Cloud Services Cookbook

In 2015, IAM managers from the US academic consortia called the CIC put together the CIC Cloud Services Cookbook. This document offers a set of recommendations on how vendors of cloud services and institutional customers should engage with each other to support federated identity.

As part of the 2016 Workplan (see REF16-3C), the REFEDS community aims to extend the Cookbook so it covers a more global scope. This wiki space will capture the ideas and suggestions and be the home for the adaptation of the Cookbook.

Notes: Specific mentions of InCommon have been highlighted; we need to figure out if those recommendations can be generalized to all federations. Comments have been added to wiki on other specific areas that will need to be modified.

- [ ] Some of the material is already slightly out of date. Would the effort to globalize this also focus on updating the material? How often do we want to commit to reviewing the material?
  - [ ] Proposal: Updating the document should be on the list of things to do, but at a lower priority than globalizing the existing material. Frequency should match the more local updates done by the CIC schools.
- [ ] If we want this to be more globally applicable, perhaps a bit more on the need to establish SSO on the campus or within the VO? They don't explain why federated authentication in a single campus setting is useful (that it is a logical development out of campus SSO). Also, it would be helpful to have references and definitions.
  - [ ] Proposal: Mention some highlights here, but otherwise point to the videos being produced by the NSRC and REFEDS on the topics.
- [ ] Perhaps add more regional case studies?
- [ ] Perhaps a section on cloud services as accessed through eduGAIN?

Executive Summary

The CIC Identity Management (IdM) Working Group has identified federated integration with "cloud" services as an area which needs special attention by the group in order to lower barriers to adoption by CIC institutions. These integrations are often complex, requiring large amounts of staff time in the policy decision-making process, the architecture of technical integrations, and result in duplicate effort between CIC institutions. It is important for the CIC institutions to present consist procedures and practices (when possible) to vendors in order to simplify the integration process and ensure that best practices are followed on the part of the vendor and the institution.

What follows is a DRAFT set of recommendations in a "do's and don't's" format that we hope will result in a recipe for success for the institutions and vendors. The guidance will be most relevant to those who sponsor, administer or support Identity Provider and Service Provider services on both the campus and the vendor sides. The content is being contributed by staff from across the CIC. Where broad consensus was not achieved (or is not anticipated) the word "consider" is used in place of "do" or "don't".

Problem Statement

Partnering with a new cloud service provider can be complicated and time-consuming. Nearly all of the hurdles can be overcome with technical or policy solutions, but forming those solutions can be an expensive process, which diminishes the value proposition of adopting cloud services.

Overview of Higher Education IDM Landscape

The identity management systems of colleges and research universities are by rule more varied and complex than in most similarly-sized corporations. Many universities source identity data from multiple authoritative systems, including HR, student records, and other systems used to identify more loosely affiliated individuals. Data from these competing sources is often reconciled into a single, master directory intended to offer the "best" identity data for the organization.

Higher education user role definitions also differ greatly from private industry. A particular user will often have multiple organizational roles (for example, a graduate student may also be considered faculty if assigned certain teaching responsibilities), and in some cases the user's *primary* role may be difficult or impossible to discern. Furthermore, user roles change very frequently (imagine a staff member who enrolls for classes one semester, but not the next), and most universities track a large number of "edge cases" (visiting faculty, adjunct faculty, special student types, emeritus, allied staff, and so-forth) each of which come with their own business rules and related service provisioning challenges. Faculty and staff may have a number of affiliations with different operating units, which in some cases may involve multiple email addresses and other identity data complexities. Depending on their responsibilities, some users may require service resources that fall outside of their primary role (for example, staff who support student systems), and this access may be achieved through one-off methods or the use of ad hoc provisioning processes.The breadth and complexity of this landscape is often surprising to vendors and commercial service providers who are new to the university environment.

Student and staff turnover is a challenge for higher ed identity management, and requires markedly different processes for user creation and service provisioning and de-provisioning. Each semester or quarter a university may process thousands of new student records, resulting in predictable but significant traffic spikes to service enrollment systems and help resources. (A cloud vendor that provides direct support to higher ed users may see a 100x spike in new user support requests during the opening days of a new semester.) During the same periods many existing users will undergo role changes (for example, undergraduate students matriculating to graduate student status) that also contribute to start-of-the-semester provisioning and support loads.

At the other end of the pipeline, student and staff de-provisioning presents a particularly difficult challenge for universities. It is sometimes said that "no one ever really leaves a university," and this viewpoint is reflected in higher ed identity management practice. Unlike the corporate world where when an employee leaves, the company retains their data and the user is expunged from the directory, de-provisioning in the higher education setting is usually carried out in large, end-of-the-semester batches (sometimes referred to as "de-provisioning runs") that can hinge on varied business rules with many exceptions and special cases. Privacy rules prevent universities from asserting control over an exiting user's data, and graduating students and retired faculty may retain access to many IT resources. Students or faculty/staff may leave the university, then return later in the same or different role with the understanding that they will reclaim their previous credentials.

Universities also play by different policy and compliance rules than private industry. Expectations for data privacy is higher, most users (especially faculty and students) are considered the primary owners of their data, and higher ed security staff are often interested in keeping watch over service logs related to administrative activity and privilege escalation. FERPA and the proper handling of FERPA-related data is an ongoing concern for IT managers, and can have surprising consequences (for example, the need to offer group services with suppressed member listing) for new service deployments. Schools must also manage legal requirements surrounding service accessibility, and thus show strong preference toward vendors that offer accessible products or can demonstrate a clear accessibility roadmap.

Managing the risks involved with federated identity and cloud services requires that all parties understand that managing risk does not mean eliminating risk. While the topic of risk management is beyond the scope of this document, identity providers as well as service providers are strongly encouraged to review the topic and to handle risks based on the likelihood as well as the possible outcome of the threat.

University organizational culture tends to favor a consensus-driven decision process, and central IT units often control only a fraction of the infrastructure and services available to the campus. As a result, it is usually very difficult to mandate top-down change, and system-wide IT changes usually require buy-in from a diverse range of stakeholders in both technical and non-technical positions. This decentralized approach to IT also leads to extremely varied campus computing environments supporting a diverse range of work and research activities.

Determining Who Our Users Are and What They're Allowed to Do

Authentication

One of the biggest benefits of federating is letting the user’s home organization handle the authentication process. The user can use their campus credentials to sign into a federated service without the SP needing to store passwords or provide mechanisms for users to change them or reset them if forgotten. The service provider can hand the user off to their home organization’s identity provider to log in and, once complete, be handed back information about the user’s identity. This approach is the preferred method in a federated scenario. It allows the campus to do their part in verifying a user’s identity through whatever means they’ve established. It also allows the user to not only have a single password, but in the case of single sign-on, only need to authenticate once to log into several applications.

This isn’t the only way to provide authentication for a cloud service, of course: there are less “federation friendly” methods such as:

- service-specific passwords
- password syncing from the user’s home organization
- calls to the user’s home campus LDAP or Kerberos service (or other alternative).

With some applications, one or more of these methods may be necessary.

Both Campus and Vendor: DON'T assume successful authentication means the user is authorized for service.

While convenient for a service to simply hand off a user to their home campus identity provider and get back the attributes it needs, authentication is only part of the story. Just because a user is able to authenticate doesn't mean that they are entitled to access the service. Entitlement to the service, or authorization, is addressed in its own section below. Within the context of authentication, it's enough to be aware that a successful authentication is usually not the only decision required to authorize the user for access to a service. In particular, vendors must understand that most universities will issue accounts to many different constituencies, often including guests, parents of students, contractors, and even the general public. This is a major difference from most traditional enterprises.

Vendor: DO let the identity provider handle authentication

There are few good reasons to handle authentication within your service instead of leveraging the benefits of federated authentication. A user's home organization can handle authenticating the user and releasing the needed user attributes to your service. This means that you don't have to keep track of a user's password or provide a mechanism for them to reset it. As long as you have the proper trust models in place to know that the user's identity provider is releasing a valid assertion to your service, it makes sense to let the identity provider do its job. It also provides benefits to the user, allowing them to use their campus password for federated services and, with single sign-on, preventing them from having to enter their password as often.

Faced with more sophisticated and more frequent phishing attacks, campuses are increasingly looking beyond simple password-based authentication. As one result, the take-up of multi-factor authentication is accelerating. To the extent that cloud services support federated authentication using campus Identity Providers, those services will benefit from campus-based moves to stronger authentication. Federation makes deployment of these technologies easy to extend to cloud services with little or no impact on vendors.

Vendor: DO rely on browser-based authentication for non-browser applications.

There are scenarios in which federated authentication is difficult. For a more historical example, federating authentication for services such as POP and IMAP isn't trivial to accomplish. The technology isn't available with all POP and IMAP servers out there, and the protocols supporting this are unsettled. Similar issues apply to newer examples such as mobile applications.

It's a common and reasonable approach in such cases to leverage a browser for authentication but transfer some evidence of the result to the non-browser application by various means. This preserves the ability to federate the authentication process and the autonomy of an organization to control the login experience and to support stronger authentication mechanisms, as discussed below. When use of a browser is impossible or impractical (e.g., for command-line applications), there are other options that still support federated authentication, such as the SAML ECP (Enhanced Client or Proxy) profile and the use of the EAP framework, currently being standardized for use with Internet applications by the IETF.

Note that when using a browser, DO rely on the standard browser application(s) on a platform rather than encapsulating the process in a special window or embedded browser. The latter approach degrades security by obscuring a platform's native signaling about the security of the connection to the organization's login page.

OAuth is also an important technology in this problem space, and is well-suited to delegating access to an application to another service, without trusting the third party with the user's password and requiring it to screen-scrape a web-based login. It can also be used in the mobile space, and in the future may be a viable solution for legacy protocols like the aforementioned IMAP example, but in such cases it generally requires a browser-based login, as discussed above.

Vendor: DON'T use service-specific passwords unless there are no alternatives.

When there are no alternatives, service-specific authentication might need to be employed. Making password authentication or syncing available off campus may be frowned upon by security staff, but the campus might consider providing a campus-hosted service for setting a vendor-specific password. This works well when the service provides an API with support for setting user's service-specific passwords.

This approach is generally better than offering a vendor-specific UI for password management because a campus can integrate the process into its traditional password management tools and policies, as well as enforce rules such as preventing overlap between passwords. Making it possible to disable functions within an application for managing an application-specific password can be critical; having multiple places to change the password can be confusing at best and error-inducing at worst.

Vendor: DO use forced re-authentication when appropriate.

When there is a significant risk that the current user is no longer the same as the person who authenticated, use forced re-authentication to reestablish the user's identity. Be aware, however, of the user frustration caused by frequent prompts to authentication. (See "DON'T over-use forced re-authentication" in the User Experience section for more information.)

Campus: CONSIDER stronger authentication over password strengthening.

Phishing is consistently identified as the most common authentication challenge at universities. Traditional metrics of password strength like entropy and a focus on mitigating brute-force attacks by requiring complex passwords, constantly changing them, and locking out accounts do nothing to address the phishing threat and annoy users greatly while raising costs.

Adding additional factors such as one-time codes, SMS messages, etc., or deploying true multi-factor solutions with hard or soft tokens are likely to be more cost effective and of more benefit. Federation also makes deployment of these technologies easy to extend to cloud services with little or no impact on vendors.

Take care however to think through how the various request, issue, activation, and reset processes are designed because it's relatively common to introduce dependencies on password authentication that can undermine the value of the additional factors.

Identifiers

An *identifier* is a special kind of attribute that is specifically designed to distinguish each account/subject/user/thing from its peers in a particular set. While almost any attribute may contribute to differentiating a subject from similar subjects, identifiers are intentionally designed to do this by themselves. It is common for a subject to possess several different identifiers, used for different purposes or generated by different information sources.

With most applications, the primary function of identifiers is to identify its users. This process is more complicated when an application relies on a system other than itself to handle the authentication process, referred to above as *externalizing* it. Such an application no longer fully controls the way the identifiers look and behave, and must therefore make sure that its needs can be met by clearly defining the properties it needs from an identifier. (**Appendix A** discusses a number of these properties.)

The use of *federation* adds yet another layer of complexity because the application may not even be operated by the same organization that is providing the authentication function, and the identifiers. This makes the adoption of standards and common practices essential, because an IdP service cannot be tailored to meet the specific needs of every application; instead it should provide a sufficient set of identifiers of different properties so as to meet a wide range of application needs.

Further, an application supporting the use of multiple IdPs across many organizations at the same time has an even more compelling need for standards, because handling widely varying approaches to user identification at the same time greatly increases application complexity and creates opportunities for bugs and security exposures.

Both Campus and Vendor: DO support a varied set of identifiers.

Many standard attributes are available for obtaining a user's unique identity. An IdP should support a variety of standard and proposed eduPerson identifiers, using a core set of underlying identifiers managed by an IDM system. A vendor should base user identification on at least one of these attributes and, for improved flexibility, more than one. Particularly valuable are the *eduPersonPrincipalName*, *eduPersonUniqueID*, and *eduPersonTargetedID* attributes. The latter two have quite well-defined properties, the former less so but it does have much wider adoption within the US higher education sector.

Both Campus and Vendor: CONSIDER the use of eduPersonTargetedID where appropriate.

The definition for *eduPersonTargetedID* can be intimidating, but once you take the time to understand it, it's a relatively straightforward and very useful attribute for applications that are designed with privacy-preserving assumptions and/or don't require attaching personally identifiable details to application accounts.

Like other identifiers, it uniquely identifies a user. In additional, *eduPersonTargetedID* has the advantage of being not only opaque but also non-correlatable, meaning that SPs can receive a unique identifier for a user without knowing the user's real identity or linking identities across services. Vendors should consider the support of this attribute if they need to identify a user internally without receiving any personally identifiable information. Identity providers can typically configure this attribute easily: if you have a stable "serial number" (which may be usable to populate *eduPersonUniqueID*), then simply use it to feed a salted hashing algorithm to generate pairwise *eduPersonTargetedID* values. Software such as Shibboleth can do this for you automatically.

However, *eduPersonTargetedID* is a particularly poor fit with applications that display identifiers to users. See "DO use appropriate attributes for friendly names," below. This is especially true when identifiers are used in the selection of other users of the application as is common to the sharing of resources. *eduPersonTargetedID*'s values are entirely unsuited to display or "externalized" use of any kind and should never be supported in such cases, to avoid a very poor user experience.

Both Campus and Vendor: DO use standard definitions of identifiers and attributes.

Resist the temptation to force something you have into something you need. If you can't support a particular identifier or meet a particular requirement, don't populate it in a way that violates its requirements or expectations. Similarly, don't create a new attribute for a specific application that has the same specifications as an existing standard attribute. In a federated system, attributes are like programming language contracts. It's better to create your own contract (that is, define a custom identifier) than to violate one. Federated protocols make defining new attributes easy. This doesn't mean you should create new attributes all the time, just that not following the rules is worse than doing so.

Both Campus and Vendor: DON'T mistake eduPersonPrincipalName for a valid email address.

In the "real world" the only identifier of consequence is the email address. Unfortunately, the campus email environment is much less "clean" than a typical enterprise, and the standard attribute for email addresses, mail, is not well-suited for user identification. It is multi-valued, and explicitly not meant as a controlled attribute limited to enterprise-assigned values. This creates a problem, since many applications want a single identifier to triple as email address, application key, and discovery hint, requiring it to be suffixed with a common domain across all users of an application from a particular IdP.

An IdP should carefully consider whether expediency dictates the population of *eduPersonPrincipalName* with a specially-managed, enterprise-scoped, non-reassignable email address. There are good reasons not to do this, but one really big reason to do it: integrating with cloud services will be vastly easier in the near to medium term (possibly longer). At the same time, vendors should think twice before assuming that the value of *eduPersonPrincipalName* is a deliverable email address, though by design it looks like one.

Campus: DO standardize internally on a stable "serial number" for users.

Much of the work in supporting good standard identifiers is simplified by a stable, opaque, well-managed underlying identifier for the users managed by an IDM system. This value should never be related to a user's more traditional attributes, and should be as stable as possible, be of fixed (and reasonable) length, and ideally be alphanumeric to avoid confusion over numeric formats. Effort to minimize duplication of identities pays off in the prevention of remediation needed later to change an identifier that's already in use.

Campus: DO make eduPersonPrincipalName useful.

The "letter of the specification" for *eduPersonPrincipalName* permits a wide range of identifier properties. But practice with this attribute today creates the expectation of a number of properties that while not strictly required, are for all intents and purposes de facto requirements of applications that rely on it.

- DON'T reassign them. This is perhaps the most critical property. Almost all applications today have no good way of handling the reassignment of an identifier to a different user. That is, most do little in the way of *de-provisioning*, which is a difficult problem when there are lots of federated, loosely coupled applications, exactly the sort that tend to rely on this attribute.
- DO use human-readable, recognizable values. Opaque identifiers are fine for some purposes, but there are other attributes better suited to that requirement, and *eduPersonPrincipalName* is in the gray area between an opaque identifier and an email address. Applications *will* display the values, often to associate users with actions and content, and users will be expected to handle them in many cases.

Authorization

Whenever a user tries to access a service or to perform a specific action with an online resource, a decision has to be made about whether to allow or deny the request. This process of evaluating a request and making an allow/deny decision is called authorization. Since this is such a ubiquitous situation the authorization problem appears in many guises covering different types of resources and actions, and different granularities of access. Some considerations follow.

Both Campus and Vendor: DO leverage eduPerson attributes for authorization.

The eduPerson specification provides a few different attributes that are useful in different scenarios for communicating authorization decisions or policy inputs to a service.

The URI-valued *eduPersonEntitlement* attribute is well suited for passing a computed policy decision or communicating different levels of access to a service via globally unique values. As a prerequisite, the campus and the vendor need to agree on how specific values of *eduPersonEntitlement* map to specific authorizations within the vendor's service. Then the campus asserts the appropriate *eduPersonEntitlement* values for users that should have a given level of access to that service. The *eduPersonScopedAffiliation* attribute, with values like faculty@foo.edu, student@foo.edu, etc., is sometimes useful for very broadly defined rules for authorization, but is not a good choice for services that need precisely defined classes of service.

One more attribute that has the potential for broad utility is *isMemberOf*. The values of this attribute are identifiers for groups to which the subject belongs. Unlike *eduPersonScopedAffiliation*, the values of *isMemberOf* are not constrained by a controlled vocabulary. The institution is free to define and assert any groups it likes, as long as the group identifiers are unique from the perspective of the service. For this reason, a good practice is to use URI-valued group identifiers, which ensure global uniqueness.

Both coarse grained and fine grained authorization can be supported with *eduPersonEntitlement* and *isMemberOf* attribute values. When an authorization policy determines whether a user gets into the application or service at all, that can be termed coarse grained authorization. It is analogous to controlling who can open the front door. When an authorization policy gets more specific about which actions a user can perform on which resources inside the service, it is an example of fine-grained access control. Groups and permissions can be more general or more specific as needed to address a broad range of granularities of access.

Both Campus and Vendor: DO be clear about where the allow/deny decision logic is evaluated.

There is no question that enforcing an authorization *policy* is the service owner's right and responsibility, although typically the policy itself may be crafted by the institution, or as a cooperative activity with the service owner based on licensing. But the steps that lead to the enforcement of that policy may be divided between the institution and the service provider in many different ways.

For example, a service provider and an institution might agree on a specific URI value for the *eduPersonEntitlement* attribute to express "the bearer is a legitimate consumer of your service under terms of our contract xyz". The IdP may then determine and assert that value (or not) at the time the user accesses the service. A successful example of this approach is the acceptance of the IdP-asserted *eduPersonEntitlement* value "`urn:mace:dir:entitlement:common-lib-terms`" by libraries and online content providers to imply that a user is entitled to access generically-licensed material online.

It's also possible that an SP will want to receive raw user attribute information from the IdP so that they can evaluate appropriate business logic at their end. Attributes may carry information less specific to services such as affiliations or group memberships. One challenge to this approach is that variations from one IdP to another in the meaning of those raw attributes will complicate a service's decision logic.

Both Campus and Vendor: DO determine whether a service is dependent on service-specific "local" user accounts.

Many if not most cloud services are designed to require local user accounts; that is, some kind of application-specific database or directory entry storing preferences, profile data, application history, ec. In such cases, institutions will need to address the creation of those accounts at the service through some form of provisioning (see the next section).

With a simple enough service model, merely provisioning an account may take care of the authorization problem. In these cases, the authorization decision is made by the institution based on service eligibility rules determined in cooperation with the cloud service provider. There are many ways the institution can compute the eligibility but in the end an allow/deny decision will translate into a decision to either provision an account for the user or not. However, if the user has previously been provided with a service-specific account, then a deny decision may need to trigger an appropriate de-provisioning action.

In some cases, often for audit reasons, institutions may opt to leave service-local user account in place, and seek to block subsequent access by invalidating the user's authentication credentials. Invalidating authentication credentials is a very blunt tool, and obviously affects access to any and all IdP-mediated service access. De-provisioning via credential invalidation is clearly sub-optimal and typically represents a situation in which the cloud service provisioning model doesn't adequately support de-activation or deletion of its local user accounts.

Whether or not a service maintains local user accounts, information bearing on the allow/deny decision can be passed to the service at each instance of user access to the service. This is typically labeled attribute-based access control (ABAC). For example, an attribute that asserts a user's entitlement to access a service under a contract with the institution might be all a service needs to determine access. In that case, *eduPersonEntitlement* would be an appropriate attribute to convey the needed information.

Provisioning and De-provisioning

When a cloud service design depends on a specific, dedicated source of user information, and an organization contracts for that service on behalf of its members, some process must be in place to keep the service-specific repository current and consistent with the organizational information on members. This process is referred to as provisioning. When organizational members lose their eligibility for a service, there should be a corresponding de-provisioning process to remove them from the service user store, or to mark them as inactive.

Provisioning is one of the most complex integration challenges for sites adopting a cloud service. Each service offering tends to come with its own proprietary provisioning approach. For an institution adopting multiple cloud services, this means devoting valuable staff resources to an ongoing series of one-off integration projects. Service providers and consumers would both benefit from standardizing the provisioning and de-provisioning process.

Just-in-time vs. Just-in-case approaches to user provisioning

There are two fundamental models for provisioning external services: Just-in-case and just-in-time. Just-in-case provisioning involves calling the vendor's provisioning API whenever a new user becomes eligible for the service, whether or not they may actually ever use it. This model can be simple, but will mean more traffic between institution and vendor, and more user information in the hands of the vendor. The primary alternative is just-in-time provisioning: Only when a user actually attempts to access the service does the provisioning step take place. This requires more sophisticated coding by the institution: Essentially a side-flow operation has to detect that a first-time service user is attempting access, and then "run ahead" of the user to create the necessary user record on the service side. Some vendors support a model of detecting first-time access at their end, but this is relatively rare.

Campus: DO expect the typical vendor to have a single, set model for creating user accounts on their systems.

Many cloud services depend on having a local record for each of its users. Often the design center of their original service offering was individual sign-up for the service. They may support institutional licenses for access to their services, but the model of a single user establishing an account on their side is often present just under the surface. At most they will have defined a single model by which a client institution can initiate provisioning of user accounts for their service. The burden is on the subscribing institution to understand the vendor's model and conform to it. This puts an increasing burden on the institution as the number of cloud services scales up. The more cloud services, the more integration models for user provisioning the institution will end up supporting. For example, a cloud service provider may specify that user information is made available in a specifically configured LDAP directory (often AD). If the institution supports two vendors, most likely two directories will have to be configured and populated.

Campus: DO practice "defensive programming" when setting up provisioning services.

Experience has taught us that in many cases, the documentation on provisioning models from the service provider is incomplete, or worse, simply incorrect. The institution-side integrators must then adopt a "defensive programming" style that features extensive testing for error conditions or simple uncaught failures, including setting up rich logging to support diagnostic work. Based on testing results under realistic conditions, campus staff will need to put in place counter-measures to detect and remediate failed operations. Seemingly random fluctuations in responsiveness under load are not uncommon and again require detection modes and counter-measures. This may or may not be documented in the service provider's guidance on "throttling" of provisioning operations. Be prepared for unannounced and possibly undocumented changes in the vendor provisioning API.

Campus: DON'T require out-of-band acceptance of Terms of Use.

It's a common practice for legal teams to insist on user acceptance of a "Terms of Use" page with many "end-user" focused cloud services like storage or personal web site hosting. Often this is done "out of band" by requiring users to agree to terms before their account is provisioned to the service, rather than "in-band" such as during a user's first login to a service.

Ideally such usage terms should be moved entirely away from use of the service and combined with a university's existing process for granting users access to the campus network and other common services, rather than repeated for each service. In particular, relying on an out of band approach makes the use of "just in time" provisioning harder, and many cloud services of this sort tend to favor that approach. "Just-in-time" also eliminates half of the provisioning problem, reducing the need for the campus to develop and maintain as much integration logic. If services begin to support attribute-based de-provisioning, all of this burden can be eliminated, making the cost of imposing usage terms out of band even higher.

If terms must be imposed, consider making the IdP the point of integration for this function.

Campus: DON'T expect robust de-provisioning support.

Currently there is little vendor support for meaningful de-provisioning, that is removing users' access and their external user record when they lose eligibility for the service. The risks flow back to the institution and the user: services are accessible to people who should not have them and user information remains with the vendor even though there is no longer a reason for them to have it.

Campus: DO handle username changes.

Service offerings are often premised on the idea that user identifiers are immutable. When changes of identifier occur for a user, it will be up to the institution to find a way to update the service records to reflect these changes. The approach will vary from service to service, adding another support burden that grows with the number of contracted services offered by the institution.

Vendor: DO support just-in-time provisioning based on user attributes passed in SAML assertions whenever possible.

If access to your service is made via the SAML protocol or similar alternatives, then each time a user authenticates to your service a number of attributes about that user will be passed to your application. Such attributes can include, among other things, user identifiers, email addresses, roles, group and entitlement values, all of which the IdP is authoritative for. Role, group and entitlement attributes are likely to bear on whether the user is authorized to use the service. The attribute information reflects the current state of user information in the Identity Provider's data store. As such, it is likely to be more up-to-date than any information stored at the service-side. This information should be used at the start of a user's first session to performing any provisioning required by the service, unless there is a business process requirement for establishing an out-of-band provisioning process. An example of such a requirement would be that the service must send electronic mail to its users prior to their first login session, informing them when they are required to perform some task within the service.

Vendor: DO keep server-side user information current based on user attributes passed in SAML assertions.

If a local user record is maintained by the service, it is essential to refresh that information store with up-to-date information from the IdP whenever a SAML assertion is received, rather than relying on out-of-band updates.

Vendor: DO consider standardizing your provisioning (and de-provisioning) APIs.

New service providers should evaluate the emerging SCIM specifications now working their way through IETF. SCIM defines a core identity and access management attribute schema, a method for extending the schema where necessary and a well-defined RESTful API for passing this information from identity stores to external services in need of provisioning. To date, de-provisioning support has been primitive where it exists at all. The draft SCIM specification addresses both provisioning and de-provisioning as complementary capabilities by design.

Vendor: DO manage your provisioning API in a way that respects the service subscriber interests.

In the early releases of a new cloud service, its provisioning APIs will understandably be subject to evolutionary change. Make sure that customer-facing documentation is accurate and up-to-date. Provide pro-active change management information to subscribing institutions, and make sure that error messages and logging are adequate for the subscribers' troubleshooting needs.

Working within the Federation

Identity Provider Discovery

In a federated environment, a service must accept logins from multiple organizations. A "discovery service" is one term for a standard way of letting a user select their home organization or chosen authentication source and be guided to the right IdP. The lack of built-in support for this concept in web browsers has been a thorn in the side of federated identity since its inception, and there are a variety of imperfect solutions used today.

Campus: DO provide a discovery service if you operate multiple IdPs.

Some organizations operate administratively as separate divisions, or campuses in the case of some university systems, and operate multiple IdPs. One way of dealing with this is to operate a proxy IdP as a stand-in for the various real systems behind the scenes, hiding the multiplexing that's going on from federated partners. This can often be a good approach, but may not be practical in some organizations.

However, operating distinct IdPs will frequently cause problems if the parent organization negotiates with vendors as a unit, and particularly if all users share a common email domain. Vendors often use email domains as a proxy for connecting an access attempt to a specific IdP, and breaking that 1:1 relationship may be difficult.

It will be advantageous in such a case to be prepared to assist vendors by operating a discovery service as a standard shared service for different applications to use. This isn't a panacea; most vendors will not support the concept of a discovery service without customizing their services, and it can introduce extra steps for users to go through, but it's better to be prepared with a viable solution than to go in unprepared and expect to be accommodated.

Campus: DO socialize the use of organizational email addresses.

Some universities have a strong "email" culture in which use of campus email addresses is common and encouraged. Others tend toward a more laissez faire attitude that fosters use of personal email addresses, particularly by students. Those in the latter camp are going to encounter major difficulties with many cloud projects, as the use of organizational email addresses often double as a user identifier (frequently one that is displayed to other users), and in particular as a discovery mechanism.

It's increasingly common to find applications that combine federation with traditional consumer access by using an email address entered by the user as a hook to recognize a SSO-enabled account and assume the IdP to use. Organizations that want, or require, users to use their institutional identity may need to increase user education and socialization around their own email domains or be faced with a de facto migration of user access to personal identities.

Vendor: DO provide a discovery mechanism for federated login.

Cloud services often tend not to address discovery because of a focus on the enterprise outsourcing use case of a single set of users accessing a compartmentalized service. In such cases, discovery is often "optimized out" by tying an instance of a service at a particular URL to the IdP to use. Even when this makes sense in the context of a particular service, a single organization may have multiple IdPs representing different campuses or user populations. Thus, avoiding discovery is ultimately a losing battle, particularly (perhaps uniquely) in the university sector.

In addition, many cloud services supporting collaborative use cases (file and media sharing, scheduling, surveys, and so forth) intrinsically need to consider the fact that a resource's location should **not** dictate the IdP to use. Doing so, as is quite common today, has a very negative effect on the viability of federation because it makes sharing resource links among colleagues difficult or impossible.

The most common discovery mechanism, when one exists at all, is typically entering an email address so that the domain can be associated with an IdP. As discussed elsewhere, this presupposes that the email address is itself a viable identifier (often not the case) and that a user will even think to use a university email address when asked for one. This approach intrinsically biases users towards Google and other large email providers, and tends to confuse in general. It also causes complexity when a service has existing accounts tried to a university's email address if that university wants to subsequently contract for the service.

Resist the temptation and deploy a local discovery service that prompts the user for the home organization directly. Many such tools exist, and integrate easily into applications and their look and feel. In exceptional cases, if a service is offered solely to participants in a single identity federation, a service offered by the federation may also be a possibility. Such centralized services may seem attractive in other cases, but are often too limiting to address requirements that inevitably go beyond the bounds of a single federation.

Technical Trust Framework

In a distributed system, security measures and controls are applied to protect exchanges of data and identity between cooperating systems. Depending on the technical measures involved, different kinds of data must be "trusted" as true/valid in order for a particular control to be safely applied. The use of cryptographic keys is a common example; keys must be associated with the right systems. Much of a system's technical risk is derived not only from the strength of these measures, but also the mechanisms by which trust is established in the underlying keys and data that such measures require.

Within a single organization, it's common for trust to be implicit or managed out of band because of proximity and convenience. It's often simple to transport a key from one machine to another, or to rely on direct communication with all of the system owners affected by a change or an exposure. Voices on the phone may be recognizable and are often sufficient to prove the identity of staff to one another. Across organizations, these traditional approaches don't suffice and one often observes a substantial degradation in the trustworthiness of the mechanisms used as a substitute, with unsecured e-mail being among the most common means of exchange.

Moving beyond that low bar is expensive on a partner by partner basis. Trust frameworks are a mechanism to elevate the level of technical trust between systems operated by otherwise independent parties. An often overlooked advantage is that they can address not only the initial establishment of trust but ongoing maintenance in the face of both routine changes and less frequent, but inevitable, breaches of security.

Note that while it is common for trust frameworks to encompass both technical trust and notions of behavioral trust (adherence to policies, contractual requirements, legal frameworks, etc.), it is more effective to consider them separate requirements that can be addressed in different, though sometimes overlapping, ways.

It must also be observed that organizations certainly have different perspectives on risk. Trust measures sufficient for one organization may be deemed wholly insufficient by another. Risk is also driven by the value of information being protected. One of the benefits of trust frameworks is to establish a strong baseline from which to manage many different relationships, even if many (often most) of those relationships might not warrant those protections in their own right.

Both Campus and Vendor: DO be prepared for the case in which a campus or vendor drops their membership in a formal identity federation.

The historical trend has been for identity federations to attract new participants and grow steadily in size over time. However it can happen that a given participant may drop from the federation and participants should plan for this contingency by having strategies to transition operationally to a bi-lateral relationship if appropriate.

Both Campus and Vendor: DO register SAML metadata with a federation.

Higher education institutions and vendors that provide services to them should view participation in an identity federation in much the same vein as acquiring an Extended Validation certificate for sensitive web services. The process of joining and registering metadata produces a strongly validated certification of the network endpoints, keys, and organizational contacts associated with an IdP or SP. While not every federated service supports SAML metadata, an increasing number do, and those that don't usually require substantially the same information, if by another name or in a non-standard format.

Federated services offering access to their own valuable data may require non-trivial verification of the information carried in metadata before accepting access from a new IdP. Having one's metadata available through an identity federation simplifies the job of technical staff in making the information available to such services, even when those services are not themselves operated by members.

Vendors offering federated access to an organization's own data (such as when outsourcing services) flip the trust equation such that the IdP organization may itself want to impose requirements on the vendor around the acceptance and maintenance of the IdP's metadata. Federations provide a good solution to that problem.

Both Campus and Vendor: DO define a process for maintaining up-to-date SAML Identity Provider and Service Provider metadata.

A SAML IdP's operating procedures should include a process for provisioning new Service Providers, and SAML metadata is the recommended standard for this purpose (even if one's software lacks native support for it). Apart from standardizing the format, subsequent requirements will depend on the risk tolerance of the organization toward the potential exposure of data in the event that incorrect information were to be registered.

For some organizations and some kinds of data, a simple e-mail exchange may be sufficient, or in some cases no exchange at all and an open policy toward use of the IdP service (referred to in Shibboleth as support for "unverified" services). It may be better to be explicit about not requiring any trust in an SP than to bother with a manual, low assurance process that increases cost without providing any meaningful benefit.

If any degree of trust in the metadata is required, then obtaining the metadata should not be viewed as a one-time process. A permanent means of exchange should be established in order to address the risk of a key compromise (as well as addressing operational agility, discussed later in this document). Requiring SPs to register metadata with a federation is a practical step one can take to elevate the level of trust in one's partners and address maintenance.

While it is quite common for commercial SPs, especially larger ones, to self-publish metadata documents, the practical fact is that this approach does not scale unless the information is accepted blindly, which in turn acts to limit the level of trust or the number of SPs supported (or both). Federations can elevate trust without sacrificing scale.

Both Campus and Vendor: DON'T expose untrusted URLs to users.

An IdP or SP may have occasion to leverage information supplied by a partner in its user interface, to describe a service, link to information about the service, render a logo, etc. Care should be taken to ensure that the source of this information is trustworthy because of the ease with which an attacker could take advantage.

In particular this use case demonstrates the risk of remotely acquiring metadata directly from a partner unless it's signed, and we discuss above the scalability limitations of directly establishing trust in a signing key for every individual system. Federations offload the cost of establishing that trust and of vetting the safety of data that might be incorporated into system operations, and of responding to cases in which unsafe data might appear.

Technical Interoperability

SAML is a wide-ranging specification covering many potential usage scenarios. By itself the SAML specification does not guarantee functional interoperability. In Software as a Service scenarios, it is crucial to agree on the particular profile of SAML being supported.

Both Campus and Vendor: DO conform to the SAML2 Web Browser SSO Interop Profile.

Ensure that your respective SAML endpoints, IdP and SP, operate in conformance with the SAML2 Web Browser SSO Interop Profile (http://saml2int.org/profile/current).

Both Campus and Vendor: DO establish a single issuer name and keypair for a given IdP or SP.

Deployment of an IdP or SP becomes much more complex and troublesome if it tries to accommodate custom requirements for identifying itself in assertions or using separate signing keys for different partners. This is a sign of a broken set of assumptions somewhere and should be resisted other than on a temporary basis.

Related to this, don't perform upgrades or make software changes by creating a second IdP or SP with its own name or key, unless you really want to permanently change to that name or key, which should be a very rare occurrence. There are always better testing and migration strategies.

Both Campus and Vendor: DON'T change signing or encryption keys unnecessarily.

The conventional approach to public key certificates is for expiration on the order of a year or two, and people often generate new keys when "renewing" certificates for their web servers. Do not take this approach with the keys used by an IdP or SP (other than for user-facing TLS/SSL ports). Changing a key is a very expensive process to undertake and often causes substantial disruption because of the lack of support for SAML metadata, or indeed any key management strategy, by many software products and many organizations.

There are no known attacks against the RSA algorithm that involve possession of large amounts of signed data. Therefore, there is no "appropriate" lifetime for a key related to anything but the cost of a brute force attack, and 2048-bit keys are not at this time considered vulnerable to such attacks. As a result, there are no technical reasons to change a key of sufficient size that has not been compromised, or is at risk of compromise. Of course, this can change due to advances in computing power or discoveries, but those events don't occur at predictable intervals.

Simply put, realize that the historical notion that keys should be changed regularly simply because of the slightly increasing possibility over time that an exposure may have occurred is based on a model in which changing a key is essentially a low cost event, and that is not the case in federation scenarios.

Both Campus and Vendor: DON'T be afraid of self-signed certificates.

Using a long-lived, self-signed certificate registered in metadata issued by a third party is the best choice for an IdP or SP. Resist well-intentioned but misinformed advice that suggests you will be "safer" using a commercial certificate.

There are no commercial sources of certificates that are relevant for SAML deployment. Applying the "logic" of the commercial market for web server certificates to other use cases is not a wise choice because the process of obtaining such a certificate does not involve verification of responsibility for a SAML IdP or SP, nor does the content of the certificate reflect its purpose. In other words it doesn't mean what people want it to mean. Furthermore, there are very few SAML implementations that don't at heart require knowledge of signing or encryption keys in advance, which defeats the purpose of relying on certificates from a third party.

Operational Agility

Another "cost" of operating distributed systems is that the degree to which they are coupled has a large effect on the cost of operating them. In other words, the more information you have to exchange and depend on staying the same, the more likely a change will take more time and effort. This also applies as you scale out; the cost of a change will increase with the number of affected systems unless you have a scaleable way of communicating them.

Securing cloud services typically will involve exchanging and maintaining a number of pieces of information, including public or shared keys/passwords, service names and locations, and configuration options. In the case of SAML, to use one example, changing any of the following are usually relevant enough to cause disruption without a process for communicating and applying the change (most other protocols similar to SAML will have comparable requirements):

- public keys
- cryptographic algorithms used
- SAML entityID (the "issuer name" associated with messages)
- locations/URLs of various system services
- SAML binding and profile options used, in certain cases

When we speak of operational agility, we refer to an understanding of what needs to be stable, what can be more easily changed, and what techniques can be used to facilitate changes when they do occur.

As an example of how not to do this (or maybe more to the point what you have to do when software is not designed properly), it's very common for major Fortune 500 companies in the federation business to announce changes to a key or a URL in customer bulletins and on Twitter, and set a flag day for every affected system to implement a change. Quite obviously, that approach fails completely when you are dealing with peers to whom you can't simply dictate terms, and even when you can, it essentially guarantees outages and bad feelings.

Both Campus and Vendor: DO make careful choices in the beginning.

The most important strategy is to avoid changing anything you don't have to. Think long and hard about the choices you make up front, and do your research to understand best practices and to understand when those practices may not work because of specific partners you may have to deal with. Avoid the "I'm just testing" trap and quick decisions that turn into production decisions, particularly if your organization doesn't do a good job of separating pilot from production.

For SAML deployments, some examples regarding best practice in this space may be found at https://spaces.internet2.edu/display/InCFederation/Recommended+Practices.

Both Campus and Vendor: DO pick good names and identifiers for services whenever possible.

Choose well-considered service identifiers and locations, and follow any naming standards in your organization for long-term, unchanging systems exposed to end users. Don't fall back on redirects to avoid making long-term choices, as not all uses will be capable of handling them easily if at all.

When it comes to URI-based names, like SAML entityIDs, avoid the use of URNs in new deployments, and use https:// URLs without non-default ports. Where possible, pick names that could be made to resolve into useful resources even if they don't initially (i.e., don't use a URL from which it would be difficult to completely control the response).

Above all, understand the stability of the system name/identifier is more important than its perfection. Don't change a name because it's convenient or desirable, only do so when it's unavoidable or when the original name was simply invalid (not a URL, not a valid name you control, etc.). Use the same care here as you would in changing the end-users' launch URL for your service.

Both Campus and Vendor: DO invest in configuration management.

Because of the range of options and features available with most federation technologies, good configuration management is really a must. Given the state of many software products, outages are essentially unavoidable in many cases, and having a detailed record of what has changed and when will be essential to fix problems and provide explanations to angry managers.

Both Campus and Vendor: DO use self-signed certificates on non-user-facing endpoints.

Both for SAML-based services and any other application in which something other than a user's client is involved, stick with self-signed, long-lived certificates. Traditional PKI models are both unpredictably implemented in software and insufficiently well-thought out in terms of application requirements. Worst of all, they often subvert themselves through hybrid, Frankenstein-like implementations that perform both certificate validation **and** certificate comparison, a scenario in which a certificate likely has to change annually or bi-annually, but also needs to be pre-installed at the right time to all partners. Essentially you get all of the bad of both approaches and none of the good.

Key management and rollover is among the hardest problems to address, and is probably the single change to avoid at all costs. Using self-signed certificates is the most secure and simple way of provisioning endpoint security because it's both stable and predictable, and it limits unnecessary changes imposed by external factors, such as compromise of certificate authority (CA) certificates.

Where possible, encourage your solution providers to adopt software profiles that ignore certificate content and do not enforce arbitrary expiration based on certificate lifetimes. This is important because even long-lived certificates eventually expire, and they obviously aren't meant to expire at any specific time imposed twenty or thirty years before.

Both Campus and Vendor: DO register SAML metadata with an identity federation.

This is a repeated recommendation, because it serves multiple goals. Registering metadata provides a set of configuration material that many, though obviously not all, partners can provision their systems with. Because there is an established set of practices around the need to refresh it daily, a large number of partners can be counted on to do so, and to therefore "see" changes made to it within 24 hours of a change.

Supplanted by appropriate features in your own software, you can safely accommodate many kinds of operational changes without service disruption, including key rollover, endpoint and URL changes, and even naming changes in some cases. Many SAML behavioral options can also be signaled using metadata, and at least in the case of Shibboleth, substantial functionality is still being added in new releases to accommodate a wider range of operational change requirements.

Both Campus and Vendor: DO understand your partners' limitations.

It's very important to have an understanding of how partners working with your system deal with the issues of trust management and operational agility discussed in this and the previous sections. Setting aside the systems that handle federation metadata according to current best practice, the focus becomes supporting the rest of the systems. All of them are going to work differently (by definition, since they eschewed supporting the standard way of doing things), though they will likely be group-able to some degree through commonality of software.

Reinforcing a point in the previous section about process, establish a plan up front for how to deal with expected or unexpected changes with each partner that has no automatic way of dealing with them. Establishing the right technical contact(s) is of course essential, as is understanding that partner's change management procedures and whether they create barriers to following your own. Create a game plan for how changes have to happen, and a clear timeline for how long they would take.

For campuses in particular, when possible, push heavily to rely on metadata as your interface to their processes. Otherwise their manual work and system deficiencies become your burden.

Sharing the Burden

Federated Incident Response

The following material is based on guidelines from REFEDS and the work on Security Incident Response Trust Framework for Federated Identity (SIRTFI).

Both Campus and Vendor: DO publish federated incident response contact information in the metadata.

Federated incident reporting and response is difficult without a standard place to find security contacts. REFEDS, provides a schema to include Security Contact information in the entity’s metadata (along side administrative and technical contact details). Make sure to publish a security contact for your organization. Use the REFEDS Security Incident Response Trust Framework for Federated Identity (SIRTFI) guide for help when choosing your security contact Choosing a Sirtfi Contact.

Campus: DO update your Participant Operational Practices document annually, creating one now if your organization does not have one.

Processes change, and it’s easy to forget to update documentation about those processes. The participant operational practices is important as it lets others understand your processes when federating. An annual audit of this documentation is the best way to remember to keep it up-to-date.

Campus: DO include the URL of your institution's privacy statement in the metadata.

Many federations provide a method for publishing a URL to your organization's privacy statement. Other federations allow for similar inclusion in metadata. Make sure to publish an up-to-date URL for your institution’s privacy policy in a place where others can find it even if it’s not in metadata.

Campus: DO implement a log retention policy.

This may seem like common sense, but it’s a practice that’s commonly overlooked. Keeping too few logs can become a problem when investigating an issue such as a security incident. Keeping too many logs or never rotating and deleting old logs can become a burden for storage and log parsing. Depending on your local requirements you may be required by law to set a limit for data retention, maintaining data indefinitely is often illegal. Make a policy and implement it. Ensure your local policy aligns with any institutional record retention policies.

Campus: DO document and advertise your procedure for responding to a federated security incident.

Just like publishing your security contact, publishing your incident response procedure helps the entire process of incident response work better. It helps those who you've federated with know what you expect of them and what they can expect in return. The documentation should include your processes for responding to a security incident related to your Identity Provider and responding to an incident with a Service Provider that you have federated with. The Security Incident Response Trust Framework for Federated Identity (SIRTFI) contains best practices for security, including incident response. Refer to this framework as a good place to start when defining your procedure.

Campus: CONSIDER whether or not your policy should be the same for a local (campus) Service Provider vs a Cloud Service.

Though cloud and local services can be very different in terms of their interaction with campus systems and the communications you have with their service managers, there can be many similarities, too. Responses to a security incident may be very similar since they both involve assessing the damage done to all services and taking corrective actions. At the least, understand how your local incident response procedure and your cloud incident response procedure mesh and tie them together where possible.

Campus: DO be aware that failing to represent user identities to the degree of authority and accuracy specified in your Participant Operating Practices may be considered a security incident.

Your identity provider is a trusted source by your service providers. The trust model breaks down if you're asserting identities to service providers that are less accurate than you indicate in your participant operational practices. This clearly is a security incident regardless of it's cause. For example, if a user is able to impersonate another user when accessing a cloud service, the campus should inform the cloud provider as they may need to take corrective action within their application.

Vendor: DO actively respond to security incidents reported by the identity provider.

When you detect a security incident at your site, reach out to notify campuses. Take appropriate action on reports of security incidents that come from campuses to you.

Both Campus and Vendor: **CONSIDER publishing the SIRTFI Assurance Profile in your metadata.**

If your organization is fulfilling the best practices in logging, operational security, participant responsibilities and incident response defined in the SIRTFI, consider indicating this in your metadata. A guide to SIRTFI is provided by REFEDS. Many of the best practices defined in this cookbook echo the SIRTFI framework and, by indicating your compliance, you show that you can be trusted to cooperate effectively in incident response.

Behavioral Trust

The term Behavioral Trust refers to the actions of users, service administrators and other support staff that impact a federated trust model. Not all aspects of trust can be managed by technology. To establish trust, the cloud service and the identity provider must establish and share clear guidelines, work together to interpret data similarly, and follow through on promises made in policies. This section will identify some areas where campuses and vendors need to concentrate to maintain a trust model that works in conjunction with the more technical controls.

Both Campus and Vendor: DO follow through a procedure for federated incident response.

The importance of creating and publishing an incident response policy has already been discussed. That effort is useless, though, if all of the key players aren't putting that policy to work. That may sound obvious, but it's far too easy when a user's account is compromised to work with the user and the identity provider to clean up the situation and think the job is done. When the user's access extends into resources in the cloud, the impact of the compromised account can be much more far-reaching. The only way to insure that all bases have been covered is to notify all services that the user has or might have access to of the compromise, and have those services acknowledge their receipt of, and response to, the report.

Identity providers should make sure that the staff who respond to account compromises are familiar with the federated incident response policy. Staff should have access to the security contacts for all federated services and should understand what those services expect when reporting a security incident. A small amount of training and planning will help to guarantee that security incidents, local or in the cloud, don't go unreported and cause larger problems down the road. Vendors should realize the significance of a compromised user account on their own service might have larger impact if the user is logging in through federated channels and have procedures for reporting the incident to the user's home organization. Due to the degree of interconnection of federations one compromise can have far reaching consequences; it is not sufficient to protect your own services and is essential that all affected participants are aware of the impact. Vendors and campuses that are SIRTFI compliant already have a method to contact each other, and have acknowledged the importance of collaboration during federated incident response.

Both Campus and Vendor: DO work with federated partners to understand how data is being interpreted.

Some services will use attributes such as *eduPersonScopedAffiliation* or *eduPersonEntitlement* to identify a user's eligibility to use the service. Use of this data requires a shared understanding of the meaning of the attribute values. For instance, an affiliation value of "student" asserted by a university might be used to indicate more than traditional students, such as those in an outreach or cooperative education program. If those students log into a cloud service only intended for traditional undergraduate and graduate students, cloud services might incorrectly grant them access. In this situation, the university and cloud service need to understand how access is granted and what various values of attributes really mean to assess their suitability in enforcing policy.

Avoid making assumptions: Spell out all of the details up front as well as whenever changes are made to identity systems or user authorization rules. A little careful planning and consideration and good communications between identity providers and cloud vendors can go a long way.

User Support

Both Campus and Vendor: DO agree on a clear delegation and division of support responsibilities.

Every campus and vendor will have slightly different approaches to user support, with a variety of expectations regarding communication style, response time, and so forth. With shared support for a service, these differences will likely lead to inefficiencies, misunderstandings, and user frustration unless support responsibilities are clearly assigned from the start the deployment. In addition to agreeing on how the support load will be shared and whether certain issues will be handled by the campus or the vendor, it's important for both parties to be on the same page with regard to common service policy issues. Otherwise vendor support may inadvertently lead users to run afoul of their school's service guidelines.

It's likely that the school and vendor both have multiple support tiers, and that certain types of support will need to be handled by a specific support tier in one or the other organization. For example, it may be that certain technical issues can only be addressed by upper-tier vendor support, but that special provisioning or privilege escalation decisions must be made by upper-tier school support. In that case, a single support issue may start in tier-1 at the school help desk, be escalated to internal tier-2 for a service management level decision, then passed to the vendor for some kind of technical support. If escalation paths between the various levels of school and vendor support are not clearly established, support may be delayed or tickets my be mistakenly dropped.

Also keep in mind that it's important for every tier in both organizations to understand which support body is best equipped to handle common support tasks. For example, a quota increase request received by tier-1 on either the vendor or school side may be best escalated to school tier-2, while a certain type of software support request may bypass school tier-2 completely, and go directly to upper-tier vendor support. If each tier understands the common escalation paths, support is less likely to be delayed by unnecessary ticket juggling.

Campus: CONSIDER maintaining a single point of contact for users.

Even in cases where vendors are well versed on school policy and willing to provide tier-1 support, you may want to consider using your local help desk as a single point of contact for ALL initial support requests. This will give you improved visibility into user activity and common service problems, and will also allow you to provide a more uniform and better contextualized support experience for your users. Of course this approach also comes with the burden of greater organizational expense, and shedding support work is a significant aspect of the cloud service value proposition for many organizations.

User Experience

It is common for cloud service adoption within organizations to be driver by cost savings concerns, and cloud service deployment teams are often very small compared to in-house service teams. In the rush to work through contractual details, purchasing headaches, and technical/logistical concerns like provisioning and federation, it can be easy to forget about the end user. However, the experience your users have around documentation, security, error handling, and support will heavily shape the perceived success (or failure) of the new service. This section touches on a few user experience concerns that the authors have found to be common across many cloud services.

Error Handling

Campus: DO provide an error web page where an SP can send users.

There are many reasons why a user, after successfully authenticating and returning to a service, might encounter an error. While many of those reasons might be due to technical failures at an SP, that's definitely not always the case. Often the data supplied by the IdP may not include necessary attribute values. In more complex cases, an SP might request particular behavior that an IdP does not support, resulting in a protocol error (in a SAML response for example).

If the SP can't locally react to and address the problem, or if the user's home organization has agreed to provide first tier support for a federated application, you'll want to have control over the information that the user sees when something goes wrong after authentication. The most flexible way to do this is by creating an error page that contains tips for users such as support contacts or links to the organization's help desk.

While this page may need to be service-specific in some cases, more often it can be a general resource for all SPs who might need it. The location of such a page should be published via SAML metadata so that SPs can rely on it with confidence that it's a legitimate resource.

Vendor: DO display user-friendly error messages.

It is good practice to always make your error messages user friendly: What happened, why it happened, and what the user might do to remedy the situation. This is especially true for federated logins. The most common causes of these errors are the user not being authorized to access a service and the identity provider not releasing required attributes. In either case, don't leave the user at a dead end. If they're not authorized, explain why. If the error is the fault of the identity provider, direct the user to a help contact from their home organization.

Vendor: DO make use of IDP error URLs in the metadata.

If your federation provides a place in metadata for an error URL, use it. It can be a very appropriate place to send a user if their identity provider isn't sending you all of the required attributes. A good flow might be an error page explaining that the user's organization isn't releasing data needed to access the service, then a link to the organization's IDP error URL where the user might contact the right staff to resolve the problem.

Interacting with the User

Vendor: DO use appropriate attributes for friendly names.

Many federated identifiers and other attributes like email address are not well suited for use as friendly names for users. They almost always end with "@institution.edu", and even the left-hand side of identifier may appear to be randomly assigned, particularly in the case of *eduPersonTargetedID*. Use other attributes, such as displayName, for this purpose.

Vendor: DON'T over-use forced re-authentication.

SAML-based authentication allows the service provider to request forced re-authentication from a user's identity provider. This means that, even if the user has already logged into another service through their identity provider, they'll have to re-enter their credentials. This can be useful for service providers that contain more sensitive resources as it provides some assurance that the user sitting at the keyboard is the same user who logged into earlier services. If used by too many service providers, though, the benefits of single sign-on are greatly reduced. If every service required forced re-authentication, there would be no visible single sign-on to the user. Before your service decides to request forced re-authentication, carefully consider if the risk of not re-authenticating outweighs the convenience of single sign-on.

Logout

The web has no intrinsic concept of user logout because it was never designed to support interactive applications beyond the use of simple forms. Authentication, when it was added to HTTP, was integrated poorly, and did not include the concept of a session. Without a session, logout has no real definition. As a result, there is no specific technical definition that aligns with any particular set of user expectations once you move beyond simple cases.

From that simpler perspective, logging out of an application tends to be simple: destroy a cookie, clear related application state, and you're done. This is almost exclusively done via explicit user interaction, but very occasionally an application may expose some kind of administrative control over active sessions.

Unfortunately the concept of Single Sign-On ruins this simple picture because it involves sessions (generally managed via cookies) that exist in several, perhaps many, hosts/domains at the same time. The web's security model makes it impossible by design to collectively manage these cookies centrally, resulting a very complex and ultimately impractical problem to solve without help from the client. The browser makers collectively refuse to address this problem, and in fact have eliminated even closing the browser as a practical option, which leaves us with brittle and confusing results for users.

The following include some mitigations, but ultimately the client must be changed to address this problem.

Campus: DO focus on desktop and mobile security.

The problem logout is trying to solve is not really about logout, but about device security (with one exception, noted below). For desktops, the solution is not logout but locking unattended workstations. For mobile phones and tablets, appropriate device management and PIN access are the proper approach, particularly because many mobile applications encourage long-lived (in some cases permanent) sessions.

What this leaves is the shared device, principally computing labs and kiosks. This is where lack of logout can be a serious problem. The best solution, particularly for labs, is to enforce authenticated access to the desktop. Users can log out via the desktop, and simple policies can be used to enforce profile separation or actual reset of browser state between users.

The exception that is difficult to address out of band is the kiosk model.

Campus: DO disable Single Sign-On or restrict the scope of network access for kiosks.

A partial approach to the kiosk problem is to bypass SSO for particular address ranges, or to provide a user-controllable option to do so. Though this does not really eliminate the risks associated with web application sessions, it eliminates one key risk by preventing automatic access to other applications. Another way to approach this is to prevent access to arbitrary network services, and limit kiosk access to only the specific applications for which it's designed.

Campus: DO support redirect-based logout.

Actual standards-based logout using SAML is very rare among vendors, but most cloud services do allow for a logout URL to support redirecting the browser back to a customer's site after an application logout takes place. It's a good idea to establish a supportable, stable location for local control over user feedback and possibly to integrate the experience with an IdP in some way.

It is **not** advisable to point a service at an IdP-supported feature or "hook" directly via this mechanism because a simple redirect from an application is by definition proprietary. It's better to relay through an externally maintained script to forward the proprietary logout into the IdP because this maintains local flexibility without losing any function. Note that often applications will allow a custom logout URL to contain arbitrary parameters, so it may be possible to tailor feedback based on the redirecting application.

Most IdP software, even if SAML logout is supported, also support a redirect-based logout that, if nothing else, will remove a session such that any subsequent requests for authentication will re-authenticate the user. This limits the damage of a session left unattended such that only applications already accessed will be at risk.

This kind of logout approach is simple to integrate with the proprietary logout capabilities supported by most services, and can usually be implemented easily even without explicit support in an IdP when the underlying session is based on cookies.

Campus: CONSIDER the issues associated with standards-based logout.

SAML, to use one example, includes a protocol for a single logout across all services, brokered by the IdP. While this protocol is well-intentioned, it is limited by both its own design and modern web considerations such as the blocking of third party cookies for privacy protection and the reality of a mixed environment of older and newer SAML implementations, some of which do not support logout at all. It also fails to take into account the reality that many applications are integrated with SAML in a way that precludes logout. Moreover, some users don't even expect an individual application logout to affect other services.

In practice, almost every attempt to use such a protocol will have an incomplete or "partial" logout result. This is not obviously explainable to users, nor is it actionable, since by definition it means the user can't make logout work as intended. Before supporting such a service, consider how you will explain to users that they remain logged into some applications and what you expect them to do about it.

Vendor: DO more than just destroy your local session, as appropriate.

For SP operators, it’s important to be aware that, if a user authenticated through their home organization’s identity provider that supports single sign-on, their session may still be valid there. If they log out of your service and you do nothing more than destroy their session on your side, they won’t be prompted to re-authenticate if they come right back in. The existing IDP session will be used and, to the user, it will appear they never logged out. If the IDP supports it, destroying the IDP session should also be part of a standard logout. Such a logout URL can be consumed from metadata. At the least, provide a mechanism for the IDP operator to configure a logout URL manually where users can be sent after logging out of your service. Not all IDPs will support this feature, unfortunately. In such cases, it’s the services responsibility to do something intelligent when there is no IDP logout URL. This may be as simple as telling the user that they’ll need to close their browser to completely log out.

Vendor: CONSIDER supporting logout requests from IDPs.

It’s possible that ambitious IDP operators from organizations you serve will want to set up single logout. When a user logs out of one service and is sent to the IDP’s logout URL, the IDP will try to destroy sessions from other services that the user accessed during that IDP session. This method is far from perfect, but for it to even be remotely successful, SPs must be able to support these single logout requests from an IDP. In reality, few IDPs use single logout because of technical implications and the over-all user education associated with it. Supporting it on your service is a relatively easy task, though, if you already support logout locally, and it adds value for those few IDPs that want to utilize it.

Data Stewardship

Responsibility for the information being shared between a campus and a service provider continues past the initial authentication and (if applicable) authorization transaction. While specific legal and policy guidelines may vary between countries or regions, types of activities (e.g., banking, health), or even types of data (e.g., government funded research, personal information), some basic best practices often apply.

Campus: DO protect your identity data.

After you release user attributes to a service, that data is no longer under your control. The service may not own the data, but they’re responsible for applying needed safeguards for how the data is used, stored, and accessed. Before an identity provider releases anything to a service, they should know about the service’s practices and security measures. In many cases, a federation’s operating agreement may address this issue. If not, or if the service isn’t a member of a federation, make sure to get satisfactory, up front answers on how your identity data will be stored and used.

Campus: DO articulate a strategy for dealing with the difference between institutional and personal user data.

Most cloud services house some kind of user data, and of course this is particularly true for applications that provide collaboration and file storage service. Many higher ed users will use cloud service accounts to store a mix of personal and institutional data. For example, a faculty member may have research work, family photographs, and departmental policy documents, all under a single account. If the user were to leave the university, the family photos and (probably) the research files would rightfully leave with the user, but the policy files should stay with the user's department. Untangling these document collections can be very time consuming, and lead to both institutional data loss and serious de-provisioning complexities.

Before launching a new cloud service, take some time to think through how users will store data on the service, and whether or not the service is prone to encourage the mixing of personal and institutional data. If so, do develop a strategy for dealing with this problem before the service goes live to your campus. In some cases providing a best practice guide may be the best solution. For other services you may want to consider leveraging a groups service or creating shared or non-personal service accounts that can be used to store data that should stay with the institution when an individual leaves their job.

Campus: DO publish a list of approved data types for your cloud services.

When a new cloud service goes live, the first question that many campus users will ask is, "What kind of data can I store on this service?" Consider publishing an approved data types chart that lists the institution's cloud services plus the approved data types for each service vendor. Note that local or regional laws may apply either based on particular kinds of activity (e.g., FERPA, HIPPA and some data localisation rules in Europe, e.g., on financial and medical), or on particular kinds of data (e.g., personal data in Europe). These laws may touch on a variety of aspects to cloud services, including their physical location, security, data retention policies, applicable jurisdiction, data portability, etc. You data classification policy should include enough categories to be useful, but not some many that the chart becomes cumbersome or confusing to your users.

Campus: DO manage data sensitivity and stewardship concerns from day one, and don't underestimate the potential impact to project scope.

This is another "look before you leap" recommendation. It's important to identify the data stewardship and compliance stakeholders on your campus at the beginning of your deployment project, bring them into your project team, and budget time for identifying approved data types as well as areas of concern that may need to be mitigated.

Appendix: US Policy and Compliance

This section covers topics that are of special interest to Higher Education, and it is assumed that the Identity Provider is being operated by a school that must comply with The Family Educational Rights and Privacy Act (FERPA).

FERPA

Although Identity Providers at Higher Education institutions are likely familiar with the following information, this section contains guidance specific to cloud service integration. It may also be of special interest to cloud service providers who do not have significant experience working in the Higher Education sector and who may receive attributes from Identity Providers that is considered to be FERPA protected by the school releasing the data. That this is not to be considered legal advice, but rather a brief and general overview of FERPA.

Overview

FERPA is a federal law intended to protect the privacy of student education records. All schools that receive funds under an applicable US Department of Education program are required to comply with the law (http://www.ed.gov/policy/gen/guid/fpco/ferpa/index.html)

FERPA classifies information into two categories:

1. Educational Records
2. Directory Information

FERPA requires institutions that make directory information publicly available, such as in an online directory or institutional white pages, to publicly define the information classified as such, and that a reasonable amount of time be given after that notice for the owner of the data to request that it not be released.

When personal information is transferred to a third party, such as a Service Provider, it must be on the condition that the information will not be disclosed to any other party without written consent of the individual considered by FERPA to be the owner of the record: the parent or student, depending on the student's age (http://www.law.cornell.edu/uscode/text/20/1232g).

The language of FERPA will be interpreted by each schools legal council and so attributes classified as Directory Information and mechanisms for compliance will differ from school to school.

Campus: DO work with data owners to consider a default release policy.

Much of the information of interest to SPs is the kind of information that many institutions historically have classified as Directory Information under FERPA. In the case of public institutions, much of this data is often considered public record for employees (allowing that this varies by state). While releasing this data as part of authentication is not the same as hosting it in a directory, it is also not materially different, and arguably constitutes a more controlled release of data at the discretion of the data subject.

InCommon members should consider the adoption of policies to release Directory Information attributes to federation services on at least some ad hoc basis. The Research and Scholarship program is an alternative to federation-wide policies and may also be of interest. Work with data owners such as the HR department and Registrar (or others as appropriate) to establish consensus about the appropriate policy to use for default release.

Recognize that a default-deny policy will act to inherently limit the value provided by an IdP, though it will likely decrease ongoing support costs. Make these decisions deliberately.

Campus: CONSIDER a user consent mechanism for attribute release.

Consider a consent-based approach to provide user control of data release. Many federation products include functionality, or support add-ons, to support user consent to the release of data to services. A consent process may be a good way of expanding the reach of an IdP by enabling users to make the decision about whether to release data, in lieu of requiring contracts for every individual service a user might access. Consent systems also provide an auditable record of approvals, and can help address a requirement for acceptance of usage terms.

Note that consent approaches have limits. They should never be used if a student or employees failure to consent would bar them from a service they are required to use, nor do they work well for attributes beyond a user's capability to understand.

A typical consent UI should be able to identify the relevant service, and attributes, concisely. InCommon membership, and the use of SAML metadata, provides a good source of reliable information to use in driving such a UI.

Campus: DO establish procedures for managing the release of restricted data.

Some cloud services will inevitably need access to restricted or sensitive data, so be prepared for this by working with policy makers to establish a process for all involved to follow. Requiring participation in a federation that follows current best practice in the identity federation space may facilitate the contract review process by covering some of the basic expectations regarding use of data and disclosure to third parties.

Campus: CONSIDER that cloud services may solicit additional identity data directly from users.

Even if only a small amount of data is provided via an IdP to a cloud service (even as little as an opaque identifier like the *eduPersonTargetedID* attribute), users may be prompted to fill in a user profile within the service that includes additional personal information. FERPA may be implicated if the use of a service is mandatory and the additional data is required.

HIPAA

This section covers the Health Insurance Portability and Accountability Act (HIPAA), which is a law intended to protect the privacy of individual's healthcare information while allowing for the flow of health information necessary to provide healthcare services and to protect the nations overall public health. Like the previous section of FERPA this is not legal advice, but provides general guidance to be considered by campuses and cloud service providers that offer services that may be used to store HIPAA protected data.

Campus: DO require that cloud service providers sign a Business Associates Agreement (BAA) if it is intended to be used with HIPAA protected data.

HIPAA's provisions for Business Associates Agreement is particularly important in regards to the use of cloud services that a campus intends to use to store protected data. A "Business Associate" is generally defined as a entity "... that performs certain functions or activities on behalf of, or provides certain services to, a covered entity that involve the use or disclosure of individually identifiable health information." (http://www.hhs.gov/ocr/privacy/hipaa/understanding/summary/)

Sample language is provided by the US Department of Health and Human Services: http://www.hhs.gov/ocr/privacy/hipaa/understanding/coverdentities/contractprov.html

Campus: CONSIDER requiring all data covered by a BAA be contained within the borders of the United States.

Discussion

It is possible that a vendor of cloud applications will store your institutional data in a data center that is located outside of the United States. Under such circumstances the data may be accessible to the foreign government of the country where the data is physically located.

Campus: CONSIDER requiring that cloud services who store communications and data sign a BAA.

Discussion

Even if institutional policy forbids sending or storing HIPAA protected in a cloud service, such a policy may not be sufficient to prevent the use of a cloud service for such purposes. Email and file storage services may be of particular concern. If the cloud provider is able to sign a BAA that could mitigate any exposure by the institution's members to that particular vendor.

Appendix: Identifier Properties

Identifiers have a number of characteristics that help to determine appropriate usage and it's important to express application requirements in terms of properties that can be mapped back to those satisfied by particular identifiers to choose the best one for the job.

Persistence

Persistence is a measure of the length of time during which an identifier can be reliably associated with a particular subject. A very short-term identifier might be associated with an application session. A permanent identifier is associated with its entry for its lifetime (which is not necessarily "forever", so permanence is just a relative notion). Typically we refer to an identifier as having persistence if it is stable over a relatively lengthy period, usually measured in at least months.

Reassignment

Many identifiers do **not** specifically guarantee that a given value will always refer to a single subject forever. Reassignment means the association of an identifier value to one subject, and then assigning the same value to a different subject at some point in the (possibly distant) future. Some applications dictate a strict no-reassignment policy, but in practice many more tolerate or ignore the issue by hoping it just doesn't come up.

Privacy

Some identifiers are designed to preserve a subject's privacy and inhibit the ability of unrelated applications from correlating activity by comparing values they receive. Such identifiers are therefore required by design to be opaque, and to have no particular relationship to a subject's legal identity or other identifiers. Note that this definition still permits sharing/commonality of the identifier among multiple applications if they are deemed to be equivalent to a single application for privacy purposes.

Human Palatability

An identifier that is human-palatable is intended to be rememberable and reproducible by typical human users, in contrast to identifiers that are, for example, randomly generated sequences of bits. There is a natural tension between palatability and both privacy and non-reassignment and they are often in opposition. The world does not have a popular solution to all three problems at once today, which feeds into the oft-noted recommendation that many applications really need to use multiple identifiers for different purposes.

Uniqueness

All identifiers must have some degree of uniqueness, within a particular "namespace" in which the identifiers are being created and managed. Sometimes this namespace is explicitly made part of the identifier (as in the case of a "scoped" identifier, see below), in which case the identifier is globally unique. In other cases, the namespace may be implicit, in which case the identifier may not stand alone without the namespace being articulated and stored in some form. This becomes particularly relevant when applications are truly federated (supporting multiple Identity Providers accessing the same data), or otherwise store identifiers in a common way.

The two most common forms of explicit namespace expression today in SAML-based systems are "scope" and using the SAML NameQualifier and SPNameQualifier XML attributes.

Scope is typically but not always derived from the hierarchical DNS, and is usually a DNS subdomain associated with the organization or system that is in control of an identifier. Scoped identifiers and other scoped attributes are made up of a delimited string, usually in the email-like form of "value@scope". Adding scope can turn a locally unique identifier into a globally unique one.

The SAML notion of name qualifiers is more complex than scope, but is directly tied to the naming of SAML-based Identity and Service Providers so that there is no ambiguity about how an identifier might relate to the systems that are exchanging it. Name qualifiers can be anything in theory, but in practice when used are in the form of URIs. There is no single way to take an identifier, and one or both name qualifiers and collectively store it as one piece of data; it is a local matter for software to deconstruct the XML into some other form. This makes the use of name qualifiers suitable only for cases in which an identifier is stored, but not exchanged, in non-XML form, or even displayed. Scoped identifiers are more compact and simpler to use for many purposes, but they require an additional layer of naming policy to govern how a particular scope relates to the parties exchanging it (i.e. who can assert what).

There are other identifiers, used less in federated systems, but very often in enterprises, that are explicitly namespaced in other ways. Distinguished names, if used correctly, are globally unique, and Kerberos principal names are qualified by realms that essentially are like the scopes described above.

Appendix: Case Studies

Lynda.com: InCommon membership, eduPersonEntitlement

This case study highlights that partnership between institutions and vendors that aligns with federated identity best practices ultimately provides value to both the campuses and the vendor of the cloud application.

Lynda.com is a cloud application that provides training videos for a wide variety of topics with particular emphasis on technology and software applications. As such, there is a single instance of the application rather than a dedicated instance per client.

In a survey sent to CIC Universities in the Spring of 2013 seven of the (then thirteen) members responded stating they had licensed Lynda.com for use on their campuses.

When Pennsylvania State University (PSU) negotiated their contract with Lynda.com, they made joining the InCommon federation and support of SAML based federated login, a requirement. PSU's technical staff noted that working with Lynda.com to support this new functionality required more staff time, but that effort was seen as an investment because it allowed them to avoid implementing a non-standard, 'one off' integration.

After Lynda.com became an InCommon member, Michigan State University (MSU) licensed Lynda.com and requested that the vendor support authorization based on *eduPersonEntitlement*. This was necessary because they did not license Lynda.com for the entire campus population. Support for *eduPersonEntitlement* allowed the campus to determine who was eligible based on the license agreement, and to assert that to the cloud application using an agreed upon value sent in the attribute when the user was authenticated by the campus.

The University of Iowa (UI) later licensed Lynda.com for campus use with the exclusion of staff who worked for the University of Iowa Hospitals and Clinics. Because of the work done by their peer institutions, the integration with the cloud application was a straightforward process that took a minimal amount of staff time. For instance, their IT staff did not have to integrate their Identity Provider with Lynda.com; instead the pre-existing integration with InCommon was used. The bulk of the work was updating their campus Identity and Access Management (IAM) system to include the new service eligibility for the licensed population. After that was completed, configuring the campus Identity Provider to release *eduPersonEntitlement* to the cloud application was a trivial process.

The value of the work done by Pennsylvania State University, Michigan State University, and Lynda.com did not just benefit The University of Iowa, but provides value to *any* InCommon member that subsequently licenses the cloud application. As well, Lynda.com benefited. They too are able to leverage their existing integration with InCommon, rather than configuring a new integration for each customer, which should reduce operating costs and the time needed to on-board new customers. Further value is added by offering a product that is easy for InCommon members to adopt.

Thus, we see that the end result was positive for all parties: the barriers for adoption of the cloud service were successfully lowered thanks to the membership in the InCommon federation, the use of the eduPersonEntitlement attribute for authorization, and the initial staff time provided by the vendor and the first schools that adopted the service to ensure best practices were followed.