

Strong authentication

implemented with the Shibboleth 2.0 IdP and Free Auth



SWITCH

Serving Swiss Universities


Kaspar Brand

kaspar.brand@switch.ch

At the 2006 AAI Info-day...

Assurance Levels – Asserting the Quality of Identities (Lukas Hämmerle)

Implementation



Setup

- CAS 3.0.5 dual login
- LDAP and x509 authentication
- X.509 certificates must match email address in LDAP repository
- Virtual keypad (against key-loggers)

Outlook

- Switch to production status
- SMS Token authentication
- Shibboleth 2.0 and AuthenticationContext

Experimental Test Home Organization

Enter your Username and Password.

Username:

Password:

Clear Login

Or log in with your X509 user certificate.

Languages: [English](#) | [Français](#) | [Italiano](#)

Experimental: Click on the grey keypad symbol in order to enter loginname and password when you sit at an untrusted computer (e.g. a public Internet terminal). This is an experimental counter measure against keylogging programs.

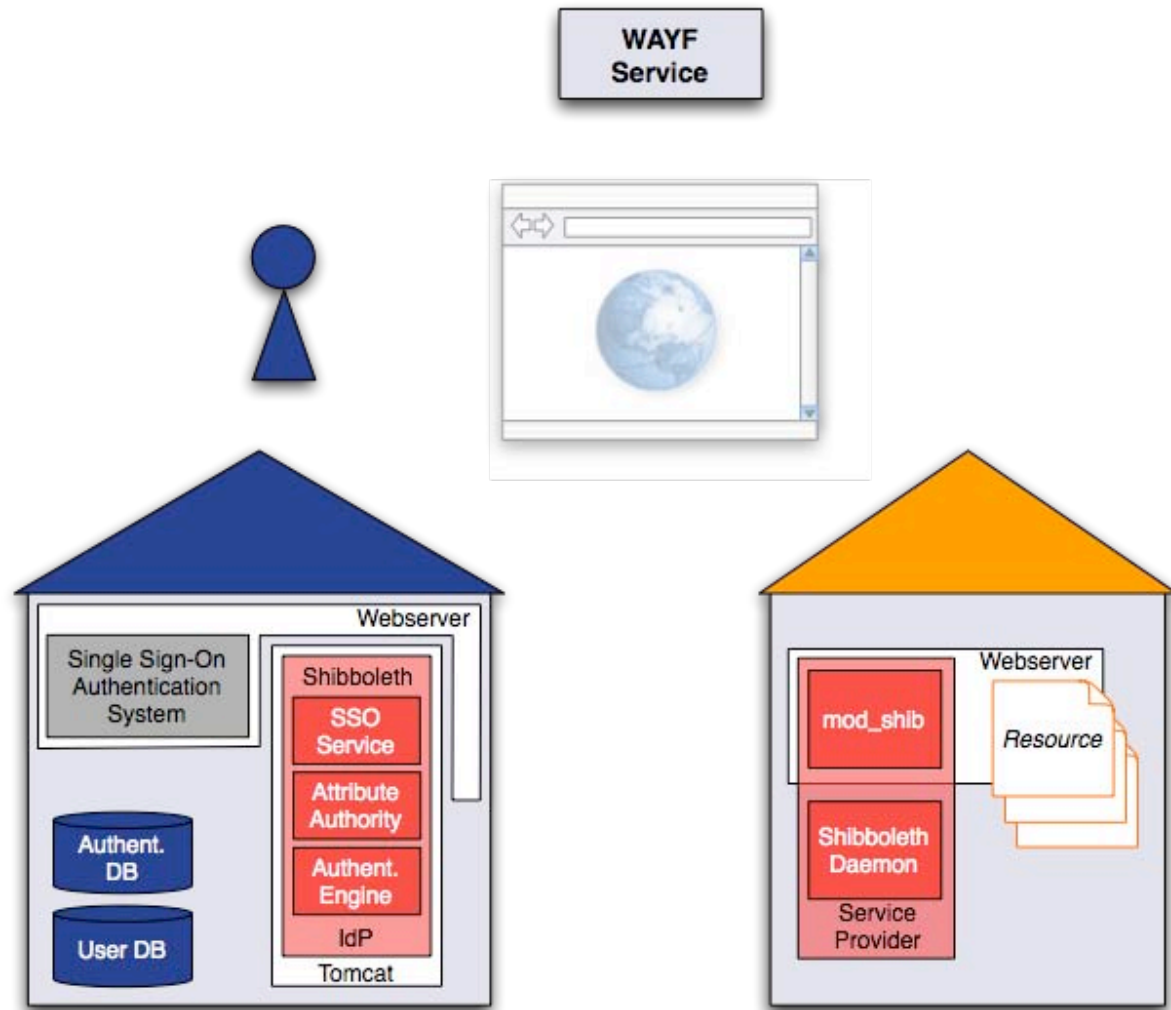
2006 © SWITCH AAI Info-Day, 21. November 2006, Bern 9

Strong authentication

- at least two out of three authentication factors:
 - something you know (password, PIN etc.)
 - something you have (card, token, ...)
 - something you are (fingerprint, iris scan, voice, ...)
- X.509 certificates aren't everybody's darling, so we were looking for a solution...
 - with a smaller footprint on the client side (if any)
 - where software is available as open source, preferably
- prototype implemented with
 - Shibboleth 2.0 (Beta) IdP
 - JPam (JAAS-PAM bridge)
 - FreeAuth PAM module

The SWITCHaai picture we're familiar with

New with the 2.0 IdP:
a (modular)
authentication engine,
which includes three
authentication handlers

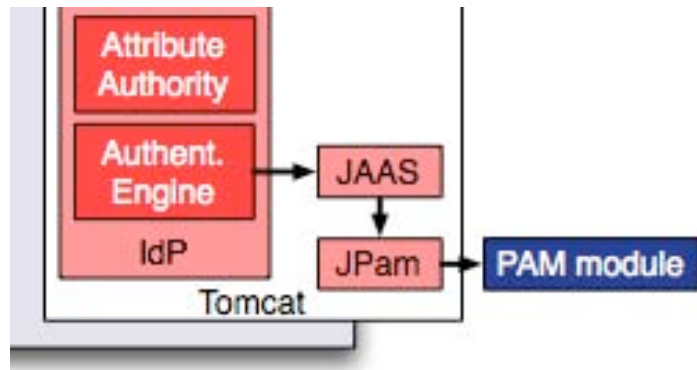


Authentication handlers included in 2.0

- *RemoteUserLoginHandler*: Web server or servlet container does authn and sets the remote user information (same as with Shibboleth 1.x)
- *UsernamePasswordLoginHandler*: based on **JAAS**, Sun's Java Authentication and Authorization Service, which uses pluggable login modules:
 - delivered with Sun's JVM: JndiLoginModule, KeyStoreLoginModule, Krb5LoginModule, LdapLoginModule, NTLoginModule, UnixLoginModule
 - JAAS is basically the Java version of PAM, but with a smaller set of publicly available login modules
- *IPAddressLoginHandler*

JPam comes in handy for us

- a Java/JAAS-PAM bridge (Pluggable Authentication Modules)
- allows using (native) PAM modules available on the platform where the IdP is running



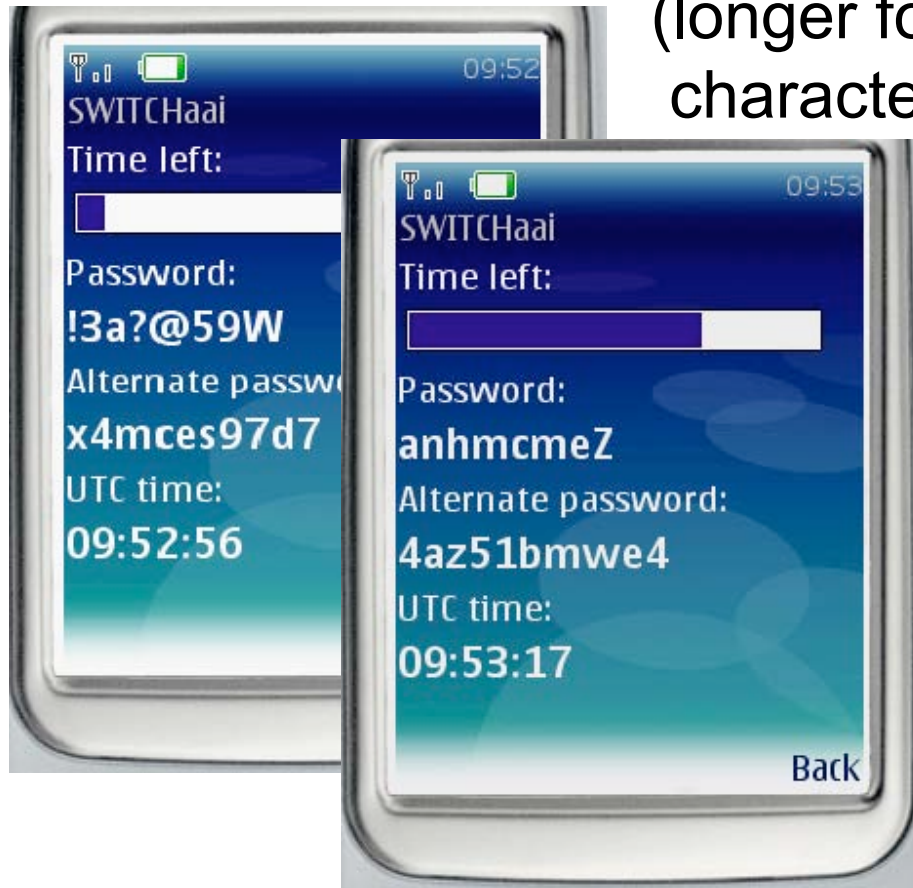
- cf. <http://www.kernel.org/pub/linux/libs/pam/modules.html> for a (large) list of available modules, supporting one-time passwords, hardware tokens (smartcards, SecurID, ...), SQL database authentication, RADIUS, IMAP etc. etc.

Free Auth (www.freeauth.org)

- project “was born out of the need for cheap/free One Time Password authentication”
- uses 8-character passwords which can be generated by any mobile phone with Java MIDlet support
- server software available either as a PAM module or as a PHP-based implementation (for Apache integration e.g.)
- shared secret must be configured on both the mobile phone and the server (for each user/account)
- passwords are time-based, so correct time settings are essential (allowable clock skew \pm 3 minutes)

Free Auth – how does it look like?

- password is valid for one minute, shown in two formats (longer form does not use special characters, easier to type on some keyboards)



Demo

- prototype implemented with the current Shibboleth 2.0 IdP beta version
- JPAM 1.1 (<http://jpam.sourceforge.net>)
- pam_freeauth (<http://www.freeauth.org>)
- MobileAuth 0.7 MIDlet (<http://mobileauth.solhost.org>)

Conclusion

- the *UsernamePasswordLoginHandler* in the Shibboleth 2.0 IdP allows a very flexible configuration of authentication methods
- either use a suitable JAAS login module, or add JPam as a bridge into the PAM world and pick the module of your choice
- strong authentication doesn't necessarily mean fiddling with X.509 certificates or buying expensive SecurID licenses

Enhanced Authorization

Use of authentication information and LoA at Service Provider



SWITCH

Serving Swiss Universities

Lukas Haemmerle

lukas.haemmerle@switch.ch

Authorization with AC on SP side

Shibboleth Service Provider 2.0:

- Request an authentication context
- Enforce authentication (again)
- Rules in .htaccess/httpd.conf or Shibboleth RequestMap
- Provides attributes:
 - Shib-Authentication-Method (how a user was authenticated)
 - Shib-AuthnContext-Class (e.g. in form of “Assurance Level”)

Use of these authorization attributes in the application itself!

Authentication Classes

Predefined in SAML2:

- urn:oasis:names:tc:SAML:2.0:ac:classes>PasswordProtectedTransport
- urn:oasis:names:tc:SAML:2.0:ac:classes:X509
- urn:oasis:names:tc:SAML:2.0:ac:classes:MobileTwoFactorUnregistered
- urn:oasis:names:tc:SAML:2.0:ac:classes:unspecified (default)

... or self-defined classes like “urn:mace:switch.ch:SWITCHaai:loa:3”

Example:

```
AuthType shibboleth
ShibRequireSession On
ShibRequestSetting forceAuthn true
ShibRequestSetting authnContextClassRef urn:[...]:X509
require homeOrganization unizh.ch
```