SessionCache

Overview

The <`SessionCache`> element configures the plugin used for the caching of user sessions and typically has a wide range of options affecting memory usage and behaviors particular to advanced features. The cache manages access to user attributes and other general session data, access to the underlying SAML assertions in advanced cases, and enforces session lifetime and timeout policy.

This element can be omitted, resulting in the "StorageService" cache type being used, on top of the default, in-memory StorageService, with other options defaulted.

As with all plugins, the **required** `type` attribute defines precisely which type of plugin to configure. Each type may have its own configuration options. The only implementation supplied with the software is the StorageServiceSessionCache.

Note that the session cache is not where you configure timeout policy, because that may be determined on an application-specific basis. So that's actually set via the <`Sessions`> element within the application configuration section(s) of the file.

Session Recovery

The SessionCache supports a feature called "session recovery" that implements a limited form of client-side session storage. It enables multiple server "nodes" running the SP that share a common secret key to react to a session cookie identifying a "missing" session by checking for a recovery cookie containing the session and loading it back into the cache, somewhat obviating a shared session cache.

The feature is enabled by supplying a list of attributes to save and transfer between nodes using the `persistedAttributes` setting. The feature also requires that a <`DataSealer`> element be defined in the configuration. This supplies the key(s) used to encrypt and decrypt the cookies, and the security of the SP as a whole is severely compromised if those key(s) are compromised. Revoking them is however a simple matter of deleting or editing a file.

Once these two configuration settings are in place, the feature "just works" and you'll start to see log information referencing the recovery of sessions if you move between nodes.

The session is necessarily limited in size, and it is advisable to ruthlessly trim down the set of attributes you need in order to keep the volume of data from becoming a problem. Failures will be generally difficult and can even cause a client to stop functioning outright with your virtual host(s). The SP inherently limits the data stored by omitting some information from the session, such as the original SAML assertion, so access to the assertion is incompatible with this feature.

An additional limitation is a lack of session timeout. Sessions continue to expire after a fixed period of time from creation (the "lifetime" of the session) but any time a session is recovered, it is assumed to be valid and is marked with that time as the last time used, so a client can repeatedly bypass any timeout setting by moving between nodes (admittedly this may be impossible for the client to control easily).

Logout is also somewhat compromised by this feature. The system does have support for tracking a "revoked" session in a shared storage service, but since such a service would essentially undermine the point of using this feature, it wouldn't be expected to be used. The presumption is that a user seeking to logout would expect and want the associated cookies to be cleared, but if a user took steps to preserve and recreate the recovery cookie, it would work on any node without access to the logout/revocation record for that session.

Reference

Common Attributes

All SessionCache plugins support the following attributes.

A number of them are a shotgun approach to dealing with a fairly obscure but serious problem that arises when an SP is monitored or performance-tested via a service account that causes an IdP to issue it assertions with a fixed Name Identifier value. This causes problems with the reverse mapping index that the cache needs in order to support SAML logout, so various settings can be used to limit the impact of this practice without a total redesign of the software to handle it.

Name	Type	Default	Description
`type`	string	`StorageService`	Specifies the type of Session Cache plugin to use.
`cacheAllowance`	seconds	0	Adds the time specified to a session's application-derived timeout setting to determine the amount of extra time, if any, to leave an expired session in the cache (this is basically "slop" time to make logout more reliable). If timeouts are disabled in a given case, then this setting still applies, so may also act as a lower bound on the practical lifetime of sessions in the cache. If both timeouts and this setting are zeroed, then the lifetime is itself the only bound on the session's expiration from the cache.
`maintainReverseIndex`	boolean	true	When false, disables the ability to reverse map from a SAML Name Identifier to the associated session(s). This is required for SAML logout, but is unused otherwise, so can be disabled to improve performance.

`reverseIndexMaxSize`	integer	0	Limits the number of sessions tracked by the reverse index for a given identifier, or no limit by default.
`excludeReverseIndex`	whitespace-delimited list of strings		Supplies a list of Name Identifier values to exclude from the reverse mapping of identifiers to sessions. Useful to maintain logout support, but exclude identifiers used in load testing or monitoring.
`persistedAttributes`	whitespace-delimited list of strings		Enables support for a new feature in V3, a session recovery capability that allows sessions to cross server nodes by saving important data to an encrypted cookie and reconstituting the session as needed. This is described above.
`unreliableNetworks` $^{3.1}$	whitespace-delimited list of CIDR masks		This is a modifier that loosens the comparison performed by the session cache when the <`Sessions`> element's `consistentAddress` setting is "true". It permits session use if both the bound address in the session and the client's current address both live within a particular network as defined by one of the values in the list.

Common Child Elements

None