Identity Management Use Cases

July 19, 2019

Version 1.5

These use cases describe how researchers, scientists, educators, students, and other community members register themselves with the public research computing environment (thereby creating a community identity) and use their community identities to access applications and services.

IDM-01: Register with a community

A **person** needs to register with a community in order to use services and resources.

In most cases, the person should experience it as follows.

1. The person opens a web browser and visits the community website.
2. Then, the person clicks a "Create account" link or button and goes through the registration process.
3. Finally, the person completes the process, at which point the person should have a username and a password (and/or other credentials) that the person can use to login to websites and applications. Also, the community should now have a user profile containing the information provided by the person during the registration process.

We'll accept any solution to this problem, as long as the following are true.

1. The registration process should be secure from third parties.
2. The person may choose his/her own username as long as no one else has registered with it.
3. At least 100 people can use the registration system at the same time.
4. The system can register at least three new identities per minute.
5. The web interfaces should consistently respond to user actions in less than three seconds.
6. The registration interface should be available and working as described at least 99.9% of the time.

IDM-02: Login to a community's user portal with a community username and password

A **community member** needs to login to a community's user portal using his or her community username and password. We assume the user has already registered. (See IDM-01.)

In most cases, the community member should experience it as follows.

1. The user opens a web browser, visits the community's user portal, and clicks "Sign In" (wording may vary).
2. Then, on the Sign In page, the user enters his/her community username and password and clicks "Sign In."
3. If the username and password is incorrect, the user is told the username or password is incorrect and asked to try again.
4. If the username and password is correct, then the user is logged in.

We'll accept any solution to this problem, as long as the following are true.

1. In Step 3, a delay should be built into the response to inhibit brute-force password guessing.
2. The interface should provide responses to user actions in less than three sec for all steps EXCEPT the one in which the user inputs an incorrect username and/or password. That step could take longer, but probably shouldn't take less than 2 sec.

3. At least 25 people should be able to be in Step 2-4 at the same time, and at least 200 people should be able to be in Steps 1-2 simultaneously.
4. The interface must be available and working as described at least 99.95% of the time.

IDM-03: Change a community user profile

A **community member** needs to update or correct information in his/her community user profile so applications and other users using the profile receive accurate information. We assume the user has already registered. (See IDM-01.)

In most cases, the community member should experience it as follows.

1. First, the user logs into the community's user portal.
2. Then, the user goes to their profile page.
3. Then, the user clicks "Edit profile."
4. Then, the user changes some fields (e.g., email address, name, etc.).
5. Then, the user clicks "Save changes."
6. Finally, the user is redirected back to the profile page, which displays the updated profile.

We'll take any solution, as long as the following are true.

1. If the user has two-factor authentication enabled, Step 3 will require the user to authenticate via a second factor before proceeding.
2. If the user changes his/her email address, it should not take longer than one minute for the email system to emit a validation email message after Step 5.
3. At least 25 users can be using the profile interface at the same time.
4. At least three users can submit profile changes per minute.
5. The interface must be available and working as described at least 99.9% of the time.

IDM-04: Login to a community's user portal with an identity from another organization

A **community member** needs to login to a community user portal using an identity from another organization because it's easier than remembering a separate community username and password. (The other organization is most likely the community member's home academic institution or employer.) We assume the user has already registered. (See IDM-01.)

In most cases, the community member should experience it as follows.

1. First, the user opens a web browser, visits the community's user portal, and clicks "Sign In" (wording may vary).
2. Then, on the Sign In page, the user clicks "Other Sign In Options."
3. Then, the user is shown a page with a list of organizations. The user selects an organization to sign in with.

4. Then, the user will be redirected to the selected organization, where they login. (If the login fails, the user may try again.)
5. If the identity from the other organization has already been linked to a community identity, the user is logged into the community portal with the user's community identity.
6. But if the identity from the other organization has not previously been linked to a community identity, the user is asked if they want to register with the community or link an existing community identity.
    a. If the user chooses to register, they are taken to the community's registration page to create a community identity. (See IDM-01.)
    b. If the user chooses to link to an existing community identity, the user is taken to a community login page where the user enters the user's community username and password. When this is successful, the identity from the other organization is linked to the community identity and the user is logged into the community portal with the user's community identity.

We'll take any solution, as long as the following are true.

1. In Step 3, the interface should have familiar branding (appearance and style) for the community.
2. In Step 3, the community may choose which identity providers they trust as linked identities.
3. In Step 3, the interface should offer a way to "remember this selection" so the user doesn't have to see this page again.
4. In Steps 5 and 6, the user portal must receive the user's community identity (as opposed to any identity provided by another organization).
5. The interface should respond to user actions in less than three seconds.
6. The interface must be available and working as described at least 99.95% of the time.

IDM-05: Link or unlink an identity from another organization

A **community member** wants to link his/her identity in another organization to his/her community identity so he/she can use either identity when logging in to a community website or application. (Or, the user may wish to unlink an identity from another organization so it can no longer be used to login.) We assume the user has previously registered with the community and with the other organization.

In most cases, the community member should experience it as follows.

1. First, the user logs into the community's user portal.
2. Then, the user navigates to the user's profile page.
3. Then, the profile page displays a link to "Manage linked identities."
4. Then, the user clicks the link and is redirected to another page that lists the linked identities, allows each to be removed, allows new linked identities to be added, and allows the default organization to be changed or cleared (The last feature mentioned configures the "remember this selection" on the Other Sign In Options page, see IDM-04.)
5. Finally, when the user is done with these activities, he/she clicks "Done" and is returned to the profile page, which displays the updated information.

We'll take any solution, as long as the following are true.

1. The solution should support linking identities from most web-based federated organizations, such as InCommon-participating campuses, Google, ORCID, etc.
2. The solution should have future plans to allow linking X.509 identities and SSH public keys.
3. In Step 4, the interface for linking identities should have familiar branding (appearance and style) for the community.
4. At least 25 users can use the interface at the same time.
5. At least three users can submit identity linking changes per minute.
6. The interface must be available and working as described at least 99.9% of the time.

IDM-06: Login to a web application or science gateway with a community identity

A **community member** needs to be able to login to a **web application** or **science gateway** (a web application integrated with the community, hereafter referred to as "the gateway") with a community identity so the gateway can securely interact with community services on behalf of the user. We assume the user has previously registered with the community and the gateway has registered as a client of the community's login service.

In most cases, the community member should experience it as follows.

1. First, the user points a web browser at the gateway site.
2. Then, the user clicks "Login," "Login with <command name>," or a similar link or button.
3. Then, the user is directed to a community-branded login page. The login page asks the user to choose an organization for logging in.
    a. If the user chooses the community organization, the user can login using their community username and password.
    b. If the user chooses a different organization, the user will be redirected to the selected organization, where they login. (If the login fails, the user may try again.)
    c. If the user chooses a different organization and the resulting identity isn't linked to a community identity, the user is asked if they want to register with the community or link an existing community identity.
        i. If the user chooses to register, they are taken to the community's registration page to create a community identity. (See IDM-01.)
        ii. If the user chooses to link to an existing community identity, the user is taken to a community login page where the user enters the user's community username and password. When this is successful, the identity from the other organization is linked to the community identity and the user is logged into the community portal with the user's community identity.
4. Finally, the user is directed back to the gateway, and the gateway is given the user's community username and credentials that allow it to access community services on behalf of the user.

It should always work like this, except for the first time the user logs into the gateway. In that case, after Step 3 completes and before Step 4, the user is presented with a community-branded page informing the user that the gateway will have access to the user's identity information and any other specific

permissions requested by the gateway. Login only proceeds if the user agrees to give the gateway this access.

We'll accept any solution as long as the following are true.

1. In Step 4, the gateway must receive the user's community identity, as opposed to any identity issued from another organization.
2. The user must be able to revoke the permissions the user granted to the gateway at any time in the future. When this happens, the gateway will no longer be able to retrieve identity data or credentials allowing it to act on the user's behalf, and any credentials previously obtained will no longer be valid for use with other services.
3. The community's login service interfaces should provide responses to user actions in less than three seconds EXCEPT when the user enters an incorrect username and/or password. That step could take longer, but probably shouldn't take less than two seconds.
4. At least 25 users can be in step 3 at the same time.
5. At least 200 users can be in Step 4 at the same time.
6. The community's login interfaces must be available and working as described at least 99.95% of the time.

IDM-07: Login to a locally installed application with a community identity

A **community member** needs to login to a locally installed **application** (a command line program, graphical desktop application, or mobile application) using his/her community identity, such that the application can securely interact with community services on behalf of the user. We assume the user has previously registered and that the application is registered as a client of the community's login service.

In most cases, the community member should experience it as follows.

1. First, the user starts the application.
2. Then, the user is prompted by the application to login. The application directs the user to the web address of the community's login service. (On most systems this means a browser application will open with the login service's address loaded.)
3. The community's login service behaves as described in Steps 3 and 4 of use case IDM-06 (including the variation for first-time-use).
4. When the login succeeds, the user is returned to the application and the application receives an authorization code. The code allows the application to retrieve the user's community username and credentials. The credentials allow the application to access community services on behalf of the user.

We'll accept any solution as long as the following are true.

1. In Step 4, the gateway must receive the user's community identity, as opposed to any identity issued from another organization.
2. The user must be able to revoke the permissions granted to the gateway in Step 3 at any time in the future. When this happens, the gateway will no longer be able to retrieve identity data or credentials allowing it to act on the user's behalf, and any credentials previously obtained will no longer be valid for use with other services.

3. Fault codes must be well-specified and understood. (The application can programmatically interpret what went wrong from the fault code.)
4. The community's login service should provide responses to requests in less than three seconds for all cases EXCEPT when the user provides an incorrect username and/or password. In that case, the response shouldn't take less than two seconds.
5. At least 25 locally installed client applications can be authenticating at the same time.
6. The interface must be available and working as described at least 99.95% of the time.

IDM-08: Login to a locally installed application with an SSH/X.509 key

A **community member** needs to login to a locally installed **application** (a command line program, graphical desktop application, or mobile application) using an SSH key or X.509 certificate, such that the application can securely interact with community services on behalf of the user. We assume the user has previously registered, the user has previously associated an SSH/X.509 certificate with his/her community identity, and the application allows key-based authentication.

In most cases, the community member wants to experience it as follows.

1. First, the user starts the application.
2. Then, the user is prompted by the application for the password to unlock the user's local SSH or X.509 private key.
3. Assuming the password unlocks the key, the application then uses the key to authenticate to one or more services.

IDM-11: Use a community identity for InCommon authentication

A **community member** needs to authenticate to a service that accepts InCommon authentication. We assume that the researcher has previously registered.

In most cases, the community member wants to experience it as follows.

1. First, the researcher visits a service that accepts InCommon identities and initiates a login.
2. When prompted for an organization, the researcher selects the community as his/her InCommon identity provider, authenticates with the community, and authorizes the community to share his/her community identity data with the service.
3. Finally, the service receives the community member's community identity and provides appropriate access to the service in accordance with its use policies.

We'll take any solution, as long as the following are true.

1. The solution must be compatible with InCommon authentication protocols.
2. In Step 2, the authentication mechanism must use an approved community identity provider.
3. In Step 3, the service must receive the community member's research & scholarship attributes.
4. The solution should not require the researcher to have an active or prior allocation in the community.

IDM-12: Single sign-on across community OpenStack resources

A **researcher** or **educator** (hereafter referred to as "the user") wants to be able to authenticate once using his/her community identity and subsequently have authenticated access to all of the available OpenStack resources in the community. (These services might include things such as a central object library (e.g. images, date, etc.), objects stored by that user on other service provider's resources, or cloud computing features such as elastic scheduling of instances.) We assume the researcher has previously registered, has allocations on each of the OpenStack resources, and is authorized to use the OpenStack API on each resource.

In most cases, the **researcher** wants to experience it as follows.

1. First, the user would connect to a service provider's authentication web page.
2. Then, the user will be redirected to the Identity Provider's (IdP) web page.
3. Then, the user will be prompted for authentication credentials.
4. Then, upon successful authentication, the user will be redirected to the SPs endpoint and an unscoped token will be returned; included in this token are a list of Identity service groups that the user belongs to.
5. Then, this unscoped token can be used to identify which groups and domains are accessible.
6. Then, the unscoped token, along with appropriate domain and project information, can be used to request a scoped token.
7. Finally, with the scoped token, the user can access services and objects that are available to that user on other OpenStack service providers.

It'll always be like that, except when the user is not using a web browser, and the Enhanced Client or Proxy (ECP) is used as an alternative mechanism.

We'll take any solution, as long as the community login service returns an unscoped Keystone token.

IDM-13: Authenticate to a resource's OpenStack API

A **researcher** or **educator** (hereafter referred to as "the user") wants to use his/her community identity to obtain credentials that can be used to authenticate with the OpenStack API on a community resource. We assume the user has previously registered, is part of a project that has an allocation for the resource, and has been authorized to use the resource's OpenStack API.

In most cases, the user wants to experience it as follows.

1. First, the user connects to a web application provided by the service provider and logs in as described in IDM-06.
2. Then, the user navigates the service provider's site to find OpenStack API credentials associated with the user's project.
3. Finally, the user configures his/her application to use the OpenStack API credentials to authenticate to the OpenStack API and use any system features that are available to the user's project.

We'll take any solution, as long as Step 2 provides standard client_id+client_secret credentials that can be used with an OpenStack API.

IDM-14: SSH access using a community identity for education

An **educator** needs his/her **students** to be able to login via SSH with their community identities to a virtual machine (VM) he/she administers so they can use the VM and the educator doesn't have to assign new IDs and passwords or SSH keys to every student. We assume the educator and his/her students have previously registered with the community. (See IDM-01.)

In most cases, the **educator** should experience it as follows.

1. The educator prepares a VM for use in a class or a training session. The system provides SSH access, and each student in the class needs to login via SSH to his/her own local account on the system.
2. The educator creates accounts on the VM for each student using their community usernames and a system-generated initial password that the educator ignores.
3. The educator follows simple instructions provided by the community to configure the VM's SSH service in such a way that the service will accept short term login tokens based on community identities and map them to the students' accounts.
4. The educator continues following the instructions provided by the community and installs a web app on the VM that enables students to use their community identities to obtain short-term login tokens for the SSH service.

Each **student** should experience it as follows.

1. First, the student gives the educator their community username so the educator can create an account for the student on the VM.
2. When the educator informs the student that an account is ready, the student follows the educator's instructions and visits the web app for the VM, logging in as described in use case IDM-06.
3. Once logged in, the student can generate a short-term login token for the VM.
4. Once a token is generated, the student uses any SSH client to connect to the VM using their community username, and copy and paste the short-term login token into the SSH client when prompted.
5. The SSH server on the VM accepts the short-term token and the student is logged into the VM.

We'll accept any solution to this problem, as long as the following are true.

1. Students must be able to login to the VM using a standard web browser and a standard SSH client without additional installation or configuration.
2. In the educator's Step 3, the steps required to configure or replace the SSH service should be as simple as possible. Ideally, the educator could request the feature when requesting the system and it would come pre-configured.

3. In the educator's Step 3, the configuration instructions should work with a standard OpenSSH server pre-installed on the VM or available via the VM's native packaging system.
4. In the educator's Step 4 (and student's Step 4), the educator's system must not have access to the students' authentication credentials (e.g., passwords).

History

	Version	Date	Changes	Author
Entire Document	1.2	7/29/2015	Use cases 1-10 documented in preparation for an XSEDE-1 re-implementation activity.	Maytal Dahan, Steven Tuecke
Entire Document	1.3	1/6/2017	Use cases 1-10 rewritten using XSEDE-2 format; revised use case 3; added use cases 11-13.	Lee Liming
Entire Document	1.5	7/19/2019	Removed unnecessary XSEDE terminology; removed use case IDM-10; changes to IDM-14	Lee Liming