Single Logout

Introduction

As of version 3.5.0, Kantega SSO supports SAML Single Logout (SLO). SLO allows users to be signed out of the IdP and all apps participating in the session simultaneously, whereas without SLO only the local Atlassian session is affected.

At least in theory. The reality is that the level of support for SLO varies wildly from IdP to IdP, and the protocol itself should be regarded as **best-effort** regardless of IdP. While it will usually work (within the limitations of the IdP), know that logout *can* sometimes fail partially or fully, leaving the user only partially logged out with orphaned application sessions. If your main objective is termination of the application- and IdP-session, SAML SLO will do what you need; every IdP with any level of support for SLO should handle this fairly reliably. However: If 100% *reliable* global logout is critical to your use-case and the occasional broken logout chain or lingering/orphaned SP session is unacceptable, consider introducing an API gateway to your architecture instead (or stick with Crowd SSO). Basically any kind of architecture based around global session state and semi-regular polling or a more robust/bespoke notification/event mechanism.

Finally, educating users is crucial to deploying single logout somewhat successfully, including:

- What to expect when clicking logout.
- What to do and/or expect when things inevitably fail occasionally.

Single Logout is currently not supported for Fisheye/Crucible.

Quick glossary

A quick index of terms used throughout this documentation.

- SP: Service Provider, e.g. Jira, Confluence
- IdP: Identity Provider, e.g. ADFS, Google, Okta, Azure AD, Keycloak, Shibboleth
- SP-initiated logout: User clicks logout in e.g. Jira
- IdP-initiated logout: User clicks logout from the IdP side (dashboard/application menu).

How it works

SAML login is decentralized in the sense that there is no global session state that all session participants share&check after authentication has taken place. Service providers (e.g. Jira and Confluence) establish their own local sessions based on an assertion/claim of identify from the identity provider at the time of login, but do not check in with the IdP (e.g. ADFS) after that to see if a session is still valid. As a result, SAML Single Logout is actually a notification protocol: The IdP maintains a record of every SP that has participated in any given user session so that when logout is initiated, the IdP can inform each session participant (possibly with the exception of the initiating SP) by sending them a *LogoutRequest.* This lets an SP know that it should invalidate the given session. Each SP must then acknowledge with a *LogoutResponse,* indicating whether session termination was successful or not to the IdP. Depending on whether logout was initiated by the IdP itself or an SP participating in the session, the user is then presented with the result in different ways:

- IdP initiated: The user clicked logout from the IdP dashboard/portal (e.g. ADFS). The IdP notifies each SP (e.g. Jira and Confluence).
    - When done: The user will usually land on the logout confirmation screen of the **IdP**.
- SP initiated: The user clicked logout from the SP (e.g. Jira and Confluence). The initiating SP redirects to the IdP with a *LogoutRequest*, which takes care of sending additional *LogoutRequests* to the other SPs participating in the session.
    - When done: IdP responds to the initiating SP by sending a *LogoutResponse.* The user will typically land on the logout confirmation page of the **initiating SP**, but it's really up to the SP as has has control passed back to itself in this case.

All of these notifications and responses are sent via the user's browser using redirects (back-channel bindings for server-to-server notification exists, but is not widely implemented, and we have no plans to support it currently). So logout is a daisy-chain of redirects from one service provider to the next. It follows that if any of the service providers in the chain (or the IdP itself) for some reason fails to redirect onwards, the chain breaks and the user is stuck at the breaking point with little or no idea how they got there; usually some arbitrary error page of the offending session participant.

This can of course be confusing to the user. To mitigate this, some IdPs implement the redirect notification chain inside an iframe. The user typically sees a "we're now logging you out, please wait" page while the IdP runs the SP notification chain inside the iframe. If any participant fails and breaks the chain inside the iframe, the IdP still has control of the outer page so the user isn't stuck on a completely arbitrary and potentially confusing error page he/she has no idea how they got to. The exact implementation and behaviour of this varies from IdP to IdP, however. As with all other SSO related tasks, we strongly recommend testing capabilities in a **staging environment** before going to production. And make sure to educate users on what to expect first.

Further reading on some of the challenges/difficulties of SAML Single-logout here: https://wiki.shibboleth.net/confluence/display/CONCEPT/SLOIssues

Implementation details and limitations

- Kantega SSO initiates SLO using the HTTP-Redirect profile only, for now. Incoming messages (LogoutResponse/LogoutRequests) handle both HTTP-Redirect and HTTP-Post binding, however, and the appropriate binding will be used to respond.
- All communication is done via the browser: There are no server-to-server calls between any service provider or the IdP. The only network requirement is that the user's browser has access to the application.
- Single logout is only initiated when the user *explicitly* clicks logout in the Atlassian app or on the IdP side (for the few IdPs that support this). As an administrator, consider how often users actually use the logout button actively (as opposed to just allowing their session to time out) before taking on the added complexity of introducing SLO.

Our implementation does not currently support *cross-device* or *cross-browser* SLO. This is a design trade-off where we don't think the cost and complexity (additional distributed caching/session blacklisting layer would be required) makes up for the benefit to most users, but admins should be aware of the limitation.

For example:

- The user **mark.miller@example.com** is signed into his IdP on both his iPad and Windows PC.
- Mark is currently logged into Jira on his Windows PC.
- Mark is also using an in-house SAML capable app on the iPad.
- Mark then clicks *Logout* from his IdP dashboard on the **iPad** (IdP-initiated logout), causing the IdP to send a LogoutRequest to all service providers currently participating in any of Mark's sessions.
- In this case, the logout chain is performed by the iPad. We are *unable* to sign Mark out of Jira as the iPad does not have Mark's session cookie: *Mark can continue to use his PC Jira session*

Identity providers

These are the identify providers we have tested with SLO so far, along with a setup guide for each and a short summary of capabilities/limitations. You should be able to configure SLO for any other IdP by loosely following these instructions, provided the IdP supports SLO by HTTP-Redirect, however.

As ever, feel free to let us know if you would like a guide for a certain IdP currently missing. Also don't hesitate to contact support for questions or other feedback.

AD FS

ADFS fully supports both IdP- and SP-initiated logout.

See: SLO: AD FS.

Azure AD

Azure AD has limited support for SLO.

While both SP- and IdP-initiated logout are reportedly supported, neither actually works. The result is that only basic SP-initiated logout is supported, which allows logout from AAD and the initiating SP.

See: SLO: Azure AD.

GSuite

GSuite does not support SAML Single logout as an IdP.

Keycloak

Keycloak fully supports SP-initiated Single logout, but to our knowledge does not support IdP-initiated logout.

See: SLO: Keycloak.

Okta

SP initiated logout is partially supported: The IdP and initiating SP's sessions are terminated, but other session participants are never notified.

IdP-initiated logout is not supported.

Okta also has a "quirk" where the user must be logged in to initiate logout, or the user gets a login prompt. SLO then proceeds, causing the user to be immediately logged out again. This creates an annoying user experience when there is more than one session participant, for example Jira and Confluence. Unfortunately, there isn't really anything we can do about it.

To illustrate this with an example where a user is logged into Jira and Confluence through Okta.

- The user clicks Logout in Jira
- The user is signed out of Okta and Jira but Confluence is never notified.

If the user now clicks Logout in Confluence:

- The user is again redirected to Okta because Confluence has to initiate single logout (there's no way to know logout has already happened).
- Okta now presents the user with a login form because the user already terminated the old Okta session.
- Upon logging in, the user is immediately signed out again.
- ... repeat for every other session participant the user wants to log out of.

See: SLO: Okta.