

Signature-Based Single Sign-On Framework

Copyright 2003-2004 net&works GmbH, sso(at)naw.de

This document is published under the Open Content License
available from <http://www.opencontent.org/opl.shtml>

This software and concepts are provided as-is,
without any statement or warranty regarding its correctness, features and security.
The terms of the GPL <http://www.gnu.org/licenses/gpl.html> apply.

Table of Contents

Signature-Based Single Sign-On

Framework.....	1	SSO Agent --- TPA Adapter (invocation: command line).....	3
Architectural Overview.....	1	SSO Agent --- TPA Adapter (invocation: PHP method).....	4
Options for Single Sign-On Integration.....	2	TPA Adapter Development.....	4
Formats and Definitions.....	3	TPA Adapters - Enhanced Features.....	4
SSO Server --- SSO Agent.....	3	Understanding SSO Agent Error Messages.....	5

This is a general description of the architectural framework and general definitions for Signature-Based Single Sign-On. Please see [here](#) for the most recent version.

Architectural Overview

Signature-Based Single Sign-On does NOT rely on a central reverse proxy, on server-to-server communication etc. but allows direct access to the TPA by securely passing a one-time-token to the browser (via URL). Thus, TPAs may be distributed across the net.

Basically, we find a 3-layer architecture:

- TYPO3 (the Single Sign-On extension) dynamically creates a link ("SSO Link") that includes the desired TPA, user name, and security information (lifetime and signature).
- The SSO Agent, located on each target system (the machine where the TPA lives), validates the incoming browser request (coming from the SSO Link), talks to the TPA Adapter, and gives back an HTTP redirect to the browser that points to the TPA itself.

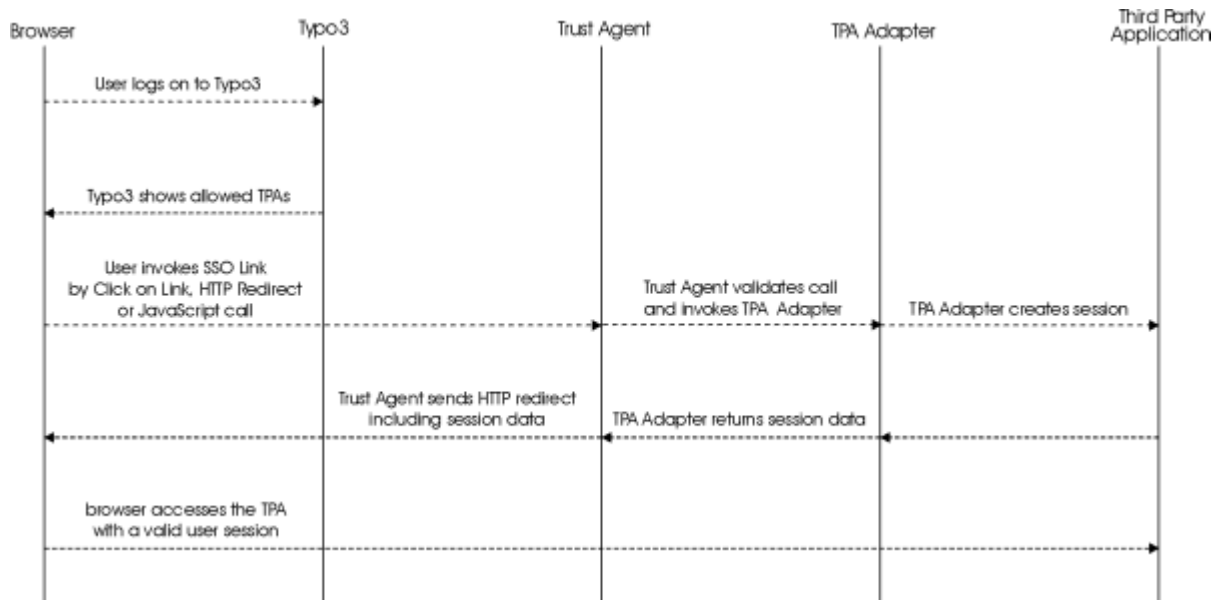
Currently, the only SSO Agent implementation is in PHP script, but it can easily be ported to other languages.

SSO Agents may be installed on multiple target system if you have TPAs on more than one server.

- The TPA Adapter is invoked by the SSO Agent (e.g. via PHP include or via system call). It creates a valid user session ("logs on the user") by application-specific means, and returns all information needed to the SSO Agent (in a defined format).

This adapter is TPA-specific - this means that you need to find or develop an appropriate adapter for every TPA that you wish to integrate. It may be written in any language you favour. See the documentation that comes with your SSO Agent for details.

See [here](#) for a repository of existing TPA adapters.



Picture 1 Signature-Based Single Sign-On: General Concept

One target server may optionally contain several TPAs, so each SSO Agent can be configured for more than one TPA adapter.

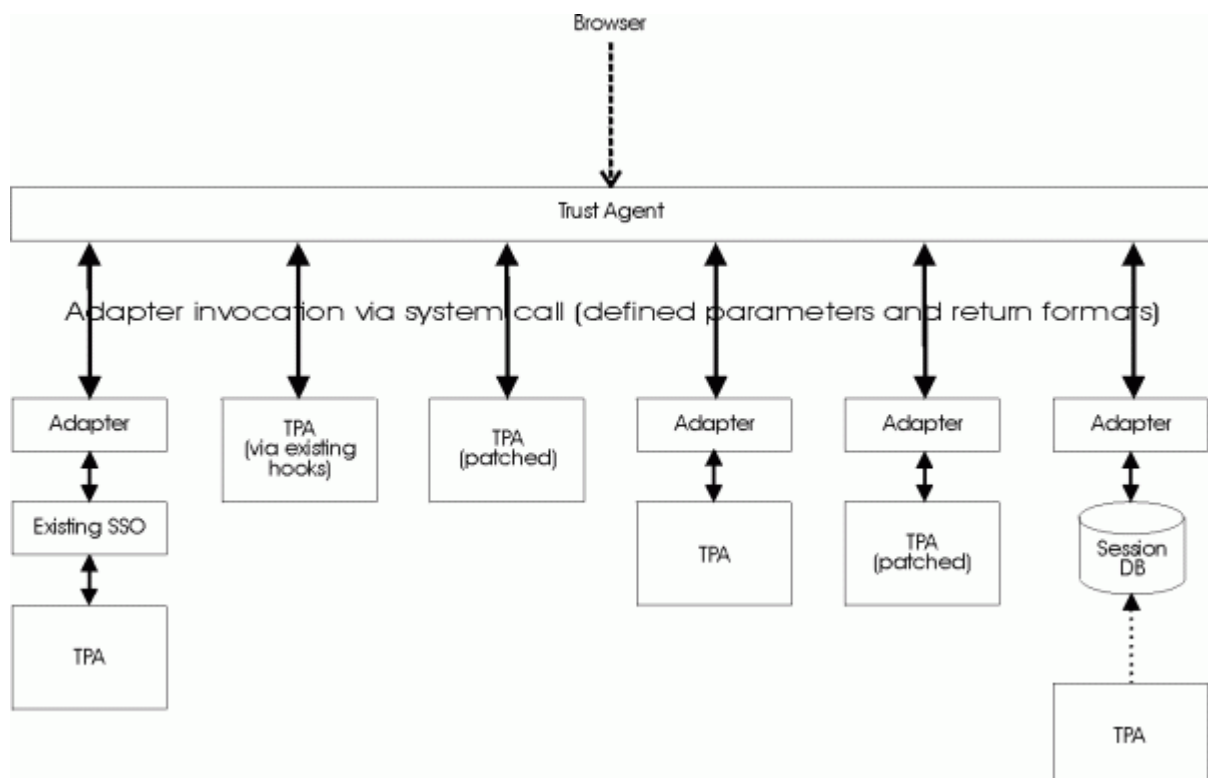
Options for Single Sign-On Integration

If you are going to integrate another TPA into your TYPO3 Single Sign-On environment, the first look will always be for existing TPA Adapters or generic ways like an adapter for an underlying Application Server (see).

In case there is no such TPA Adapter available, you will need to create your own. The two major options you have are:

- Use an existing single sign-on method supported by your TPA that is trust-based (i.e. does not require the user's password). Although this is rather uncommon, it is the recommended way if possible. What you effectively do is build an adapter to the single sign-on method supported by your TPA; the TPA itself is not affected.
- Find a way to create a user session in your TPA. This may be done by hooks in your TPA, patching it, external code, by direct access to some session database etc. - it all depends on the nature of your TPA. Obviously, all this may become pretty tough if your TPA is not open source.

This leads to a considerable number of alternative variants; some of them are shown in the following picture.



As a convention, each of these variants is called "TPA Adapter", even there if is no dedicated (or no TPA-specific) adapter software.

In rare cases, implementations may make sense that do not separate the SSO Agent functionality from the TPA adapter. These are called "Monolithic TPA Adapters".

Formats and Definitions

When developing own components, please make sure to follow these standards.

SSO Server --- SSO Agent

The SSO Agent is invoked via HTTP(S) access.

The SSO Server (TYPO3 extension) generates a link to the SSO Agent with the parameters

user	the user id that is to be logged on to the Third Party Application
tpa_id	the id configured in the SSO Agent that identifies the desired TPA
expires	the absolute time until when this link may be used (seconds from 1970)
signature	the RSA/3DES signature (using the private key configured) of the above keys and values as represented in the URL string. The order of the parameters must not be changed.

There may be more parameters may follow in upcoming versions.

Example:

```
https://my.tpa/sigsso.php?user=mytestuser&tpa_id=MyOwnApp&expires=1076925657&signature=9c79040a0a
cf28a3512a4bea6609dd2a31cbbf7421b77817f97162d051f03ee76729e3f98d26690a5bc7967f6e6c38f0454fe71a9603c6a126
d853f40b898721abf051a10b24100afd65458b95d2d5c37a4a5faa17c6be041caf028a9863049dcb15ae7c6cef47bfb171c47c50
b3424fa5d6660207b57d5e2737a76c6eb8837187b317e0171029a4705c30dc4c7dcc0716c797ae712df82be315814ef5da5b21d8
aeaf1537bdb16a6342bfbea64834853681d09461affa8ae09eb2388a104e1194141532cdc982ad9850cce6357065d8a1f5498f49a
e7ba30865ed5d1d2760332528fd1455c025f4bdf7702c83bd44dc839f5dc824d1582b024efb6a6a0aa3855
```

In PHP, the signature generation may read:

```
$this->data = 'user='.$user.'&tpa_id='.$tpaId.'&expires='.$expires;
[...]
```

```
openssl_sign($this->data, $this->signature, $this->pkeyid);
```

SSO Agent --- TPA Adapter (invocation: command line)

In this scenario, the TPA Adapter is invoked via system call.

Command line parameters are given for input values. Output values are to be returned on STDOUT.

Input values:

- all parameters must be preceded by a double dash ("--")
- parameters that must be supported by an SSO Agent and may be used by a TPA Adapter are:
 - remote_addr
 - agent
 - url
 - user

Example:

```
/path/to/tpa/script --remote_addr=%remote% --agent=%agent% --url=https://redirect.url/index.php --user=%user%
--moreparameters=anything_you_need
```

Output values:

The TPA Adapter must return a redirecturl.

The TPA Adapter can also return Cookie Parameters (see An example output of the TPA Adapter for two cookies may look like: for the parameternames). Each parametername and value is separated by a whitespace (<tab> or <space>). Each pair in a new line.

Each cookie must start with 'CookieName'. If you need to set more than one Cookie you only need to send as many Cookie parameter sets as you need. The SSO Agent will notice that different Cookies are to be set.

An example output of the TPA Adapter for two cookies may look like:

```
redirecturl https://redirect.url/index.php
CookieName  valuenam1
CookieValue value1
CookieExpires expires1
CookiePath  path1
CookieName  valuenam2
CookieValue value2
CookieExpires expires2
CookiePath  path2
CookieDomain domain2
CookieSecure secure2
```

SSO Agent --- TPA Adapter (invocation: PHP method)

In this scenario, the TPA Adapter is invoked on the PHP level.

In your SSO Agent config, just give the PHP include file (php://path/to/tpa/script.php) and the redirect-URL.

Example:

```
/path/to/tpa/script.php -url=https://redirect.url/index.php
```

This will include the script and call the function `sso($user,$remote_address,$user_agent,$redirect_url)` in the included script.

The included script needs to return an array with the following format :

```
Array(
    [redirecturl] => $redirecturl
    [0] => Array(
        "CookieName" => $cookienam1
        "CookieValue" => $cookievalue1
        "CookieExpires" => $expires1
    )
    [1] => Array(
        "CookieName" => $cookienam2
        ...
    )
)
```

where "redirecturl" is mandatory (and may include session ID et al.), cookies are optional.

'[1..*]' => Array' are arrays containing further cookies in case you need more than one cookie.

TPA Adapter Development

To develop your own adapter, one way would to inspect the existing TPA code, find the point where user logon happens and patch this directly.

In the long run you may find it a better idea to duplicate that piece of code and patch this copy, since future upgrades of the TPA software won't affect the adapter (as long as the session handling is unchanged). You may also strip down the code to the necessary parts for the adapter.

Now, what exactly does "patching the code" mean here?

- The patched code create a new session for the given user without a password
- this session data will be stored in the TPA database
- The TPA Adapter must return a redirect URL where the browser should be redirected to.
- Optionally, the TPA Adapter can return cookie parameters (see above)

If your TPA is using the "Basic Authentication" mechanism, it may be a bit more tricky to adapt this - probably you will have to patch the actual application instead of just adding an external adapter.

Make sure that the TPA Adapter cannot be called from outside e.g. via web access. A way to do this is would be to place the file in a safe directory, or to check for the existence of environment parameters.

TPA Adapters - Enhanced Features

a) TPA "Change Password" Handling

If you want to avoid some potential confusion, you should hide any existing "change password" functionality in your TPA

from users that came in via SSO. Otherwise such a user may use this "change password" and expect the new password to work with TYPO3 (which it does not, since this "change password" only affects the TPA's own user registry).

An even better way would be redirect the TPA's "change password" link to TYPO3's "change password", again: for SSO users only.

Anyway: What you need to implement is

- some flag in the user session that marks it as an "SSO-session" and
- a switch that hides or changes the TPA's "change password" link for "SSO-sessions".

[Of course... If you happen to have a nice environment with a common user/password repository (e.g. shared LDAP) in place, you will not need all this. Beyond the scope of this document, though...]

b) Policy Enforcement for Single Sign-On

As another enhancement, you may want to check the user's role (i.e. group membership) during TPA Adapter sign-on. A possible policy would be to deny single sign-on access for administrators, another would be to restrict certain users or roles to special IP addresses or networks.

Understanding SSO Agent Error Messages

Below is a list of error messages that can occur from an SSO Agent.

Key	Possible Reasons
user_missing	can only be caused by bug in Typo3 extension or by intentional manual manipulation of SSO Link (attack)
tpaid_missing	can only be caused by bug in Typo3 extension or by intentional manual manipulation of SSO Link (attack)
expires_missing	can only be caused by bug in Typo3 extension or by intentional manual manipulation of SSO Link (attack)
signature_missing	can only be caused by bug in Typo3 extension or by intentional manual manipulation of SSO Link (attack)
tpaid_unknown	misconfiguration in Typo3 extension or Trust Agent config: TPA_ID must match
x.509key_missingconf	X.509 key not configured in Trust Agent config
x.509key_missingfile	X.509 key not found (or not accessible) at the location configured in Trust Agent config
usedtokens_missingconf	"used tokens" file not configured in Trust Agent config
usedtokens_missingfile	"used tokens" file not accessible at the location configured in Trust Agent config
usedtokens_allreadyused	SSO Link has been used before. This may have different reasons: <ul style="list-style-type: none"> • User did a doubleclick (instead of single click) • User opened TPA before and now tries to open it a second time (without Typo3 having generated a new SSO Link) • Page containig the SSO Link is cached somewhere (server, browser, or inbetween proxycache). Note: Typo3 caching is coded to be always disabled for pages that contain Single Sign-On content elements. • Replay attack – someone intentionally tries to reuse the SSO Link
signature_invalid	can only be caused by bug in Typo3 extension or by intentional manual manipulation of SSO Link (attack)
tpa_error	TPA adapter returned an error – hopefully including some useful hints (Example: "user xy unknown in this application")
logfile_missingconf	logfile not configured in Trust Agent config
logfile_missingfile	logfile not accessible at the location configured in Trust Agent config
expires_exceeded	SSO Link lifetime exceeded – either it is actually exceeded because the user waited to long until clicking the link (consider increasing the Lifetime value in the configuration of the Single Sign-On content element) or the clocks are out of sync