CAS 4.1 Roadmap

The following items are ***proposed*** to fit the scope of the CAS 4.1 release. We are just focusing on the big picture here. Other smaller issues may also be fit candidates depending on the nature of the change.

> **Scope**
>
> This document specifically addresses the scope for the CAS Server 4.1. Features for official CAS clients that would want to take advantage of 4.x features should be documented and discussed elsewhere.

> **All about draftness!**
>
> This is just a draft and may be heavily edited as development moves on. Items that will not fit the release schedule and timeline will be removed from this list. We are just trying to gather and collect proposals for the release.

Open Items

~~Front Channel SLO~~

~~The existing front-channel SLO feature in CAS4 is still experimental. Improvements could be made in terms of UI or client integration.~~

~~Rather than dancing with the client, we could directly call apps from the CAS logout page/flow to logout. This can securely done in parallel invocations (via hidden images, iframes, etc) and possibly may require the creation of a new field in the service registry for "logout urls". We would need a specific url for logout that would be used for front channel SLO, with logout message that is SAML like, which is to include the service ticket. The message can be hashed and zipped and sent along as a GET request. This would allow the CAS server to present 3 SLO options:~~

1. ~~Back channel~~
2. ~~front channel with specific logout URL; also useful in cases where the app does not have proper client support for SLO~~
3. ~~front channel with no logout URL; in this case we simply call the service url as before which is useful for cases where there is no specific url for logging out on the app side.~~

~~This may require mods to the protocol.~~

~~We have to be VERY careful with the wording of front channel SLO on the UI. We cannot never guarantee a logout from the app POV, but can emphasize that a logout message has been sent to the application. It is still up to the application to decide how to handle the logout.~~

~~Proposed by Jérôme Leleu~~

~~Oauth server support~~

~~CAS server can be customized to act as an OAuth server. Presently the OAuth implementation requires that the client receives the TGT to pass to the profile as an access token. Also, the implementation attempts to release all attributes rather than those are allowed due to the limitations in current design. The following alternatives may be used instead:~~

1. ~~Rather than passing the TGT directly to the client, we simply *encrypt the entire CAS validation response at the time of authorization* using a technology such as JWT. The encrypted response is sent to the client, and submitted back again to the profile endpoint at which point it is decrypted again and attributes are released.~~
2. ~~Alternatively, rather than encrypting the whole response we could just simply *encrypt the TGT and the service* and pass it along to the client, which when received again as the access token, would be decrypted to release user attributes.~~
3. ~~Or, we could design a new ticket type, that is a true access token made up of random strings that is passed to the client. This would replace the TGT as the token, and does not require any kind of decryption.~~

~~Spring Security's OAuth support may be a candidate to review.~~

~~Proposed by Jérôme Leleu~~

~~Encryption of ticket registry data~~

~~Encrypt/Hash the ticket registry as appropriate to avoid people either stealing or tampering with the registry, either in the wire, in memory or on disk. Proposed by Proposals to mitigate security risks under SEC-9.~~

~~Proposed by Jérôme Leleu~~

Done Items

Management App Facelift

The CAS services management webapp is in dire need of attention. Improvements to UI, display of fields as well as support for OAuth services, attribute filters, and other service settings and types would be considered.

By Misagh Moayyed

Support consumption of SAML's Metadata MDUI

Consider the service registry can be augmented to retrieve the MDUI info for a given entityID from the IDP's metadata sources, in cases where CAS is handling authn for a Shib Idp. The Shibboleth-CAS authn plugin is already equipped to pass along the entity id. This task would be to ensure the received entity id can in fact be looked up, MDUI retrieved and consumed, finally rendered on the CAS login page.

Note that the CAS server as of 4.1 has the ability to display a logo and description for each service access in the registry.

Proposed by Bill Thompson, Misagh Moayyed

Secure release of client credential, PGT and (optionally) CAS attributes

Modify the clearpass mechanism so that the credential is returned as an encrypted authentication attribute. Modify service registry to allow public keys that would encrypt the password as well as the PGT. Optionally, allow for a configuration that would also encrypt other user attributes. This is a proxy approach that is used to verify and authenticate the service, and in the case of clearpass and PGT, should greatly reduce the pain of callback URLs. The public key should be generated by the client and can be sent to the CAS server via insecure means such as email. By default, all other attributes would be sent as plain text OOTB for backwards compatibility and debugging purposes. Rather than using a different or newly invented method, we simply trust SSL.

Potentially deprecate the clearpass module and update the docs. This may also impact the CAS protocol in the way that PGT and credentials are sent over, as well as the encryption of attributes.

We are NOT going to remove the existing methods of callbacks, but simply would deprecate them for now and can mention the new method as an optional feature of the protocol for now.

This is discussed and proposed under SEC_8: Proposals to mitigate security risks

Proposed by Misagh Moayyed

Role-based Access Control and Authorization

Provide a facility to enable role-based access control that would attempt to decide service access rules based on user attributes. By Misagh Moayyed

Deprecation of uber-webapp and jboss-cache modules

Neither of the two modules have received any attention really, (other than to make sure they don't have breaking APIs) and we have hardly ever had a question on the mailing list on either. Subsequent CAS versions may choose to fully drop the module. By Misagh Moayyed

Refactoring of attributes filters, and attributes per service

Attribute filters in CAS can be configured per service. We'd like to take that one step further, and describe attributes release policies in conjunction with filters, such that policies can figure out the set of allowed attributes per service, that can be renamed virtually. Proposed by Misagh Moayyed

Proxy config per service should authorize callback urls

The proxying feature of a given service in CAS does not currently "authorize" the service, other than to ensure it can be reached via SSL. At the recommendation of the AppSec group, and well articulated by Jérôme Leleu and David Ohsie, improvments can be made so that the pgtUrl can be controlled and authorized by the CAS config, so we know who really is allowed to received the PGT, in addition to the proxy.

CAS-specific truststore for proxycalls and handling SSL certs

CAS currently uses the JDK's default truststore to establish ssh handshakes specially for proxy calls. This can be improved by providing a CAS specific truststore, that would be empty by default. Untrusted proxies can be imported inside this particular store. Separating the store from Java's default always helps with platform upgrades that may cause prev changes to be overwritten.

This is proposed under SEC_5:
https://wiki.jasig.org/display/CAS/Proposals+to+mitigate+security+risks

Note that the default keystore would possibly be used in addition to the already available certs in Java. We simply just want to avoid polluting the default,and allow adopters to carry over their store, irrelevant of jdk version.

Client-side Spring Webflow session management

The spring webflow's conversation state is managed by the server, which causes issue due to web session timeouts. lets move the management of this session over to the browser. Marvin S. Addison has developed a perfectly suitable solution. By Misagh Moayyed

Retiring JIRA and using Github Issues for task tracking

JIRA seems too heavyweight for what it's now being used for, which is mainly tracking issues and improvements. Github issues provides a more pleasant alternative. We have to do a little bit of work to create some appropriate tags that correspond to our existing JIRA issue types, but that's quite simple to do, takes very little time and the process is in fact quite customizable. Every issue can be assigned to a milestone, and may be tagged with many other decorations that JIRA provides. Issues can be assigned to developers, can have "Affects Version" and "Fixed in Version" and many other tags that we feel may be more relevant. By Misagh Moayyed

GitHub CAS Downloads

Rather than providing binary downloadable artifacts per release on the jasig website, it seems like the release engineer for a given CAS release has all the right permissions and tools to take advantage of the Github's releases feature, where the binary artifact, cas-webapp as well as release notes can directly be hosted and uploaded there. The jasig website could then perhaps just include a link to the latest release, or to the download area. Proposed by Misagh Moayyed

CAS Protocol on Docs Site

Moving the CAS protocol off the Jasig website and onto the GH pages docs site: I had a lot of trouble keeping to the syntax of the WYSIWYG editor, which truly was necessary work. So in the spirit of synchronicity, I'd like to include the protocol doc in the documentation somewhere, so that it stays with the version of the CAS software that is released. Proposed by Misagh Moayyed

JSON Service Registry

Persistence of service definitions into a JSON file. Marvin S. Addison and Unicon have both solutions that could be merged and consolidated into one awesome registry! By Misagh Moayyed and Marvin S. Addison