Provisioning Workstream Fit Gap Review

Overview

The Provisioning subcommittee has defined its scope as the means and mechanisms by which IAM data is kept consistent across all consumers and providers of identity data. The subcommittee largely sees this scope in terms of Enterprise Application Integration and proposes to leverage well establish patterns and open source components in this area. This essentially covers the propagate and maintain phases of the IAM data lifecycle, and includes data flows "SOR to/from Registry" and "Registry to target systems (e.g. directories, LMS, Google Apps, etc)".

Deliverables

The Provisioning subcommittee perceives the O4 effort overall as partially a development effort, partially a community definition and direction effort, and partially a surveying and recommendation effort. With that in mind they expect to organize their work into three sets of deliverables:

1. Recipes for implementing provisioning in HE
2. Toolkit of existing and to be built open source tools
3. Collection of implementation stories based on 1 & 2

The overall approach is to iterate on all three deliverables while focusing on a small number of implementation stories in each phase. The implementation stories are to be derived from actual production deployments based on 1 & 2 at partner institutions.

Identified Implementation Stories

- CMU WorkdayHR to PersonRegistry
- UWisc OracleHCM to Person Registry
- Grouper to apps and directories

Future implementation stories could include:

- non-HR SOR to Registry ("guest" systems)
- provisioning to cloud-based services (GAE, Office365, etc)
- direct SOR to SOR (student <--> HR system)
- Registry to common HE specific apps (LMSs, research management, etc)

Capabilities

CIFER Provisioning aims to make identity integration and provisioning easier to build and maintain by leveraging open source tools and community expertise. The target outcome is a loosely-coupled and agile IAM infrastructure that leads to a more efficient and secure system.

Potential Toolkit Components

- Kuali Rice
- Apache ServiceMix
- Grouper PSP (formerly ldappcng)
- talend

References

- https://spaces.at.internet2.edu/download/attachments/25861378/Straw+Man+OSIdM4HE+Report.pdf
- https://spaces.at.internet2.edu/display/OSIdM4HEteam/Draft+Provisioning+Work+Stream
- https://spaces.at.internet2.edu/display/OSIdM4HEteam/Rough+Provisioning+Architectural+Diagrams

Comments & Questions

From a potential adopter view point I come away with the idea that adopting O4 Provisioning would mostly allow me to swap out proprietary or home-grown "glue code" with open source standards-based solutions with community/commercial support. Certainly there is value there, but is there another part of the story that we're missing?

> *Rob: FWIW, I think that part of the story may have more to it than might be immediately apparent, especially among higher ed institutions.*

> *In a lot of organizations (mine included) there's a not-so-subtle back-pressure imposed on other parts of the identity infrastructure by incumbent provisioning mechanisms – "glue code" as well as (quite frequently) manual provisioning processes. Apart from the obvious issues with localized fragility and increased cost of doing business, I think many of us find that that "glue code" has taken on almost the behavior of the portland cement in concrete – it isn't just holding everything together; it's also immobilizing it. Most of our home-grown provisioning solutions tend to be more or less tightly integrated with both existing authoritative sources and existing identity consumers, and as such, whenever one of* ***those*** *systems needs to change, our provisioning solutions need to be changed along with. All too often, that's either an expensive redesign effort (if we're lucky enough to be able to undertake it at that level) or a differently-expensive "patching" effort. I'm sad to admit I've more than one case in which there's running code in our infrastructure that was originally designed to work with one system, then re-fitted with a translating layer to integrate with another, then* ***re****-re-fitted with a different translating layer to integrate with yet another, either because a ground-up redesign was too large an undertaking for the organization at the time it was needed or because no one could comfortably decipher the original intent of long-departed programmers in embedding arcane business logic within the incumbent systems.*
>
> *While I won't say that O4 Provisioning will make it any easier, necessarily, to dissociate one's infrastructure from its history, it* ***will*** *provide a much more loosely-coupled and hopefully more agile option to which to transition...*

What does it mean to drive federation deep into the IAM architecture from a provisioning perspective?

> *Rob: That's an interesting question, and one I'm not sure we've really discussed all that thoroughly, to date. It's something we should probably spend more time on in an upcoming call, as well as discussing it here.*
>
> *For the sake of target practice, I might mention a few thoughts:*
>
> *(1) There are interesting ramifications for the sorts of privacy concerns often addressed by federations when we consider provisioning, for example, identity information out of a given organization's identity system and into an application operated by one of its federation partners. Traditionally, I think federations have taken as given that identity assertions are to be generated at the time of (or very close to it) users' interaction with relying parties, and their operating rules and policies have often dictated that relying parties either not build persistent caches of identity attributes based on assertions made through the federation, or at the very least that they openly and explicitly divulge precisely what, why, how, and for how long they are caching such attributes. This not only addresses issues with reliability of information for the relying parties, but also preserves the IDPs authority to change what it asserts over time according to current identity state and local policy, and gives the IDP very precise control over attribute release within the federation. Normally, we think of provisioning in terms of persistent data stores, though – one doesn't normally think of "provisioning" attributes to an application for the duration of a session or transaction, although interestingly enough, the same mechanisms that do one can easily do the other, really.*
>
> *(2) That said, there are some very common cases that are inevitably sticky for certain relying parties within federations. One that comes to mind is the perennial case of the relying party that's perfectly happy to use assertions provided through the federation to establish a user's identity and even some of the user's attributes, but that's designed around its own user data repository and requires that the user have some local, application-specific identity information in order to make full use of the software. This is where the concept of just-in-case versus just-in-time provisioning becomes very interesting, I think – the just-in-case model works very well in some single-site scenarios, but (I suspect) starts to fail pretty dramatically in federated use cases, while the just-in time model is much easier to implement in federated cases, but raises those questions from (1) above about federation attribute use policies...*
>
> *(3) To the extent that we're also scoping provisioning and integration to encompass delivery of identity data* ***to*** *registries as well as* ***fr om*** *registries, there are probably some new use cases introduced by federation. Grouper, for example, has support for federated group synchronization between Grouper instances at different institutions, along with all the attendant external subject management – that is likely a model for a more general approach to one kind of federated identity provisioning, although exactly how that plays out when the objects being synchronized are "person" objects rather than groups or arbitrary subjects may be more complicated than I'm thinking at first blush...*

How does this proposed work enable efficient and secure access to cloud-based services?

> *Rob: To the extent that we're talking about lifecycle management of provisioned identity information across multiple cooperating (but not necessarily site-managed) service providers, I think all the stuff we'd normally say about good automated (de)provisioning tools and practices within an institutional ecosystem would translate directly to the cloud -based service space, and to the extent that the mechanisms we're talking about employing – SCIM, SAML attribute transfer, PSP, event-driven message queue management – have all (or almost all) individually been tested in various cloud-based scenarios, what we're talking about doing should be very effective at translating the already demonstrated benefits of automated (de)provisioning of identity information for site-managed services into the cloud.*
>
> *To the extent that much of what we're talking about doing amounts to rigorously defining the protocols, APIs, and data models for those boundaries between, eg., the registry and the provisioning engine, or the provisioning engine and a provisioned resource, our efforts should make it much more efficient (arguably more secure, as well) to manage enterprise identities in cloud-based services... The trick in that case, I suppose, will be getting buy-in for implementation of those interfaces from major cloud providers of interest to our community (and/or collaborating on the development of implementing those interfaces for those cloud services ourselves).*
>
> *Our first key use case objective involves a cloud-based service (Workday) integration with a site-managed registry, so we should get some good, practical experience early in the process from that.*

What is an "identity & affiliation lifecycle management engine"? How does it relate to provisioning? Should it be further defined as it is listed as major deliverable?

Rob: Probably so, and I suspect it's already been somewhat further defined during the discussion at the Phoenix f2f earlier this week. We (as a group) should probably try to review the ammended spreadsheet Keith sent out on Wednesday with an eye toward that particular deliverable, and see if we can refine the description further.

My stake in the ground (and everyone, please, pull it up and move it around, as I'm clearly not the best one to be planting any of these stakes 🙂 might be that it relates to the overall provisioning and integration effort in a similar fashion to how the the matching engine relates to the registry effort – that is, it's the holy grail of this space, in a way.

My sense of it would be that while most of the earlier work we're doing pertains to maintaining state consistency for identities by virtue of processing incremental changes to individual attributes (or at most to small clusters or tightly coupled attributes) – something that can probably be implemented with little more than typographical translations and one-to-one mapping mechanisms. For example, if an ERP changes the home address zip code associated with an individual, it's a simple typographical translation that can convert that to a transition for the "label address" for that individual in a registry, and in turn a simple mapping operation to determine that that update should result in the update of a "postal code" attribute in a downstream whitepages service.

Somewhat more elusive, but far more interesting (IMO) are the cases in which the translations needed across a provisioning or integration mechanism are more complex, involving either multiple source system conditions or "real business logic" to complete. For example, consider a case in which a student graduates and transitions from being a student to being an alumnus. In one sense, that may be a simple transition, but in another sense, it's an entirely more complex operation – the system of record for a whole host of attributes associated with the student may change, since the student's authoritative data may cease coming from the student information system and begin coming from a completely separate alumni or development ERP, which in turn may be the trigger for actually assigning the student an alumni affiliation, which in turn may mandate a whole host of changes in the states of downstream provisioned resources that go well beyond simply updating downstream systems with the individual's new affiliation value. There may be entire systems from which the ex-student needs to be deprovisioned as a result of the transition, other systems to which the student needs to likewise be provisioned, and still others in which a whole plethora of access controls need to be modified to reflect the individual's new aggregate state.

That's probably by no means the whole of it, but I think that sort of management of what I think of as "synthetic state transitions" that may not arise simply from a single change in a single attribute value but may depend on and result in a whole host of data changes is a big part of what I, at least, think of when I think of lifecycle management engines versus simple provisioning engines.