC plugin, adapter, module, or SDK

If you're developing a Web application in a development framework like Django or Flask, or if you're deploying a Web application like WordPress, Drupal, or Liferay, there's most likely a plugin or module that you can install to use the XSEDE Web SSO service. Appendix B offers notes on some of these environments. If you can't find it in the product documentation or appendix, try a Google search for "*productname* OIDC plugin". Then follow the instructions in the rest of this section to register your application and configure your plugin.

If you're writing your application from scratch, you should look for an OIDC SDK for the programming language you're using. Try a Google search for "*language* OIDC SDK" or "*language* OIDC client". Follow the instructions below to register your application, configure the SDK to use the Web SSO service, and write the code necessary to call the SDK and use the XSEDE identity data it returns.

Registering your application

Like all OIDC services, XSEDE's Web SSO service requires applications to be registered. You'll be registering your application with Globus. To register a new application, visit developers.globus.org, click "Register your app with Globus," and sign in. Then, ***read the special XSEDE Web SSO instructions below*** and then follow the instructions from the Globus Auth Developer's Guide to register your application.

Special XSEDE Web SSO registration instructions:

1. The first step in registering a client application at the Globus Developers site is to create a project. Each project can have multiple administrators. After you create a new project or select an existing one, click the "Add..." button in your project and select "Add/remove admins." Add **xsede@globusid.org** as an administrator of your project. There are two reasons to do this. First, it enables us to track your application's use of the Web SSO service. (NSF asks us to do this. See Appendix A for details.) Second, it enables us to help you if you lose track of your client registration or if you change jobs and someone else needs to manage the registration. (See Continuity service for application registrations below.)

2. When registering your application with Globus, *you must check the **Required Identity** box and select "XSEDE."* This instructs the Web SSO service to always return the user's XSEDE identity, regardless of which organization the user authenticates with. Users who haven't already registered with XSEDE or linked an XSEDE identity are directed to do so as they sign on. If you don't enable this function when you register your application, the service won't behave as described in this documentation.

3. Check the **Pre-select Identity Provider** box and select "XSEDE." This instructs the Web SSO service to make XSEDE the default organization on your app's login page.

4. In the **Terms and Conditions URL** box, enter the URL for the XSEDE Acceptable Use Policy: https://www.xsede.org/ecosystem/operations/usagepolicy . Appendix A explains why this is important.

Configuring OIDC plugins, modules, and SDKs

The Web SSO service provides a standard OIDC interface, so it will work with existing OIDC plugins, modules, and SDKs. You'll need to configure your OIDC interface to use XSEDE's service, however. The settings in Table 1 will allow an OIDC interface to use the XSEDE Web SSO service.

OIDC setting	**Web SSO service**
Authorization endpoint	`https://auth.globus.org/v2/oauth2/authorize`
Token endpoint	`https://auth.globus.org/v2/oauth2/token`
UserInfo endpoint	`https://auth.globus.org/v2/oauth2/userinfo`
OIDC metadata URL	`https://auth.globus.org/.well-known/openid-configuration`
OIDC scopes	`"openid email profile"`
Client ID	Obtained by registering your application (see above)
Client Secret	Obtained by registering your application (see above)

Table 1. OpenID Connect (OIDC) configuration settings for the XSEDE Web SSO service

The OpenID Connect (OIDC) model

XSEDE's Web SSO service provides an OpenID Connect (OIDC) authentication interface. This interface provides three different ways for applications to authenticate users, two of which are documented here. (The third method is not permitted for most applications.) Both methods result in the application gaining access to identity data about the user from XSEDE, and possibly other identity providers as well.

- **Confidential application** - By far the most common method, this is intended for web portals (hosted web applications). The application runs on a web server and the user interacts with it via a browser.
- **Native application** - Less commonly, OIDC can also be used in applications that users download and run on their own systems or access via a command line. These applications lack a browser interface, but the user will use a browser when signing on to the application. (The browser doesn't have to run on the same system as the application.)

When the user wishes to sign on, the following interactions take place. *If you're using a third-party OIDC plugin or module, all of these interactions are handled by the plugin or module and your application only sees the user identity that's returned.*

1. The application directs the user's web browser to the Web SSO service. (Native applications prompt the user to open a browser to a specific link.)
2. The Web SSO service guides the user to authenticate using an identity provider of the user's choice (this may or may not be XSEDE).
3. If the user hasn't already linked an XSEDE identity to the identity the user authenticates with, the Web SSO service guides the user to do so, including registering with XSEDE if necessary.
4. The Web SSO service requests the user's permission to release identity data to the application.
5. The Web SSO service then returns one or more tokens to the application that can be used to access the user's identity data. (For native applications, the Web SSO service gives the user a code to pass on to the application, and the application uses the code to obtain the tokens by which it gains access to the user's identity data.)

If you're using an SDK, you'll add code to your application to call the SDK for steps 1 and 5 above and to extract the resulting identity data. (Steps 2-4 are interactions between the user and the Web SSO service.) Details on the necessary code are provided with your SDK. If you're using Globus's SDK, see "Globus Auth documentation and resources" below.

Globus Auth documentation and resources

If you're using a third-party OIDC plugin or module to access the Web SSO service, you're unlikely to need the following resources. If you're writing your own code using an SDK (including the Globus Python SDK), the following references provide details on how to code applications to use the Web SSO service.

Globus Auth provides XSEDE's Web SSO service. The *Globus Auth Developer's Guide* and *Globus Auth API Reference* offer general and detailed guidance, respectively, on how to develop applications using Globus Auth. When using these references, bear in mind that you're following the special XSEDE Web SSO instructions described in the previous section, and this affects the service's behavior.

The Globus Python SDK provides a high-level Python interface for the Web SSO service. Using the Python SDK will simplify your application code and make the interaction with the Web SSO service easier to understand and debug. The SDK Tutorial provides several simple examples that use the SDK for

authentication, though it's important to notice that the examples use the native application method (for locally installed applications), as opposed to the confidential application method used by web portals.

The XSEDE Globus ID Explorer application (see description below) provides a helpful way to see and learn about the identity data returned by the Web SSO service. The GitHub repository for this application provides simple, but good, example code for a web portal using the Web SSO service.

Globus's *Modern Research Data Portal* is a design pattern used in application developer tutorials that illustrates how to develop a portal using Globus Auth and related services. It includes code, a working portal that you can install on your own server, and a narrative code walk-through explaining how the code works.

The Globus Jupyter Notebooks GitHub repository provides interactive Python examples that use Globus Auth. The examples are meant to be run in Jupyter notebooks, an interactive, web browser-based Python environment that you can install on your own system.

Using XSEDE identities in your application

The Web SSO service returns identity data as a JSON object containing name/value pairs. These pairs are called *claims*. The specific claims included in the response are determined by the Web SSO service, the XSEDE IdP, and the consents granted by the application user. (Users must provide consent for their identity data to be released to an application.) Figure 4 provides an example of this response.

In Figure 4, the claims that contain XSEDE IdP data are: name, email, organization, and preferred_username. The name, email, and organization values come from the user's XSEDE user profile. The value of the preferred_username claim is the user's XSEDE username with the suffix "@xsede.org".


```
{
   'identity_provider_display_name': 'XSEDE',
   'sub': 'b3804ef4-d274-11e5-94fb-1f2e32b95784',
   'preferred_username': 'ysvenkat@xsede.org',
   'identity_provider': '36007761-2cf2-4e74-a068-7473afc1d054',
   'organization': 'University of Illinois at Urbana-Champaign',
   'email': 'vyekkira@illinois.edu',
```

```
   'name': 'Venkatesh Yekkirala'
}
```

Figure 4. Identity claims returned by the Web SSO service

If you're using a third-party OIDC plugin, module, or SDK, it might decode the JSON object for you, returning only some of the claims. When you configure the plugin, you'll probably need to choose which of these claims will be used for the username in your application. The following claims will most likely be offered as choices: email, sub, preferred_username, and name.

Whether you're using a third-party plugin or your own code, **the preferred_username value is the best choice for uniquely identifying users in your application.** XSEDE, by policy, does not change or recycle usernames, so a given username will always refer to the same person.

When using the preferred_username claim, you may strip off the "@xsede.org" part or leave it. If you strip it off, you should take care that it actually contains "@xsede.org" and not something else. There are rare error conditions where a different IDP appears in the preferred_username value, and it will be harder to detect and resolve those conditions if you strip off the IdP domain without checking that it is, indeed, "@xsede.org".

We strongly recommend against using the email or name fields to uniquely identify users in your application. Users often change these values in their XSEDE profile and they aren't guaranteed to be unique to a single person. This will confuse your application's user database and result in a poor user experience. You can use the email, name, and organization fields as extra information about each user, as long as they are associated with a unique username.

Like the preferred_username, the sub (short for *subject*) claim is guaranteed to be unique for each user. You might have noticed, however, that the sub claim isn't listed above as coming from XSEDE. It is generated by Globus the first time the user uses Globus to authenticate to XSEDE. Other than its use in the Web SSO service, this value is unknown to XSEDE. If, at any time in the future, XSEDE changes its Web SSO service from Globus to something else, this value would become meaningless and you would need to reassign all of your users to a new identifier. It isn't likely that XSEDE or Globus will provide a migration plan for applications that use the sub claim to identify XSEDE users, so we don't recommend that approach.

Affiliated institutions

There are two ways to access the user's identity data using Globus Auth. The first, which is described in the previous section, is to look at the **ID Token** returned on login. The second is to request and use the linked identities provided by Globus Auth. These two methods are described in the following sections.

Data available in the ID Token

The ID Token is described in the previous section. It is a data structure returned on login, and it contains several *claims* about the user's identity. For applications that follow the configuration steps described in the previous section, the claims in the ID Token will always be provided by XSEDE, because such

applications are configured to specifically request the user's XSEDE identity. The XSEDE ID Token includes an `organization` claim, specifying the organization the user reported when registering with XSEDE, which is then managed as part of the user's XSEDE user profile. It is intended to be the user's "home" institution: either their academic institution or their employer.

Figure 1 shows an example of an ID Token. In this example, the user's `name` is "Lee Liming," and Lee's `organization` is "University of Chicago." This organization comes from Lee's XSEDE user profile and Lee can manage its value by editing his profile in the XSEDE User Portal. We know that this data comes from XSEDE because the `identity_provider_display_name` is "XSEDE" and the `identity_provider` has a UUID corresponding to XSEDE.

```
{
   "email": "lliming@uchicago.edu",
   "identity_provider": "36007761-2cf2-4e74-a068-7473afc1d054",
   "identity_provider_display_name": "XSEDE",
   "name": "Lee Liming",
   "organization": "University of Chicago",
   "preferred_username": "liming@xsede.org",
   "sub": "b16bb6bc-d274-11e5-8e43-33a31885627e"
}
```

Figure 1. The identity claims in an ID Token returned by Globus Auth

Data available in linked identities

With a bit of extra code, applications can also request the user's *linked identities*. Linked identities may provide additional institutional affiliations. Beyond their basic XSEDE identity, an individual may *link* identities from other institutions, such as their academic institution or other research systems. The identities that appear depend entirely on the user's decision whether to link other identities or not, and if so, which specific identities to link.

In order to access linked identities, the application must request access to linked identities, and the user must grant this access to the application. This usually happens the first time each individual logs into the application.

Linked identities data is accessed via the OpenID Connect `UserInfo` API call. (Details are provided in the following section.) Figure 2 provides an example of the `UserInfo` results when linked identities are requested. The linked identity data is provided in the `identity_set` claim. In this example, the user has three linked identities: one from the University of Chicago, one from the ORCID system, and one from XSEDE. (The identity provider is named in the `identity_provider_display_name` and `identity_provider` claims in each linked identity.) The University of Chicago and XSEDE both provide an `organization` claim, but ORCID does not. From this example, we can see that the user has registered with the University of Chicago, ORCID, and XSEDE systems, and the user is most likely "based" at the University of Chicago, because: (a) in general, universities only maintain accounts for students, faculty, staff, and alums, (b) University of Chicago is the organization listed for the two identities that provide an organization, and (c) that is where the individual receives email.

An application could reasonably interpret these results as follows. The individual is based at University of Chicago, and also has affiliations with the ORCID system and the XSEDE system. Since ORCID exists largely to provide public profiles about researchers and their activities, the ORCID ID provided by ORCID (`0000-0002-4930-6145`) could be used to obtain further public information about the individual's research activities.


```
{
   "organization": "University of Chicago",
   "name": "Lee Liming",
   "identity_set": [
      {
         "organization": "University of Chicago",
         "name": "Ronald Liming",
         "sub": "b16b12b6-d274-11e5-8e41-5fea585a1aa2",
         "username": "lliming@uchicago.edu",
         "identity_provider": "0dcf5063-bffd-40f7-b403-24f97e32fa47",
         "identity_provider_display_name": "University of Chicago",
         "email": "lliming@uchicago.edu"
      },
      {
         "name": "Lee Liming",
         "sub": "7de9194c-d464-4ea7-aaff-0f3814c1e713",
         "username": "0000-0002-4930-6145@orcid.org",
         "identity_provider": "0519206d-f21c-4771-990a-282a12bb666b",
         "identity_provider_display_name": "ORCID",
         "email": null
      },
      {
         "organization": "University of Chicago",
         "name": "Lee Liming",
         "sub": "b16bb6bc-d274-11e5-8e43-33a31885627e",
         "username": "liming@xsede.org",
         "identity_provider": "36007761-2cf2-4e74-a068-7473afc1d054",
         "identity_provider_display_name": "XSEDE",
         "email": "lliming@uchicago.edu"
      }
   ],
   "sub": "b16bb6bc-d274-11e5-8e43-33a31885627e",
   "identity_provider": "36007761-2cf2-4e74-a068-7473afc1d054",
   "identity_provider_display_name": "XSEDE",
   "preferred_username": "liming@xsede.org",
   "email": "lliming@uchicago.edu"
```

```
}
```

Figure 2. UserInfo results from Globus Auth when linked identities are requested

How to access affiliated institution data

The previous section describes two ways to access affiliated institutions when logging users in via the XSEDE Web SSO service: via the ID Token and via linked identities. (Again, XSEDE Web SSO uses Globus Auth, so the rest of this section refers to Globus Auth.) This section explains how to access both kinds of data in your application.

ID Token

The ID Token is easy because it requires nothing more than the normal OpenID Connect (OIDC) login flow. The ID Token is returned from the login flow and can be accessed by your application directly. This is described in the previous "Using XSEDE identities in your application" subsection.

Linked identities

Accessing linked identities is slightly more challenging, but it still uses standard OpenID Connect interfaces. There are two steps to accessing linked identities. First, your application must request access to the user's linked identities. Second, your application must use the OIDC UserInfo API to access the linked identity data.

To request permission to view linked identities, add the following scope to the list of scopes your application requests when logging users in. (This is in addition to the usual OIDC scopes, "openid email profile".)

```
urn:globus:auth:scope:auth.globus.org:view_identity_set
```

Just add this to the three normal scopes in your application configuration or code. When users login to your application, Globus will ask their permission to allow your application to see their linked identities. The user must give this permission in order for your application to access his/her linked identities. The user will only be asked to provide permission once.

To access the linked identity data, your application must call Globus Auth's UserInfo API call. This is the standard OIDC UserInfo interface. The only difference is the data that is included in the results. When the linked identities scope (above) is requested, the UserInfo results will include the `identity_set` claim as described in the previous section.

The UserInfo call is a standard OpenID Connect API call. Here is Globus's documentation for this call.

How to get help

Technical support

The Web SSO service is supported by Globus team members who participate in the XSEDE Help Desk (support@xsede.org). Application developers and application users who need help with the Web SSO service should send email to support@xsede.org. Help Desk personnel should assign Web SSO tickets to the Globus members of the Help Desk team.

Notify XSEDE support staff about new applications

When you get your application working with the Web SSO service, please send a brief email message to the XSEDE Help Desk (help@xsede.org) describing how your application uses the service. It will speed our response to any support questions that come up later. Please tell us the Client ID from your application registration, the OIDC plugin/adapter/module or SDK your application uses, and the specific identity claims used by the application. Please also confirm that you set the Required Identity setting to "XSEDE" as described in "Registering your application." If you mention that this information is for use by the XSEDE XCI team, it will help us record the information in the right place.

Continuity service for application registrations

Applications that use the Web SSO service must be registered. This registration is part of the ongoing maintenance for the application. The registration interface allows multiple administrators of registrations, so if the original developer leaves the application in someone else's care, the new person can continue using the application's original registration.

XSEDE's XCI team provides a Globus identity, xsede@globusid.org, that can be added as an administrator for application registrations. Adding this identity as a project administrator for your application registrations ensures that XCI staff will be able to recover your application's registration if/when you change jobs without assigning a new project manager.

XSEDE Globus ID Explorer

XSEDE provides the XSEDE Globus ID Explorer application, which allows application developers and application users to view and manage their identity data and application permissions in the Web SSO service. Application developers can see the JSON data structures and identity claims returned by specific interfaces. The source code for this application provides good example code, including examples of accessing linked identities, accessing authentication events, and use of helper pages.

Preview environment

Globus provides a preview environment in which the next version of the Auth service is available for application testing. Application developers can register and test their applications in the preview environment the same as they do in the primary environment. The Globus Preview environment and instructions for its use are described at https://docs.globus.org/how-to/preview/.

Globus application developer email list

Globus provides an email list for application developers: developer-discuss@globus.org. Significant changes to the Globus Auth service are announced ahead of time on this list. The list is also available for Q&A with the Globus team. The list's archive and instructions for joining the list are available at https://www.globus.org/mailing-lists.

Please give us feedback

We would like feedback! Please go to the URL below and let us know your experiences using these instructions and/or getting help from our team.

https://software.xsede.org/node/3099/vote

Appendix A - Usage data Q & A

The National Science Foundation (NSF) asks XSEDE to collect usage data for *all* XSEDE services. The XSEDE Web SSO Service–provided by Globus Auth–is included in this request. Consequently, XSEDE collects usage data for application logins that use the Web SSO Service and Globus Auth.

Q: Why is XSEDE collecting usage data for Globus Auth?
A: XSEDE has been asked by NSF to collect usage data for *all XSEDE services* in order to better understand the ROI (return on investment) for each service. In a nutshell, NSF and its peer review panels want to be assured that each XSEDE service is worth the funds and effort spent on it. Globus Auth is one of these services, so NSF is interested in its use.

Q: What data is collected?
A: For Globus Auth, XSEDE obtains usage records from Globus. Each record contains a timestamp, an identifier for the application being logged into, and the XSEDE username that logged into the application. No other details are included.

Q: Isn't this private data? It includes Personally Identifiable Information (PII).
A: Yes, the data contains XSEDE usernames, which are PII. The XSEDE Acceptable Use Policy (AUP)–which all XSEDE users must acknowledge at least once a year–states that user identities and usage data will be shared among XSEDE partners for the purpose of operating the system. XSEDE and its predecessors have been collecting this data for decades from allocated XSEDE resources (computation & data services).

XSEDE manages this data very carefully to avoid disclosure. Now, at NSF's request, we're extending usage data collection to cover other parts of the XSEDE system.

Q: Why are logins to my application included in XSEDE usage data?
A: If your application uses the XSEDE Web SSO service, it not only uses the Web SSO service, it also uses XSEDE user identities. Both XSEDE user identities (which are managed and provided by XSEDE) and Globus Auth (the login mechanism) are services provided by XSEDE, thus NSF is interested in their use.

Q: Am I responsible for this data sharing?
A: Technically, no. To be precise, Globus is sharing its usage data with XSEDE. Your application uses Globus for logins, and Globus is sharing that use with XSEDE. However, because this happens in the context of your application, we want to be certain you're aware of this, are ok with it, and are informing your users (by linking the XSEDE AUP in your Globus client registration) that usage data is shared with other XSEDE partners. If you believe this complicates your responsibilities to your users, please share those concerns with us.

Q: How does linking the XSEDE Acceptable Use Policy in my Globus client registration help?
A: When each user logs into your application for the first time, Globus displays a page asking the user to allow Globus to share the user's identity information with your application. If your client registration includes a terms & conditions link, Globus will display it for the user on this page. The same link is also displayed on Globus's Consents page, which lists applications the user has allowed to use their identity data. This allows users to see the terms & conditions of any apps they use with Globus.

Q: What if my application has its own Terms & Conditions statement?
A: Use it instead! Globus client registrations allow each application to provide a privacy policy and a terms & conditions statement. If your application has a privacy policy or terms & conditions, please add it to your client registration.

Appendix B - Notes for specific application environments

The following notes offer starting points for integrating the XSEDE Web SSO service (based on Globus Auth) with popular application development environments. These are not substitutes for documentation, but provide links to documentation.

Python Social Auth (Django, Flask, Pyramid)

Python Social Auth provides an OIDC/OAuth user login and registration module for Django, Flask, and Pyramid applications. It includes a Globus backend, so all you need to do is register your application with Globus and configure your application to use the Globus backend with your Client ID and Client Secret. If you're using Django, a step-by-step example will get you started in about ten minutes.

- Python Social Auth documentation
- Python Social Auth configuration details for Globus
- Register your application with Globus
- Step-by-step example for Django

JupyterHub

JupyterHub is a popular education and learning tool, enabling students to launch active notebooks in their browsers. If you host a JupyterHub, you can configure it to use Globus for user logins. Even better: when students launch notebooks using your JupyterHub, their notebooks will have an OAuth access token for the Globus transfer service based on their XSEDE identity, so they can execute code that transfers files to/from XSEDE systems within the notebook. The ability is provided by the JupyterHub OAuthenticator, so all you need to do is register your JupyterHub with Globus and configure it to use Globus and your Client ID and Client Secret.

- Sign in with Globus in JupyterHub
- JupyterHub OAuthenticator
- Register your JupyterHub with Globus
- Configure JupyterHub OAuthenticator for Globus

Wordpress

Wordpress websites and blogs can use Globus for user logins and registration. A free plugin from the Wordpress Marketplace enables the functionality. You must register your Wordpress site with Globus and configure the plugin with your Client ID and Client Secret. A step-by-step example will get you running in about ten minutes.

- Step-by-step example for Wordpress sites
- Register your application with Globus
- Free plugin for Wordpress

Atlassian Confluence and Jira

You can configure a Confluence Wiki or Jira issue tracker (from Atlassian) to use Globus user login and registration. A licensed add-on—available in the Atlassian Marketplace—enables the functionality. You must register your Jira or Confluence site with Globus and configure the add-on with your Client ID and Client Secret. A step-by-step example will walk you through the setup.

- Step-by-step example for Atlassian applications
- Register your application with Globus
- Add-on for Jira
- Add-on for Confluence