Authentication, Authorization and Allocation (A3) System

Status Quo

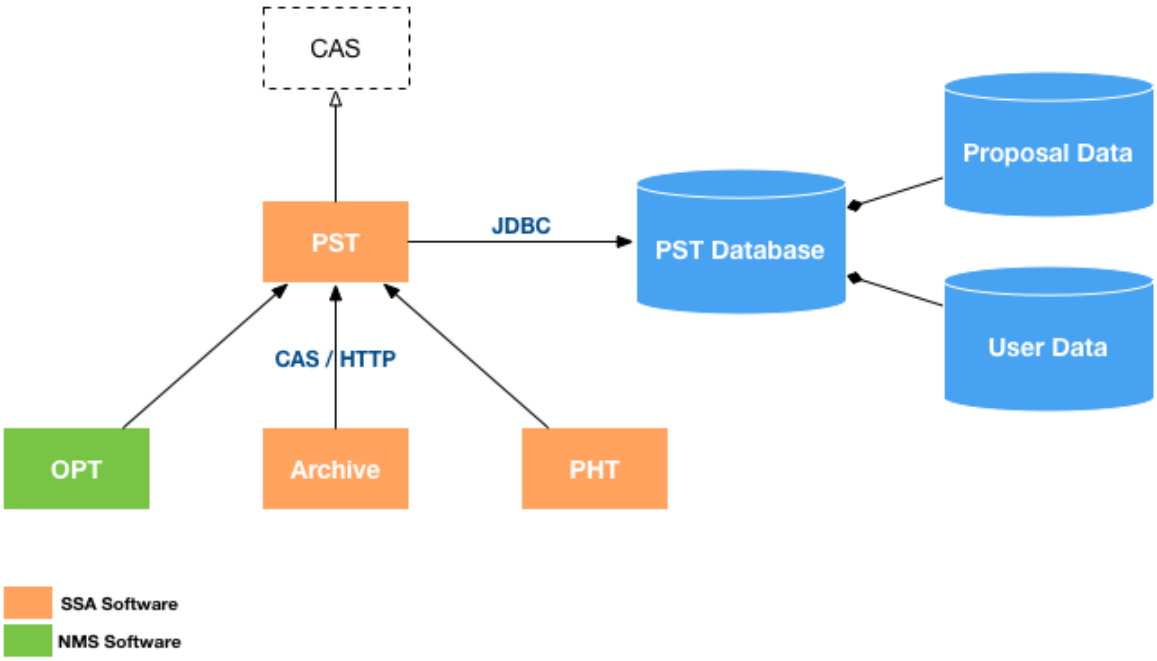
As a quick review, *authentication* is the process by which a system determines the **identity** of the user. By contrast, *authorization* is the process by which a system determines what **permissions** the user has.

In contrast to many other systems, NRAO does furnish some functionalities to users who have not been *authenticated*—the principal example of this being the archive allowing downloads to proceed based on the email address a user has entered anonymously. We consider such users *identified* but not *authent icated*—but not exactly *anonymous* because we do have a means of contacting the user.

Allocation is a future direction for this system, in which the identified user with permission to access a resource must also have that resource allocated to them. The details of that system are not well understood at this time.

Authentication

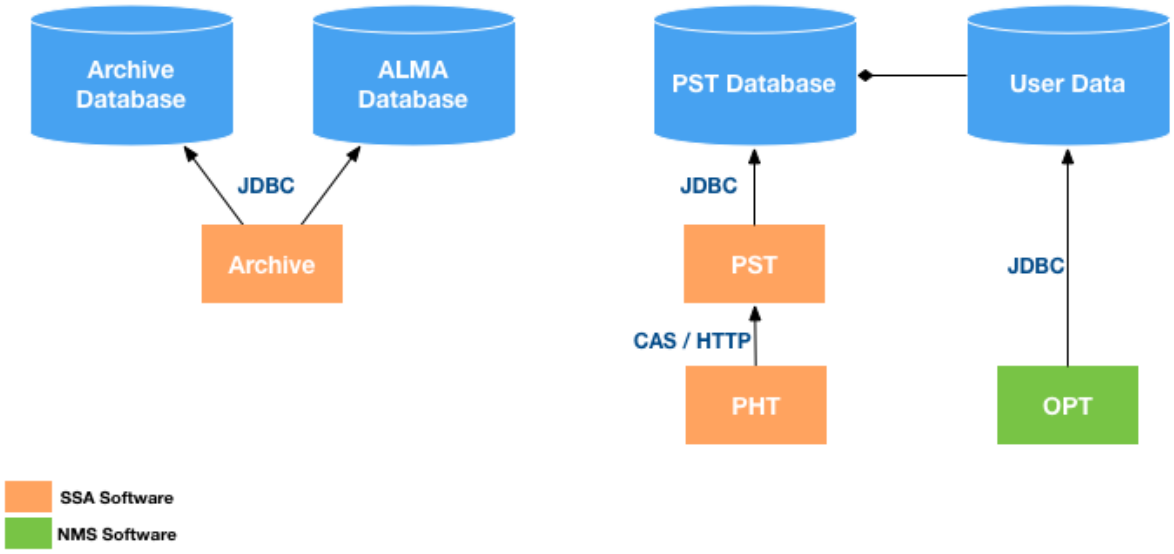
Authentication in SSA systems is well-behaved, using the CAS protocol to identify users, as shown in this diagram:



The systems are directly accessing the PST, but only via the CAS protocol. If another CAS URL is configured in these applications that also hosts the CAS protocol instead of the PST, they will continue to authenticate properly. This fact is leveraged by the summer school and synthesis imaging workshop OPT deployments, which are configured with a separate standalone CAS.

Authorization

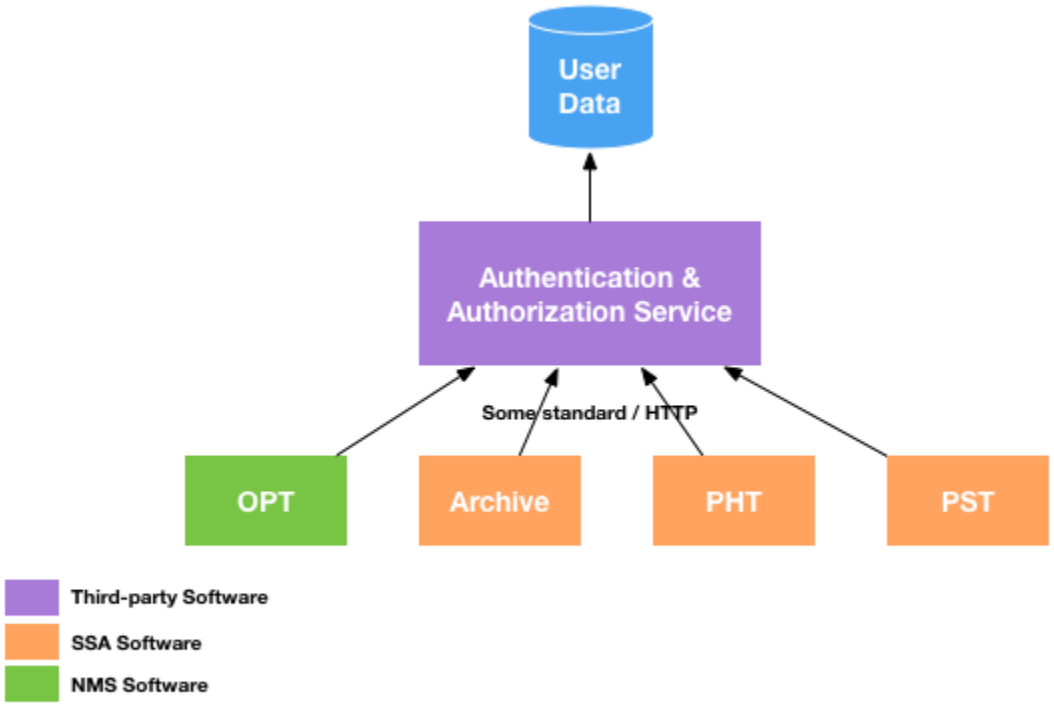
Authorization with CAS is limited. CAS will apprise the software of groups that the user is assigned to, but that functionality doesn't appear to be perfectly hooked up in the PST CAS installation. As a result, there is no standardization for authorization.



As you can see, most of the applications have custom authorization logic which accesses databases directly. The exception is the PHT, which uses the user's group in CAS to determine whether they are authorized to use the PHT; if they are, all functionality is available. The OPT accesses group information in the PST database directly, and the archive uses the authenticated identity to search both its own and ALMA's databases for authorization.

Ways Forward

The ideal end scenario would be one in which a single chart is sufficient to express both authentication and authorization:



The details of what that protocol or service might be remain vague at this time. A non-exhaustive list of candidates for the service or protocol itself would be:

- CAS
- Shibboleth
- OAuth

The details there matter since different regimes come with significant pro/cons:

1. OAuth has wide tool support, including in HTCondor, and is easier to integrate with 3rd-party social web applications
2. Shibboleth is widely used in industry and provides truly federated authentication and authorization as well as mutual authentication; with ALMA also supporting it, the federated authentication problem would be solved for us
3. CAS is still widely used in university settings as well as a large number of internal systems that we may not want to change, but does leave us holding the bag (as we are now) with federated authentication with ALMA

Taking it for granted that something can be decided, there are several ways to get there with different risks.

Top-down implementation (risky)

It seems like the riskiest approach would be to rip out the authentication system from the existing PST and replace it with something new. This creates downstream work for all the other applications. The approach would look something like this:

1. Decide the technology for the replacement
2. Extract the user management stuff from the PST
3. Use replacement technology front-end if possible or else build a new account management front-end
4. Enumerate the downstream applications using CAS and the PST user database
5. Update each of the downstream applications

Progressive implementation

A much less risky approach would be one in which the TTAT project slowly removes functionality from the PST until only the user management functionalities remain. This gives us a small legacy system to manage while running two protocols in parallel atop the same database.

1. Decide the technology for the replacement
2. Once TTAT reaches a production deployment stage, remove proposal functionality from existing PST. The PST is reduced to a legacy user portal.
3. Augment the legacy user portal with the new authentication and authorization technology (either in-code or atop the same database)
4. Enumerate downstream applications using CAS and the PST user database
5. Update each of the downstream applications to use the new authorization technology

6. Create or deploy a new user portal based on the new technology
7. Decommission the legacy user portal

This approach has several advantages:

1. Neither the TTAT nor the Workspaces projects need to adopt the entire problem of authentication and authorization
2. Greatly reduced possibility of breaking legacy applications

The disadvantages are:

1. Lifetime of the PST application is greatly prolonged
2. Still potentially creates a moment when missed applications suddenly stop working

Miscellaneous

Ryan Fox's work

In 2014–2015, SSA had an employee Ryan Fox, who built a new account system by extracting the existing one from the PST into a separate application and database. The code repositories for these are hosted in gitdev.nrao.edu as follows:

- git@gitdev.nrao.edu:accounts-db-java
- git@gitdev.nrao.edu:accounts-db-python
- git@gitdev.nrao.edu:accounts-gui
- git@gitdev.nrao.edu:accounts-query
- git@gitdev.nrao.edu:userdb

It would require some rework to modernize these, but not probably a huge amount. This would be especially useful following the top-down implementation plan but may also turn out to be helpful in the progressive approach towards the end.

Modernizing CAS

We are currently stuck on CAS 3.0, but modern versions of CAS are up to version 7 or so. Simply upgrading it may support the existing applications while opening up significant authorization possibilities. Research into this was conducted last year by Richard Falardeau, which I think mostly wound up in the following repo:

https://open-bitbucket.nrao.edu/projects/SSA/repos/user-portal/browse

This should feed the first step of both implementation plans, determining the successor technology.