

Dive into PAS

Wichert Akkerman

PAS background

PAS is a *pluggable* Zope User Folder:

- ⇒ Most functionality defined in interfaces
- ⇒ Functionality is implemented by plugins
- ⇒ Plugins can be enabled per interface
- ⇒ PAS user folder is a wrapper around the plugins

PAS interfaces

- ⇒ IExtractionPlugin
 - ILoginPasswordExtractionPlugin
 - ILoginPasswordHostExtractionPlugin
- ⇒ IAuthenticationPlugin
- ⇒ IChallengePlugin
- ⇒ ICredentialsUpdatePlugin
- ⇒ ICredentialsResetPlugin
- ⇒ IUserAdderPlugin
- ⇒ IRoleAssignerPlugin

PAS interfaces, part 2

- ⇒ IUserFactoryPlugin
- ⇒ IAnonymousUserFactoryPlugin
- ⇒ IPropertiesPlugin
- ⇒ IGroupsPlugin
- ⇒ IRolesPlugin
- ⇒ IUpdatePlugin
- ⇒ IvalidationPlugin

PAS interfaces, part 3

- ⇒ IUserEnumerationPlugin
- ⇒ IGroupEnumerationPlugin
- ⇒ IRoleEnumerationPlugin
- ⇒ IRequestTypeSniffer
- ⇒ IchallengeProtocolChooser

PlonePAS interfaces

- ⇒ IUserIntrospection
- ⇒ ILocalRolesPlugin
- ⇒ IUserManagement
- ⇒ IMutablePropertiesPlugin
- ⇒ ISchemaMutablePropertiesPlugin
- ⇒ IMutablePropertySheet
 - ISchemaMutablePropertySheet
- ⇒ IGroupManagement
- ⇒ IGroupIntrospection
- ⇒ IGroupSpaceManagement
- ⇒ IGroupDataTool
- ⇒ IGroupTool

PAS concepts

- ⇒ A user is a complex object:
 - created using a special plugin type
 - properties are stored in property sheets
 - groups and roles determined by plugins
- ⇒ User id and login name are not the same thing!

Request validation

Zope calls upon a User Folder for authentication

- ⇒ Request validation:
 - extract all user ids
 - for each user id:
 - create a user object
 - try to authorise the request
- ⇒ If no user is authorised Zope issues a challenge

Validation: user extraction

- ⇒ use all IExtractionPlugin plugins to find all possible users
- ⇒ for each possible user:
 - test for emergency user
 - try to authenticate using all IAuthenticationPlugin plugins

This process can result in multiple possible user ids

Validation: user creation

- ⇒ create a user object using an IUserFactoryPlugin plugin
- ⇒ get property sheets using IPropertiesPlugin plugins
- ⇒ get groups using IGroupsPlugin plugins
- ⇒ get global roles using IRolesPlugin plugins

Validation: authorisation

- ➔ create Zope security manager and query it

Validation: challenging

A challenge asks a user to supply the right credentials.

The PAS procedure is:

- ➔ Use IChallengeProtocolChooser plugins to determine protocols
 - This can use IRequestTypeSniffer to determine the current request type
- ➔ Ask all IChallengePlugins to issue challenges iff they support the right protocol

Searching

Searching for users and groups is done with the IUserEnumerationPlugin and IGroupEnumerationPlugin plugins.

Results from all plugins are combined.

Examples!

Caveats

- ⇒ PAS eats exceptions
- ⇒ Users can disappear at any moment
- ⇒ User id and login name are not the same thing!
- ⇒ Exchanging information between plugins is hard
- ⇒ (Scalable) user management is hard

Enterprise issues

- ⇒ LDAP
- ⇒ Multiple user sources

More information?

- ⇒ Example code is available:
 - <https://code.wiggy.net/svn/opensource/zope/simplon.pasexamples>
 - <https://svn.plone.org/svn/collective/PASPlugins>
 - <https://svn.plone.org/svn/plone>
- ⇒ tutorial on plone.org later this year
- ⇒ zope-pas and plone-users lists