Campus Crusade for Christ International Grouper Project Page

Welcome to the Campus Crusade for Christ International Grouper page

Provisioning Consumer Info, based on ESB Consumer

https://spaces.at.internet2.edu/display/Grouper/Provisioning+Consumer

Deploying Grouper to Multiple Environments

We recently went live with our identity management system based on Grouper 1.6.3, and we made some modification to handle deploying Grouper to multiple environments.

Our goal was to allow the exact same Grouper JAR and WAR files to be deployed to any environment. We would then add environment-specific configuration files to a consistent location on each server, separated from the Grouper deployment location, so that fresh deployments would not overwrite these files. Any settings found in the environment-specific configuration files will override settings found in the default configuration files. This is similar in concept to how morphstring allows passwords to be externalized.

We identified six locations that needed to handle these externalized settings:

1) grouper.properties

2) grouper-loader.properties

3) grouper.hibernate.properties

4) sources.xml

5) log4j.properties

6) web.xml for the CAS contrib plugin for Grouper UI

The first 3 are handled by GrouperUtil.java.

The fourth is handled by SoruceManager.java (part of subject.jar).

Log4J was straight-forward using JVM options.

The UI's web.xml was a little trickier.

To achieve our goal, we made the following modifications to Grouper:

1) Create a custom sub-class of java.util.Properties which can load multiple properties files. One file contains default settings, and the other contains environment-specific "override" settings. Our custom properties file also has the ability to internally (and invisibly) handle encrypted property values using Jasypt (to achieve a similar goal as morphstring).

2) Modify GrouperUtil.java to use our custom Properties class instead of the default.

3) Modify SourceManager.java (part of subject.jar) to also load an overlay file, and changed the XML parsing slightly so that it properly merges the overlay settings before initializing sources.

4) We used the "-D" option on the JVM command line to pass environment-specific variables for use in log4j.properties.

5) Since we use CAS for authentication, we wrapped the CAS filter in a delegating filter.* The delegating filter creates a proxy of the filter init-params, allowing variable values to be pulled from external sources using the standard ${} syntax.

*Basic code for creating a delegating filter that allows externalizing of init-params can be found here:http://www.coderanch.com/t/79094/Websphere/environment-variable-referance-Web-xml