# ABAC Web Service Installation Manual

*Document ID: PROTOGENI-ABAC-INSTALL-1.0*

*Version: 1.0*

*Prepared By: Jay Jacobs*

## Contents

## 1. Overview

This document describes how to run and install the ProtoGENI ABAC integration. The ABAC web service is a stand-alone web service for attribute-base access control. The initial integration is implemented for use with the ProtoGENI reference component manager implementation; however, usage is not limited to core framework usage. Client applications are supplied for Java and perl.

### 1.1 Reference

The ABAC web service distribution can be found at http://groups.geni.net/geni/wiki/ABAC. The distribution contains ABAC, an umodified reference-cm-2.0.2, and a modified reference-cm-2.0.2a.

Details and code for the ProtoGENI  reference component manager can be found at http://www.protogeni.net/trac/protogeni/wiki/ReferenceCM . The current ABAC integration supports version 2.0.2.

The reference component manager is tested using the standard ProtoGENI test scripts. The details for testing and the test scripts are available at http://www.protogeni.net/trac/protogeni/wiki/TestScripts.

### 1.2 System Dependencies

The ABAC web service can be run on any machine with a recent (> 1.4) of the Java virtual machine. The JDK will be necessary for building the ABAC software as described in section 4.0. The prerequisites include: JDK (1.6.), tomcat (5.5), axis (1.x), Bouncy Castle (1.38), junit(1.0), ant(1.4), jython(2.4), and m2crypto. The emulab experiment contains a reference-cm experiment image and the ABAC-WS requires the following packages:

A sample of installation commands for the reference-cm are included below:

```
sudo yum -y install gcc
sudo yum -y install httpd
sudo yum -y install libvirt
sudo yum -y install mod_ssl
sudo yum -y install mysql-server
sudo yum -y install perl-Crypt-SSLeay
sudo yum -y install perl-Frontier-RPC
sudo yum -y install perl-RPC-XML
sudo yum -y install perl-suidperl
sudo yum -y install perl-TimeDate
sudo yum -y install perl-XML-LibXML
sudo yum -y install perl-XML-Simple
sudo yum -y install qemu
sudo yum -y install xmlsec1
sudo yum -y install xmlsec1-openssl
```

The reference-cm client scripts are written in python and require the m2crypto cryptographic library which can be installed as follows:

```
sudo yum –y install m2crypto
```

A sample of installation commands for ABAC are included below:

```
sudo yum -y install java-1.6.0-openjdk
sudo yum -y install java-1.6.0-openjdk-devel
sudo yum -y install bcprov
sudo yum -y install junit
sudo yum -y install jython
sudo yum -y install ant
```

The following packages are also useful but not necessary.

```
sudo yum -y install lynx
sudo yum -y install xorg-x11-fonts-Type1
sudo yum -y install graphviz
```

Lynx is a command-line browser which is useful for checking axis/tomcat status when port forwarding is not available. The type 1 fonts and graphviz package are needed for the negotiation visualization software

## 1.3 Building and Deploying ABAC

The protogeni-abac emulab experiment has a precompiled version of ABAC and the necessary dependencies install. To build and deploy ABAC from source, please use the following instructions. Apache tomcat is needed—an archive is available in `/proj/geni/tarfiles/ABAC`. The default tomcat location is `/usr/local/tomcat5.5`

1. Set the JAVA_HOME environment variable

```
export JAVA_HOME=/usr/local/jdk1.6.0
```

2. Download and unpack the source archive from at [http://groups.geni.net/geni/wiki/ABAC](http://groups.geni.net/geni/wiki/ABAC):

```
tar jxvf abac-src.tar.bz2
```

3. Build the main ABAC library

```
ant compile
```

4. Build the wsdl classes and create ws-abac.jar

```
ant ws-jar      # compiles wsdl classes and deployment jar
```

Note: The wsdl target should not overwrite FeddABACBindingImpl.java and you should see the following output:

```
[java] FeddABACBindingImpl.java already exists, WSDL2Java will not
overwrite it.
```

5. As root deploy the jar and deployment descriptor

```
        sudo ant deploy
```

Note: server-config.wsdd appears in the directory from which
ant is run. This means that if you invoke ant from the wsdl directory, it will
deploy the jar file but not the wsdd file and the target will fail.

6. Reload the axis webapp from a page similar to: [https://localhost:8443/manager/html](https://localhost:8443/manager/html), where localhost is the machine name where tomcat is installed and ABAC deployed.

Click reload on the right. Click the name axis (or the axis deployment directory). For development changes restart the tomcat servlet engine using the following:

```
export JAVA_HOME=/usr/lib/jvm/java-openjdk
sudo -E /usr/local/tomcat5.5/bin/startup.sh
```

For convenience, startup and restart scripts are available in the test directory.

7. Add a the bouncy castle provider to JRE's trusted path. On emulab systems, the `bcprov.jar` file need to be symbolically linked into `jre/lib/ext` as follows:

```
cd $JAVA_HOME/jre/lib/ext
sudo ln -s /usr/share/java/bcprov.jar .
```

Warning: The bouncy castle provider must be linked into jre/lib/ext or the webservice will generate a server error. Some distributions may do this by default.

## 2. Policy Generation

ABAC policies are a combination of X.509v3 identity certificates and X.509v2 attribute certificates. Policy generation is still in development. For credential issuers that do not have an emulab generated key pair and public certificate. A script is supplied to automate the process of credential creation, so the user can reuse existing PKI or customized keystores for crednential generation process. The next section describes the process in detail.

## 2.1 Policy Creation

ABAC policy is implemented as a set of identity certificates (X.509v3 identity certificates) and credential certificates (X.509v2 attribute certificates). Identity certificates are identical in structure to the X.509 certificates already in use by ProtoGENI and emulab and are required for verifying the credential certificates. The private keys associated with credential issuer should be secured once the credentials have been issuer. In special cases, it may be desirable to destroy the private key after credential creation (e.g. once-time use).

### 2.1.1 Certificate Generation

For credential issuers who do not have an emulab generated key pair and public certificate, key-gen.sh is a script to generate self-signed 2048-bit X.509 certificates and private key in PEM format:

```
key-gen.sh <issuer-name> <passphrase> <distinguished_name>
```

where distinguished name should be of a form similar to:

```
DN="/C=US/ST=California/O=ProtoGENI\
      /CN=common-name /emailAddress=issuer@myorg.org"
```

For examples of distinguished names, please refer to RFC2459, which is available at http://www.ietf.org/rfc/rfc2459.txt.

Java's keytool requires private keys unlocked and in DER format, so the script will also convert the private keys into DER format. Modify the following example for each issuer:

```
 ./create-pair.sh geni <passphrase> \
   "/C=US/ST=California/O=NSF GENI/CN=geni/emailAddress=geni@geni.org"
```

When using Emulab generated certificates and private keys, only the private keys need to be converted. PEM-encoded public key certificates do not require any special security. The addpair.py script can be used to add a key-pairs to their respective keystores for credential generation.

```
./addpair.py –a geni –k public.jks –s signatory.jks –p <passphrase>
```

Only the –a option is mandatory. A –h help option will print usage information.

### 2.1.2 Credential Generation

Credential generation can scripted using the jython CertFactory a priori or a run-time using the the createcredential jython script as follows:

```
./createcredential.py  -o <output_file> -p <keystore-passphrase> \
      -k <keystore>  -c <credential_text>
```

The output files is a base64 encoded X.509 v2 attribute certificate, signed by the issuer alias specified.

### 2.1.3 Credential Verification

Generated credentials can be verified using the ctest.sh with the full path name of the base-64 encoded ABAC credential. Validation is done against the deployed public keystore.

```
./ctest.sh geni-user.b64
```

### 2.1.4 Policy Generation Example

Policy generation is a two-stage process and can be automated using the createpolicy.py script, which converts a series of $RT_0$ credentials into directives for creating identity and attribute certificates using the tools in the previous sections. A sample configuration for ProtoGENI should look similar to the following:

```
geni.all <-- geni.user
geni.user <-- geni.admin
geni.admin <-- geni.creator
geni.creator <-- urn:publicid:IDN+emulab.net+user+jjacobs
geni.creator <-- urn:[GeniUser: utahemulab.jjacobs, IDX: 1]
```

Running the createpolicy.py script will generate something similar to the following:

```
# create the key pair for geni
./create-pair.sh geni importkey \
 "/C=US/ST=California/O=GENI/CN=geni/emailAddress=geni@geni.org"
# add keys for geni into default keystores
./addpair.py -v -a geni
# Credential generation follows
./createcredential.py -o geni-all-geni-user.b64 \
      -c "geni.all <-- geni.user"
./createcredential.py -o geni-user-geni-admin.b64 \
      -c "geni.user <-- geni.admin"
./createcredential.py -o geni-admin-geni-creator.b64 \
      -c "geni.admin <-- geni.creator"
./createcredential.py -o geni-creator-urn-jjacobs.b64 \
      -c "geni.creator <-- urn:publicid:IDN+emulab.net+user+jjacobs"
```

```
./createcredential.py -o geni-creator-hrn-jjacobs.b64 \
     -c "geni.creator <-- urn:[GeniUser: utahemulab.jjacobs, IDX: 1]"
```

## 2.2 ABAC API Description

The ABAC web services have java, perl, and python clients. For testing the web service sample scripts for java clients are provided. Use the following examples for creating a negotiation context and performing an access request against it:

### 2.2.1 Context Creation

A trust negotiation needs to be performed within a specific negotiation context. Use the create.sh script to create a context with a specified alphanumeric id.

```
create.sh <context_id>
```

### 2.2.2 Credential Population

The X.509 attribute certificates are loaded into the negotiation context using the add.sh script.

```
add.sh <context_id> <x509v2_attribute_cert_file>
```

### 2.2.3 Access Request

An access request initiates the a trust-target negation within a given negotiation context. The request returns a Boolean result on whether `issuer.role_name` can be matched against the `subject`.

```
trust.sh <context_id> <issuer> <role_name> <subject>
```

## 2.3 Example Usage

The following example can be used to set up a negotiation context and load it with a set of ABAC credentials. The first four commands create a context and then add generic ProtoGENI policy:

```
./create.sh geni

./add.sh geni crypto/p-geni-admin.b64
./add.sh geni crypto/p-geni-all.b64
./add.sh geni crypto/p-geni-user.b64
```

Trying to create a sliver will fail at the point because the end-user credentials have not been added. The following add end-user credentials.

```
./add.sh geni crypto/jjacobs-creator-hrn.b64
./add.sh geni crypto/alefiya-creator-hrn.b64
```

The ABAC perl client will perform and access request as follows.

```
./trust.sh geni geni creator \
     “urn:[GeniUser: utahemulab.jjacobs, IDX: 1]”
```

Examples of the reference component manager are in the next section.

## 3. Reference Component Manager Tests

The reference component manager comes with test scripts in the test directory of the distribution. This section explains example usage of the defauls python scripts.The examples below assume the reference component manager has been installed at the machine host.protogeni-abac.geni.emulab.net. Using the –d option on the script will print out extra debugging information include the URIs which indicate explicitly which core framework component is called and its location.

### 3.1.1 GetVersion

To verify the that the reference component manager is operating properly, use the getversion.py script:

```
./getversion.py -d \
-m https://host.protogeni-abac.geni.emulab.net:443/protogeni/xmlrpc cm
```

The getversion.py script executed above will connect to a component manager at the specified URL and return the version information. This is a useful command to verify that the reference-cm is installed and running properly. If you receive an error check that the apache httpd is running as follows:

```
sudo /etc/init.d/httpd status
```

The daemon can be started using the start argument:

```
sudo /etc/init.d/httpd start
```

### 3.1.2 Discover

Once the reference-cm operation has been confirmed by using getversion.py, the discover.py script can be used to discover the rescources provided by the reference component manager.

```
./discover.py \
      -m https://host.protogeni-abac.geni.emulab.net:443/protogeni/xmlrpc \
      > advert.rspec
```

The discover.py command queries the component manager for available resources and returns them in the form of an RSPEC.

### 3.1.3 CreateSliver

In order to create a new rspec file in my.rspec similar to the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<rspec xmlns="http://www.protogeni.net/resources/rspec/0.1"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.protogeni.net/resources/rspec/0.1
http://www.protogeni.net/resources/rspec/0.1/request.xsd"
       type="request">
  <node component_uuid="urn:publicid:IDN+host.protogeni-
abac.geni.emulab.net+node+pc1"
        component_manager_uuid="urn:publicid:IDN+host.protogeni-
abac.geni.emulab.net+authority+cm"
        virtual_id="pc1"
        virtualization_type="emulab-vnode"
        exclusive="1">
    <node_type type_name="pc" type_slots="1"/>
    <interface virtual_id="control"/>
  </node>
</rspec>
```

Using one or more nodes listed in the discovery rspec advertised in the results of the previous section, change the node and cm references as necessary. Then use the following command to create the sliver:

```
./createsliver.py -d -n abac \
    -m https://host.protogeni-abac.geni.emulab.net:443/protogeni/xmlrpc/cm \
     my.rspec
```

### 3.1.4 Delete Slice

To delete the sliver, reuse the rspec file for invoking the createsliver script.

```
./deleteslice.py -d -n abac \
    -m https://host.protogeni-abac.geni.emulab.net:443/protogeni/xmlrpc/cm
```