

OpenSSO QA Test Automation

Authentication Service Test development and execution

Prepared by Sridhar Enugula

1. Introduction

This document describes the architectural / implementation details primarily focusing on the authentication service. This document describes the following:

- Requirements
- How tests are organized
- Execution Details
- Tests/Features in framework
- Interpreting the report
- Debugging the test failures
- How to add new tests

2. Requirements

The primary requirements are:

- Authentication Service automated testing assumes that the user is familiar with the QA automation framework (please read “Overall Architecture of the Automation Framework” document for more details about the framework and workspace)
- Basic knowledge about the Federated Access Manager 8.0
- Federated Access Manager 8.0 samples war is deployed/configured on the supported container.
- Execution of the Windows NT (NT) authentication test cases on non-Windows platforms requires the Samba client (e.g. /usr/sfw/bin/smbclient). Create the directory <CONFIG_DIR>/<DEPLOY_URI>/bin and copy the smbclient binary to this directory. For Solaris, the Samba client smbclient may be found at /usr/sfw/bin/smbclient. To download the Samba client for Solaris/SPARC go to <ftp://ftp.sunfreeware.com/pub/freeware/sparc/10/samba-3.0.25a-sol10-sparc-local.gz>, for Solaris x86 go to <ftp://ftp.sunfreeware.com/pub/freeware/intel/10/samba-3.0.25a-sol10-x86-local.gz>, and for Linux it is in the RPM samba-client go to <http://rpmfind.net/linux/rpm2html/search.php?query=samba-client> to find this package.
- Execution of the JDBC authentication test cases requires the presence of the mysql-connector-java-3.1.14-bin.jar in the Federated Access Manager (fam.war) war file. This should be done as follows:
 - Download the mysql-connector-java-3.1.14-bin.jar from <http://dev.mysql.com/get/Downloads/Connector-J/mysql-connector-java-3.1.14.zip> from <http://mysql.he.net/>.
 - Create a directory in which the contents of the fam.war file will be exploded.
 - Navigate (cd) to the directory that was created.

- Use the jar command to explode the war file (e.g. jar xvf <path-to-fam.war>).
- Copy the mysql-connector-java-3.1.14-bin.jar to the WEB-INF/lib directory.
- Recreate the war file (e.g. jar cvf fam.war *).
- Undeploy the current FAM war file and deploy the updated fam.war file in the web container.
- Restart the web container instance on which the war was deployed.
- Execution of the Unix authentication test cases on Solaris/SPARC, Solaris x86, and Linux requires the amunixd daemon to be running on the system under test. This should be done as follows:
 - Download fam.zip for the FAM 8.0 build that will be tested to a directory (e.g. /usr/tmp/fam_zip) on the system under test.
 - Unzip fam.zip in a local directory on the system under test (i.e. unzip fam.zip).
 - Make the scripts in tools directory executable (i.e. chmod -R +x fam/tools/helpers).
 - Navigate to the helpers bin directory (i.e. cd fam/tools/helpers/bin).
 - Execute the amunixd shell script (i.e. ./amunixd).

3.How Tests are Organized

Authentication Service Framework provides several features. Currently test cases in the following areas are being automated :

- Functional Query Parameters Tests
- Service/Chain Based Authentication
- Redirection Tests
- Account Lockout
- Profile Attribute Tests
- Session Upgrade
- Application Authentication
- Post Authentication Processing
- User Authentication Attributes

Test Organization details:

1. Directory and file structure for authentication module features is laid out as below :

- The above listed features are packaged under the “authentication” package.
 <TEST_HOME>/source/com/sun/identity/qatest/authentication
- Each test feature implementation has the different java classes representing the feature.
 <TEST_HOME>/source/com/sun/identity/qatest/authentication/<feature_name>.jav
- The utility/common classes authentication module are at
 <TEST_HOME>/source/com/sun/identity/qatest/common/authentication
- All properties and configuration files for each feature are at
 <TEST_HOME>/resources
- Runtime classes and xml files will be generated under
 <TEST_HOME>/<TEST_SERVER1>/built/classes

2. Authentication tests are divided under the following different groups :
 - ff_ds : Embedded DS for User Management & Directory Server as the Service Management repository
 - ds_ds : Directory Server for User management & Directory Server as Service Management repository
 - ff_ds_sec : Embedded DS for User Management & Directory Server as the Service Management repository with keystore configured (secured)
 - ds_ds_sec: Directory Server for User management & Directory Server as Service Management repository with keystore configured(secured)

4. Execution details

4.1 How to execute tests :

This section describes how to execute authentication tests.

1. Deploy Federated Access Manager 8.0 and configure
2. Make sure you have the <TEST_HOME>/resources/Configurator-
<hostname>.properties that corresponds to the test server on which the tests are run. Please read, “Overall Architecture of the Automation Framework” document for more details.
3. If the QATest framework will be used to create a user datastore, edit the<TEST_HOME>/resources/config/configGlobalData.properties with the correct settings for the SMSGlobalDatastoreConfig1.* properties that will be used to configure the datastore.
4. For authentication module tests, each of the supported authentication module need to be configured properly for the tests to run. Each feature of the authentication service mentioned above needs module configuration. These modules are created at runtime based in the module configuration data provided. In order for the tests to execute properly, Modify the following properties files:
 For Functional Query Tests (AuthTest) modify the following file.
 <TEST_HOME>/xml/authentication/<module>TestData
 For all other tests, modify the following configuration data file
 <TEST_HOME>/resources/authentication/authenticationConfigData.properties
 For Eg: for ldap auth module config end user definately need to change the following:
 ldap.ipplanet-am-auth-ldap-server=opensso.java.net:389 =>Change to <userldapservice>
 ldap.ipplanet-am-auth-ldap-base-dn=dc=opensso,dc=javadc=net
 ldap.ipplanet-am-auth-ldap-bind-
 dn=cn=amldapuser,ou=DSAME,Users,dc=opensso,dc=javadc=net
 ldap.ipplanet-am-auth-ldap-bind-passwd=secret123
 Similarly, change for the all the other modules. NOTE: only the password and server related information need to be changed for appropriate attributes, all other attributes can be left as it is.(default). Windows NT and Unix authentication tests will be skipped on Windows based platforms.
5. Change the following parameters in <TEST_HOME>/build.properties file
 - QATEST_HOME
 - TEST_MODULE to “authentication”
 - EXECUTION_MODE to appropriate group as described in section # 3 above
 - REPORT_DIR to the location where you would like to store execution results.

4.2 The Execution details :

1. Run the following command to execute authentication module:
ant -DERSER_NAME1=<test_host_name> module
2. Each test feature is executed based on the feature related properties file. The properties for each test need to be modified appropriately based on the expected output and the appropriate parameters .(For more details refer to Test Details section of this document)
3. When each test/feature is run, first the required setup for that test is done, this is done in the setup method of the feature related java class. This is governed by BeforeClass annotation. This is basically prepares system for execution of the tests.
4. The @Test annotation in the java source files represents each test. Each test can be run randomly unless there is a Dependson annotation for that test or class.
5. @Afterclass annotation represents what should be done finally. In most cases cleanup is done after the tests have been executed on the system.
6. Execution and test results for each feature is dependent on module configuration(authenticationConfigData.properties) and test properties(<featurename.properties>)

5. Test Details :

5.1 Current Test Features/Classes/Data/Property files :

Test Class/Feature	Related PropertiesFiles	Test Related Module Configuration Property files	Description
AuthTest.java	AuthTest.properties	authenticationC onfigData	Please read #1 below
ChainTest.java	chainTest.properties	authenticationC onfigData	Please read #2 below
RedirectTest.java	RedirectTest.properties	authenticationC onfigData	Please read #3 below
AccountLockoutTest.java	AccountLockoutTest.properties	authenticationC onfigData	Please read #4 below
ProfileAttributesTest.java	ProfileAttributesTest.properties	authenticationC onfigData	Please read #5 below
SessionUpgrade.java	SessionUpgrade.properties	authenticationC onfigData	Please read #6 below
ApplicationAuthTest.java	ApplicationAuthTest.properties	NA	Please read #7 below
PostAuthenticationProcessingTest.java	PostAuthenticationProcessingTest.properties	authenticationC onfigData	Please read #8 below

UserAuthAttributeTest.java	UserAuthAttributeTest.properties	authenticationConfigData	Please read #9 below
----------------------------	----------------------------------	--------------------------	----------------------

- **# 1 Functional Query Parameters Tests :**

Before running the tests , the following need to be properly configured

- **authenticationConfigData :** This is file contains module configuration data and the values of each of these module specific data should be properly configured or creation of each module instance. This is located at <TEST_HOME>/resources/authentication.
- **TestProperties:** This is the test case related properties data file. This file is located under <TEST_HOME>/resources/authentication/AuthTest.properties. This file is read for all the different types of tests that are performed for each module for all the query parameters.
- **Adding Test Cases:** To add new Test cases, configuration data for each module and corresponding test case related data need to be added to these properties files mentioned above.

- **#2 Service/Chain based Authentication Tests:**

Chain/Service is one or more of authentication modules configured for each service and the user needs to provide credentials for each module that is configured for that service.

- **How to Run :** Before running the tests , the following need to be properly configured
- **authenticationConfigData :** This is file contains module configuration data and the values of each of these module specific data should be properly configured or creation of each module instance. This is located at <TEST_HOME>/resources/authentication.
- **TestProperties:** This is the test case related properties data file. This file is located under <TEST_HOME>/resources/authentication/ChainTest.properties. Each chain in this file corresponds to number of chains to be executed. When number of chains to be executed are increased. The Corresponding data for each chain should be provided such as (Chain1,Chain2,Chain3) for each of the chain what are the module involved , usernames/passwords for each chain login should be provided in this file.
- **Adding Test Cases:** To add new Test cases, configuration data for each module and corresponding test case related data need to be added to these properties files mentioned above.

- **#3 Redirection Tests :**

Authentication service supports Redirection URLs , The success and failure URL query parameters ("goto"/"gotoOnFail") takes the highest priority when the authentication is success or failure. This can be provided with any other query parameters such as module based authentication/User/realm etc....

- **How to Run :** Before running the tests , the following need to be properly configured
- **authenticationConfigData :** This is file contains module configuration data and the values of each of these module specific data should be properly configured or creation of each module instance. This is located at <TEST_HOME>/resources/authentication.
- **TestProperties:** This is the test case related properties data file. This file is located under <TEST_HOME>/resources/authentication/RedirectTest.properties. These tests can be run for different realms or sub-realms and the corresponding modules for these. Each Module should have the corresponding go and gotoOnFail URLs so when the tests are run for these realms/modules the corresponding URLs should be verified. The pass and fail messages for each of the goto should be properly changed.

- **Adding Test Cases:** To add new Test cases, configuration data for each module and corresponding test case related data need to be added to these properties files mentioned above #2
- **#4 AccountLockout :**
This feature enables the user to be locked after “n” number of attempts with invalid credentials. The user gets a warning before the actual lockout happened based on the lockout attributes configured for the lockout and warning.
 - **How to Run :** Before running the tests , the following need to be properly configured
 - **authenticationConfigData :** This is file contains module configuration data and the values of each of these module specific data should be properly configured or creation of each module instance. This is located at <TEST_HOME>/resources/authentication.
 - **TestProperties:** This is the test case related properties data file. This file is located under <TEST_HOME>/resources/authentication/AccountLockoutTest.properties. The number of attempts the system should be configured for lockout and the warning should be provided in this properties file. The module on which the test should be performed should be provided, appropriate user accounts for lockout and warning should be provided.
 - **Adding Test Cases:** To add new Test cases, configuration data for each module and corresponding test case related data need to be added to these properties files mentioned above #2.
- **#5 Profile Attribute Tests :** Authentication service supports several user profile attributes that can be used post authentication process. When the user successfully authenticated the user profile will be fetched from the configured datastore based on this attribute.
 - **How to Run :** Before running the tests , the following need to be properly configured
 - **authenticationConfigData :** This is file contains module configuration data and the values of each of these module specific data should be properly configured or creation of each module instance. This is located at <TEST_HOME>/resources/authentication.
 - **TestProperties:** This is the test case related properties data file. This file is located under <TEST_HOME>/resources/authentication/ProfileAttributesTest.properties. Each attribute for the user profile need to be provided with the corresponding data such as users to login, password, module and expected passmessages.
 - **Adding Test Cases:** To add new Test cases, configuration data for each module and corresponding test case related data need to be added to these properties files mentioned above #2.
- **#6 Session Upgrade Tests :** Authentication service supports session upgrade, In session upgrade the user authenticates to module1 and module2 the session is upgraded with the AuthType as module1|module2, the correct AuthLevel, and the proper session ID.
 - **How to Run :** Before running the tests , the following need to be properly configured
 - **authenticationConfigData :** This is file contains module configuration data and the values of each of these module specific data should be properly configured or creation of each module instance. This is located at <TEST_HOME>/resources/authentication.
 - **TestProperties:** This is the test case related properties data file. This file is located under <TEST_HOME>/resources/authentication/SessionUpgrade.properties. Each attribute for need to be provided with the corresponding data such as users to login, password, module and expected passmessages.
 - **Adding Test Cases:** Configuration data for each module and corresponding test case related data need to be added to these properties files mentioned above #2.

- **#7 Application Authentication Tests:** When an agent identity authenticates using application authentication, the user should receive a non-expiry SSOToken.
 - The following properties files should be configured to run these tests:
 - **Test Properties:** The path of this properties file is <TEST_HOME>/resources/authentication/ApplicationAuthTest.properties. There are two properties in this file, the agent identity name and the agent identity password.
 - **Adding Test Cases:** A method performing each new test case should be added to the ApplicationAuthTest.java. Any new properties used in the new method should be added to the test properties file.
- **#8 Post Authentication Processing Tests:** These tests use the “User Attribute Mapping to Session Attributes” feature of the authentication service. In one test, a variety of user profile attributes are stored in an authenticated user's SSOToken and the values are retrieved from the user's SSOToken using the SSOToken.getProperty method. In the other tests, a variety of user profile attributes are stored in an authenticated user's SSOToken and an attempt is made to modify the SSOToken property which should result in a SSOException.
 - The following properties files should be configured to run these tests:
 - **authenticationConfigData :** This is file contains module configuration data and the values of each of these module specific data should be properly configured or creation of each module instance. This is located at <TEST_HOME>/resources/authentication.
 - **Test Properties:** The path of this properties file is <TEST_HOME>/resources/authentication/PostAuthProcessingTest.properties. This file contains the username and password a user that will be authenticated, a list of attribute name/value pairs that will be stored in the user's SSOToken, and properties for the test name and description for each test.
 - **Adding Test Cases:** A method performing each new test case should be added to the PostAuthProcessingTest.java. Any new properties used in the new method should be added to the test properties file.
- **#9 User Authentication Attribute Tests:** These tests set various combinations of authentication related user attributes such as inetuserstatus, iplanet-am-user-login-status, iplanet-am-user-account-life, and nsaccountlock. Then with those particular attributes set the tests verify whether the user is able to authenticate or not.
 - The following properties files should be configured to run these tests:
 - **authenticationConfigData :** This is file contains module configuration data and the values of each of these module specific data should be properly configured or creation of each module instance. This is located at <TEST_HOME>/resources/authentication.
 - **Test Properties:** The path of this properties file is <TEST_HOME>/resources/authentication/UserAuthAttributeTest.properties. This file contains the authentication module that will be used for authentication and the test case specific information (i.e. the user name, password, the user attribute names/values that should be set, the message that should be appear after authentication is performed, and the description of the test.
 - **Adding Test Cases:** The UserAuthAttributeTest.properties file should be updated with the properties listed in the above “Test Properties” item.

NOTE : The corresponding properties for each test need to be read properly and changes need to be made as suggested in the property files such as For Eg: If the “goto” URL is changed for the Redirect Tests, the corresponding passmessage need to be changed similarly for all other tests as mentioned in detail above.

5.2 Common Files

There are multiple common files that are used by authentication module.

1. AuthenticationCommon.java
This class contains common methods to perform the Zero Page Login and verification postive and negative tests for the functional query tests
2. AuthTestConfigUtil.java
This utility class contains common methods to create the authentication module instances, services,users,realms etc
3. AuthTestsValidator.java
This class contains common methods to validate authentication scenarios.
4. CreateTestXML.java
This class contains common methods to create test based xml files during run time.This generated xml file is used/passed to the WebTest which verifies the test.WebTest is the utility that takes the username and password form data and submits the form check if the result is the expected result, all the user/password/result is passed as an xml file. This file is generated by this class for appropriate test : Here below is the sample of generated xml file for a test
<url href="http://teal16.red.ipplanet.com:80/openfm?module=ad-1">
<form name="Login" IDButton="" >
<input name="IDToken1" value="administrator" />
<input name="IDToken2" value="secret12" />
<result text="administrator" />
</form>
</url>

6. Interpreting Authentication Automated Test Results

After execution of authentication test module, the results will be located under

<REPORT_DIR>/<TEST_HOST_NAME>/<EXECUTION_MODE>/EXECUTION_DATE_TIME> , browse index.html in this directory.The overall results will be displayed. Click on the link for <EXECUTION_MODE>-authentication fopr the detailed test report for that execution group, Evaluate the number of tests passed/failed/skipped.More details on reading the test results , please refer to overall automation framework document.

7. Debugging the automated tests

Debugging the failures is very critical thing in automation, Here are several ways to debug the authentication failures:

- Check the test execution log named “logs”. This detailed log containing the execution logging

output for each test case can be found in the file

<REPORT_DIR>/<TEST_HOST_NAME>/<EXECUTION_MODE>/<EXECUTION_DATE_
TIME>/logs.

- If the test case is verified/executed by the WebTest as mentioned above check the appropriate run time xml files generated under <TEST_HOME>/<SERVER_NAME1>/built/classes directory.
- Check the server logs and find the root cause of the failures.
- Execute the test case manually to differentiate if it is a product or test bug or change in the product that causing the failure.
- If the Windows NT (NT) authentication is failing, please review the fourth requirement in section 2 “Requirements”.
- If the JDBC related authentication is failing, please review the fifth requirement in section 2 “Requirements”.