NERC Data Grid Security

Installation Guide

Version 0.6

Document Log

Version Number	Date	Comment
0.1	04/11/05	First Draft
0.2	21/02//06	Draft for installation at NOCS
0.3	07/04/06	Updates following installation at NOCS
0.4	25/07/06	Include deployment model and details about SysV style init scripts for web services.
0.5	16/01/07	Instructions for installation of python packages and associated C library dependencies from source and corrections for MyProxy installation. Installation instructions apply to NDG-Security Post Alpha release 0.72.
0.6	17/08/07	Updated for NDG Beta release. • Installation of python packages is now via distutils eggs. • Python services use Twisted.

Contents

1. REFERENCES

1. http://grid.ncsa.uiuc.edu/myproxy/ - NCSA MyProxy site

2. http://grid.ncsa.uiuc.edu/myproxy/fromscratch.html - NCSA MyProxy installation instructions

3. http://www-unix.globus.org/toolkit/docs/4.0/security/ - Globus 4.0 and Security

4. NDG Security - Security Measures for Installation [v0.2, 7 September 2005], http://bscw.badc.rl.ac.uk/bscw/bscw.cgi/d77103/NDG%20Security%20-%20Security%20Measures%20for%20Installation

2.INTRODUCTION

This is a guide for system administrators and developers deploying NDG security at a data centre.

2.1Pre-requisites

- For NDG Security Web Services: a host running RedHat Enterprise AS4 or later is recommended. Other Linux distributions may also be suitable.

- For MyProxy: a separate host machine (See MyProxy for details of operating systems supported). The host must be secure: if possible a dedicated machine with minimal other services running on it. It should be kept up to date with patches and system logs monitored regularly.

- MyProxy and Security web services hosts must be configured to link with an NTP server to enable clocks to be synchronised with security services running at other NDG sites.

- Access to a web server if security for web based applications is required. The web server must be able to be configured to support HTTPS.

- [MySQL 3.23 or greater or Postgres – these are optional and are required for the NDG CredentialRepository only]

- Python 2.4 or later

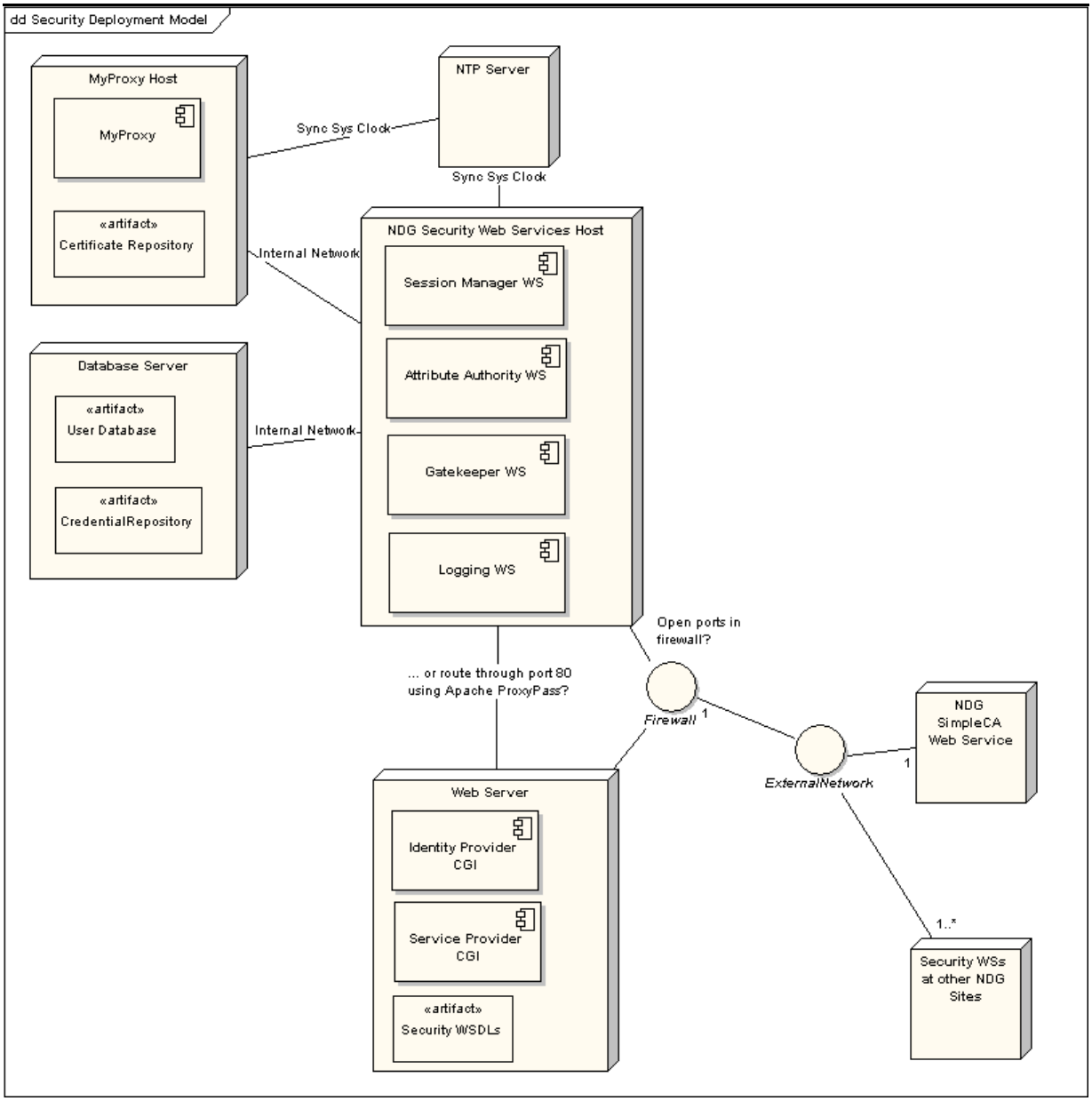
- Python distutils utility

- OpenSSL is required at version 0.9.8 or greater

Also note document NDG *Security - Security Measures for Installation* (see Ref 1above).

2.2Deployment Model

The following diagram gives an example deployment configuration for NDG security services.



All services are positioned behind the firewall. MyProxy is installed on a dedicated machine in order to make its repository as secure as possible. Connections to MyProxy may be made from the Session Manager web service only from within the internal network.

In the above, security web services are run together on the same host but this does not have to be the case. They can be run on separate servers. Similarly, the web server is on a separate host but could be run on the same machine as the web services if it was felt to be appropriate.

In the above diagram Attribute Authority accesses a user database. It is assumed that the target site has a database to store user and user role/access right information. This information needn't be stored by means of a database and could be represented in some other way. It is for the data provider to decide. Similarly, the Session Manager web service interfaces with a Credential Repository. This is a database in the above but could be some other kind of permanent store.

Databases are on a separate server to the web services host. Web services access the databases over the internal network.

Finally, the web services have ports exposed in some way through the firewall to enable communication with other NDG security web services at other sites.

2.3Software Installation Components

Python software is package using distutils eggs. These are divided into separate components to suit the particular installation required:

- ndg_security_server – components required to run services
- ndg_security_common – components required by both server and common eggs
- ndg_security_client – components for building clients to NDG security services. For example, a data provider's web application server would these to enable the securing of access to resources or an organisation's Identity provider would need these to authenticate and allocate authorisation attributes to users.
- ndg_security_test – unit tests for all components
- ndg_security – install all: client, server and common components

Eggs rely on the distutils easy_install command to manage installation but NDG security uses an additional script ndg_security_install.py to install eggs and carry out the additional installation tasks to correctly configure the software.

The following additional packages are required:

- Globus MyProxy 4.0.1 (or later) – source installer tar ball may be downloaded from the Globus site (http://www.globus.org/toolkit/downloads/4.0.1/)
- NDG SimpleCA client package tar ball – configures target machine to trust the NDG CA.

These two packages should be installed on the target host for MyProxy.

3.INSTALLATION

This section is divided into the Python installation and MyProxy. Note that you will almost certainly wish to install MyProxy on a separate secure server to the other Python based security services.

3.1Python Packages

Log in to the target host as `root`. Change to a suitable directory to hold temporary installation files.

3.1.1distutils

The first step is to install Python distutils, the package that enables the use of Python eggs. Download the distutils bootstrap script:

```
$ wget http://peak.telecommunity.com/dist/ez_setup.py
```

You may need to set the environment for a http proxy at your site. For example,

```
$ export http_proxy=http://yourproxyurl.com:8080
```

Run the bootstrap script. Make sure to use the correct version of python in your system path. Some systems may have multiple python versions installed:

```
$ python ez_setup.py
```

Once completed, you can delete ez_setup.py.

3.1.2NDG Security Packages

NDG security uses a wrapper to distutils easy_install to enable custom installation steps to be correctly carried out. Download the script from the NDG distribution site:

```
$ wget http://ndg.nerc.ac.uk/dist/ndg-security-install.py
```

Now carry out the installation of the NDG security python packages:

```
$ python ./ndg-security-install.py -a
```

The script options can be checked using the –h option. –a selects all packages for installation. If there are problems with the installation, see the Troubleshooting Guide in the Appendices section 4.5.

3.2NDG Web Services Configuration

3.2.1NDG Security System Configuration Directory

Properties files set the configuration settings for NDG security *server side* settings. Templates for these are contained within the ndg_security_server installed in your python distribution's site-packages directory. A future version of the ndg-security-install.py script will extract these and install at a suitable location on the file system. For the moment though, this is a manual process.

Create a configuration area under your servers /etc directory:

```
$ mkdir /etc/ndg
$ mkdir /etc/ndg/security
```

/etc/ndg/security is recognised by the Python security software by the NDGSEC_DIR environment variable. This variable can be set in the environment of the user account used to run the security services or can be set in the init scripts used to automatically start up the services from server boot up (See sections 3.3.3).

Locate the ndg_security_server egg and copy its conf/ directory into the configuration area. For example if you are using python installed in /usr/local then the conf/ directory will be in:

```
/usr/local/lib/python<python version num>/site-
packages/ndg_security_server-<version info>.egg/ndg/security/server/conf
```

Copy as follows:

```
$ cp /usr/local/lib/python<python version num>/site-packages/ndg_security_server-<version
info>.egg/ndg/security/server/conf /etc/ndg/security
```

The conf/ directory will containing Session Manager and Attribute Authority properties XML files, certs/ directory for storing certificates and attCert/ directory for storing Attribute Certificates issued by the Attribute Authority.

3.2.2Certificate Generation

The Session Manager and Attribute Authority web services require individual X.509 certificates as a means to identify them in the various interactions required for user registration, authentication and authorisation. These may be created by similar means to the host certificate creation.

Change directory to `$NDGSEC_DIR/conf/certs`. The certificates will be stored here. Make a new private key and certificate request for the Session Manager:

```
$ openssl genrsa –out sm-key.pem 2048
$ openssl req –new –key sm-key.pem –out sm.csr
```

The private key may be password protected if required by adding the –des3 option to the genrsa command. Type in a password when prompted. The req command will prompt you for the components of the Distinguished Name for the new certificate. When prompted for the Common Name, enter 'SessionManager'. The other fields can be set as required but by convention for NDG, the Organisation field has been set to NDG and the Organisation Unit to the individual data provider name e.g. BADC. All other fields have been omitted. You can skip individual fields by enter '.' When prompted.

Forward the request file to the NDG CA. The CA will issue a certificate file. Copy this file as `$NDGSEC_DIR/conf/certs/sm-cert.pem` . The request file can be deleted once a certificate has been obtained from the CA.

Repeat this process for the Attribute Authority, selecting 'AttributeAuthority' for the Common Name.

```
$ openssl genrsa –out aa-key.pem 2048
$ openssl req –new –key aa-key.pem –out aa.csr
```

It is recommended that the Session Manager is run over https to keep user login credentials secured. A server certificate and key will be required in addition to enable this. These can be added to the `$NDGSEC_DIR/conf/certs` directory and can be referenced by the Session Manager's properties file.

A copy of the NDG Certificate Authority's X.509 certificate is also required. Obtain this from the NDG CA administrator and copy it into the `$NDGSEC_DIR/conf/certs` directory.

3.3Session Manager Configuration

Configuration parameters may be set via a properties file. In addition, the SessionManager can optionally make use of a Credential Repository database. This enables the credentials that users acquire during a session to be stored so that they may be retrieved. When installed, the default configuration set in the Session Manager properties file is to *not* use a Credential Repository. If this is the case, skip this section.

3.3.1Session Manager Credential Repository

Create the Credential Repository database. In the example below a MySQL database is assumed. Notes on installing MySQL are given in the Appendices section 4.1.

```
$ mysql –u root –p
mysql> create database ndgCredRepos;
```

Make use of the script initCredReposDb.py to create the tables. As the globus user, run the script. Enter the password for the ndgUser account when prompted and type yes to confirm creation of the tables:

```
$ cd $NDGSEC_DIR/bin
$ ./initCredReposDb.py –u root
Database password:
Are you sure you want to initialise the database tables? (yes/no) yes
Tables created
```

To check that the tables have been created, restart the database client:

```
$ mysql –u root –p –D ndgCredRepos
mysql> show tables;
+------------------------+
| Tables_in_ndgCredRepos |
+------------------------+
| UserCredential         |
| UserID                 |
+------------------------+
2 rows in set (0.00 sec)
```

A separate account should be created for the Session Manager to access the database. It should have sufficient permissions to be able to read and write records. For details of how to create an account in MySQL see the Appendices section 4.1.9.

3.3.2Session Manager Properties File Settings

Edit sessionMgrProperties.xml in $NDGSEC_DIR/conf and modify the default settings:

```
<?xml version="1.0" encoding="utf-8"?>
<sessMgrProp>
    <portNum></portNum>
    <useSSL>Yes</useSSL> <!-- leave blank to use http -->
    <sslCertFile>$NDGSEC_DIR/conf/certs/server-cert.pem</sslCertFile>
    <sslKeyFile>>$NDGSEC_DIR/conf/certs/server-key.pem </sslKeyFile>
    <!--
    PKI settings for signature of outbound SOAP messages
    -->
    <useSignatureHandler>Yes</useSignatureHandler> <!-- leave blank for no signature
-->
    <certFile>>$NDGSEC_DIR/conf/certs/sm-cert.pem</certFile>
    <keyFile>>$NDGSEC_DIR/conf/certs/server-key.pem</keyFile>
    <keyPwd></keyPwd>
    <caCertFile>>$NDGSEC_DIR/conf/certs/cacert.pem</caCertFile>
    <!--
    Set the certificate used to verify the signature of messages from the
    client.  This can usually be left blank since the client is expected to
    include the cert with the signature in the inbound SOAP message
    -->
    <clntCertFile></clntCertFile>
    <sessMgrEncrKey></sessMgrEncrKey>
    <sessMgrURI></sessMgrURI>
    <cookieDomain></cookieDomain>
```

```
    <myProxyProp>
             <!--
             Delete this element and take setting from MYPROXY_SERVER environment
             variable if required
             -->
             <hostname>ENTER THE FULLY QUALIFIED HOSTNAME OF THE SERVER</hostname>
             <!--
             Delete this element to take default setting 7512 or read
             MYPROXY_SERVER_PORT setting
             -->
             <port>7512</port>
             <!--
             Useful if hostname and certificate CN don't match correctly.  Globus
             host DN is set to "host/<fqdn>".  Delete this element and set from
             MYPROXY_SERVER_DN environment variable if prefered
             <serverDN></serverDN>
             -->
             <!--
             Set "host/" prefix to host cert CN as is default with globus
             -->
             <serverCNprefix>host/</serverCNprefix>
             <!--
             This directory path is used to locate the OpenSSL configuration file

             The settings are used to set up the defaults for the Distinguished Name
of
             the new proxy cert. issued

             GLOBUS_LOCATION or GRID_SECURITY_DIR environment variables may be used
             but the settings can be independent of any Globus installation

             -->
             <girdSecurityDir>$NDGSEC_DIR/conf</gridSecurityDir>
             <openSSLConfFileName>openssl.conf</openSSLConfFileName>
             <tmpDir>/tmp</tmpDir>
             <!--
                    Limit on maximum lifetime any proxy certificate can have -
                    specified when a certificate is first created by store() method
             -->
             <proxyCertMaxLifetime>24</proxyCertMaxLifetime> <!-- in hours -->
             <!--
                    Life time of a proxy certificate when issued from the Proxy
Server
                    with getDelegation() method
                    -->
             <proxyCertLifetime>8</proxyCertLifetime> <!-- in hours -->
             <caCertFile>$NDGSEC_DIR/conf/certs/cacert.pem</caCertFile>
       </myProxyProp>
       <simpleCACltProp>
           <uri></uri>
         <xmlSigKeyFile></xmlSigKeyFile>
         <xmlSigCertFile></xmlSigCertFile>
         <xmlSigCertPwd></xmlSigCertPwd>
    </simpleCACltProp>
       <!--
       <simpleCASrvProp>
           <certExpiryDate></certExpiryDate>
           <certLifetimeDays></certLifetimeDays>
           <certTmpDir></certTmpDir>
           <caCertFile></caCertFile>
           <signExe></signExe>
           <path></path>
       </simpleCASrvProp>
       -->
    <credReposProp>
           <modFilePath></modFilePath>
```

```
        <modName>ndg.security.common.CredWallet</modName>
        <className>NullCredRepos</className>
        <propFile></propFile>
    </credReposProp>
</sessMgrProp>
```

Notes

- The property file reading software will expand any environment variables included in the file.
- Openssl.conf file uses the standard OpenSSL configuration file format. It is used by the Session Manager MyProxy client to formulate a certificate request for a proxy certificate generated for a users session when they login. An example is given below. The important section to reference is `[ req_distinguished_name ]`

```
#
# SSLeay example configuration file.
# This is mostly being used for generation of certificate requests.
#

RANDFILE               = $ENV::HOME/.rnd

####################################################################
[ ca ]
default_ca      = CA_default            # The default ca section

####################################################################
[ CA_default ]

dir             = ./demoCA              # Where everything is kept
certs           = $dir/certs            # Where the issued certs are kept
crl_dir         = $dir/crl              # Where the issued crl are kept
database        = $dir/index.txt        # database index file.
new_certs_dir   = $dir/newcerts         # default place for new certs.

certificate     = $dir/cacert.pem       # The CA certificate
serial          = $dir/serial           # The current serial number
crl             = $dir/crl.pem          # The current CRL
private_key     = $dir/private/cakey.pem# The private key
RANDFILE        = $dir/private/.rand    # private random number file

x509_extensions = x509v3_extensions     # The extentions to add to the cert
default_days    = 365                   # how long to certify for
default_crl_days= 365 # DEE 30  # how long before next CRL
default_md      = md5                   # which md to use.
preserve        = no                    # keep passed DN ordering

# A few difference way of specifying how similar the request should look
# For type CA, the listed attributes must be the same, and the optional
# and supplied fields are just that :-)
policy          = policy_match

# For the CA policy
[ policy_match ]
countryName             = optional
stateOrProvinceName     = optional
organizationName        = match
organizationalUnitName  = optional
commonName              = supplied
```

```
emailAddress            = optional

# For the 'anything' policy
# At this point in time, you must list all acceptable 'object'
# types.
[ policy_anything ]
countryName             = optional
stateOrProvinceName     = optional
localityName            = optional
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

######################################################################
[ req ]
default_bits            = 1024
default_keyfile         = privkey.pem
distinguished_name      = req_distinguished_name
req_extensions          = v3_req

[ req_distinguished_name ]
# BEGIN CONFIG
0.organizationName               = Level 0 Organization
0.organizationName_default       = NDG
0.organizationalUnitName          = Level 0 Organizational Unit
0.organizationalUnitName_default = BADC
#1.organizationalUnitName          = Level 1 Organizational Unit
#1.organizationalUnitName_default = localdomain
commonName                      = Name (e.g., John M. Smith)
commonName_max                  = 64
# END CONFIG

[ v3_req ]
nsCertType                      = objsign,email,server,client
basicConstraints                = critical,CA:false
```

3.3.3Twisted Python server .tac file

Python security services use the Python Twisted package application server. A special .tac configuration file is loaded by the Twisted server. Copy this from the ndg_security_server to the NDG security conf/ area:

```
$ cp /usr/local/lib/python<python version num>/site-
packages/ndg_security_server-<version info>.egg/ndg/security/server/server-
config.tac $NDGSEC_DIR/conf
```

3.3.4SysV-style Boot Script

The Session Manager can be configured to start up at system boot of the host machine. A SysV style start up script `ndg-sm` is provided in the installation in:

/usr/local/lib/python<python version num>/site-packages/ndg_security_server-<version info>.egg/ndg/security/server/`share`

To configure, install this file:

```
$ cp /usr/local/lib/python<python version num>/site-
packages/ndg_security_server-<version info>.egg/ndg/security/server
/share/ndg-sm /etc/rc.d/init.d
$ chkconfig --add ndg-sm
```

Edit the ndg-sm so that it uses the NDGSEC_DIR environment variable to point to the correct location of the .tac file in the conf/ directory. User and group ID settings can be made to run under alternative account to root. If used ensure that $NDGSEC_DIR is set with the necessary permissions to enable access.

3.4Attribute Authority Configuration

The Attribute Authority also has a properties file for the setting of configuration parameters.

3.4.1Attribute Authority Properties File Settings

Edit attAuthorityProperties.xml in `$NDGSEC_DIR/conf` and modify the default settings:

```
<?xml version="1.0" encoding="utf-8"?>
<AAprop>
       <!--
       'name' setting MUST agree with map config file 'thisHost' name
       attribute
       -->
    <name>BADC</name>
    <portNum>SELECT A SUITABLE PORT NUMBER FOR RUNNING THE SERVICE</portNum>
    <!--
    PKI settings for transport level encryption
    -->
    <useSSL></useSSL> <!-- leave blank to use http -->
    <sslCertFile></sslCertFile>
    <sslKeyFile></sslKeyFile>
    <sslKeyPwd></sslKeyPwd>
    <!--
    PKI settings for signature of outbound SOAP messages
    -->
    <useSignatureHandler>Yes</useSignatureHandler> <!-- leave blank for no signature
-->
    <certFile>$NDGSEC_DIR/conf/certs/aa-cert.pem</certFile>
    <keyFile>$NDGSEC_DIR/conf/certs/aa-key.pem </keyFile>
    <keyPwd></keyPwd>
    <caCertFile>$NDGSEC_DIR/conf/certs/cacert.pem </caCertFile>
    <!--
    Set the certificate used to verify the signature of messages from the
    client.  This can usually be left blank since the client is expected to
    include the cert with the signature in the inbound SOAP message
    -->
    <clntCertFile></clntCertFile>
    <attCertLifetime>86400</attCertLifetime> <!-- Measured in seconds -->
       <!--
       Allow an offset for clock skew between servers running
       security services.  - Use minus sign for time in the past
       -->
    <attCertNotBeforeOff>0</attCertNotBeforeOff>
    <!-- Location of role mapping file -->
    <mapConfigFile>$NDGSEC_DIR/conf/mapConfig.xml</mapConfigFile>
    <!-- All Attribute Certificates are recorded in this dir before dispatch
    to SOAP requestor
    -->
    <attCertDir>$NDGSEC_DIR/conf/attCert</attCertDir>
    <!--
    File prefix and suffix for files stored in attCertDir
    -->
    <attCertFilePfx>ac-</attCertFilePfx>
    <attCertFileSfx>.xml</attCertFileSfx>
    <dnSeparator>/</dnSeparator>
    <!--
    Settings for custom AAUserRoles derived class to get user roles for
    given user ID
    -->
    <userRolesModFilePath></userRolesModFilePath>
    <userRolesModName></userRolesModName>
    <userRolesClassName></userRolesClassName>
    <userRolesPropFile></userRolesPropFile>
</AAprop>
```

3.4.2User Roles Interface

The Attribute Authority given a valid user proxy certificate serves an attribute certificate containing authorisation roles for that user. It is for the data centre to determine how these roles map to the users

identity as given by their Distinguished Name given in the proxy certificate. Typically, a data centre might have a user database which relates user id to authorisation roles.

The Attribute Authority provides a programmatic interface to determine the roles to user id relationship. A custom python class may be written to perform this task. See the Appendices section 4.4.

3.4.3Role Mapping

The role mapping file is stored in the `$NDGSEC_DIR/conf` directory as `mapConfig.xml` This is an XML file which relates local roles at the target data centre to roles of other trusted data centres. These role mapping are made by agreement between data centres.

```
<?xml version="1.0" encoding="utf-8"?>
<AAmap>
     <thisHost name="yourSiteIdentifier">
          <wsdl>yourSiteAttAuthorityURI</wsdl>
          <loginURI>yourSiteLoginPageURI</loginURI>
     </thisHost>
    <trusted name="BODC">
          <aaURI>bodcAttAuthorityURI</aaURI>
          <loginURI>bodcLoginPageURI</loginURI>
          <role remote="aBODCrole" local="aLocalRole"/>
    </trusted>
    <trusted name="NOCS">
          <aaURI>nocsAttAuthorityURI</aaURI>
          <loginURI>nocsLoginPageURI</loginURI>
          <role remote="aNOCSrole" local="anotherLocalRole"/>
    </trusted>
    <trusted name="PML">
          <aaURI>pmlAttAuthorityURI</aaURI>
          <loginURI>pmlLoginPageURI</loginURI>
          <role remote="aPMLrole" local="yetAnotherLocalRole"/>
    </trusted>
</AAmap>
```

<todo: >

3.4.4Twisted Python server .tac file

Python security services use the Python Twisted package application server. A special .tac configuration file is loaded by the Twisted server. Copy this from the ndg_security_server to the NDG security conf/ area:

```
$ cp /usr/local/lib/python<python version num>/site-
packages/ndg_security_server-<version info>.egg/ndg/security/server/server-
config.tac $NDGSEC_DIR/conf
```

3.4.5SysV-style Boot Script

As with the Session Manager, the Attribute Authority can be configured to start up at system boot of the host machine. A SysV style start up script `ndg-aa` is provided in the installation in:

/usr/local/lib/python<python version num>/site-packages/ndg_security_server-<version info>.egg/ndg/security/server/`share`

To configure, install this file:

```
$ cp /usr/local/lib/python<python version num>/site-
packages/ndg_security_server-<version info>.egg/ndg/security/server
/share/ndg-aa /etc/rc.d/init.d
$ chkconfig --add ndg-aa
```

Edit the ndg-aa so that it uses the NDGSEC_DIR environment variable to point to the correct location of the .tac file in the conf/ directory. User and group ID settings can be made to run under alternative account to root. If used ensure that $NDGSEC_DIR is set with the necessary permissions to enable access.

If required, add any additional environment settings required to connect to a user database.

3.5Python Unit Tests

Python unit test scripts are provided to enable the system to be checked to confirm that it is running correctly. These are located in the ndg_security_test egg in the site-packages/ directory of the python installation.

<todo: >

3.6Globus MyProxy

3.6.1MyProxy and NDG Security Background

NDG Security makes use of MyProxy from the Globus toolkit to store user's authentication credentials. If a participating data centre supports user accounts then it will need to deploy its MyProxy repository.

The NDG SessionManager web service acts as a client to MyProxy. When a user is registered at a site, it generates a new public/private key for the user and an X.509 certificate request. It sends the latter to the NDG Simple CA (Certificate Authority) for signing. A new X.509 certificate is issued and returned. The SessionManager uploads the public and private key into the MyProxy repository and associates a username and pass-phrase with these credentials.

When a user subsequently logs in at their site, again the SessionManager is called. It passes the username and pass-phrase provided to MyProxy. MyProxy matches these with the X.509 certificate it holds and issues a *proxy* to that certificate. The proxy certificate represents the user's ID internally in the interactions between the various NDG components.

MyProxy runs as a service `myproxy-server` on its host machine and user credentials are held in a directory on the file system. It is important to secure the host to ensure the credentials are not compromised. (Also see Ref 1above.)

3.6.2MyProxy user account and the repository location considerations

MyProxy may be installed as root or using a separate user account. The latter is preferable as it provides an extra level of security. Note that the MyProxy repository will be in a standard location.

- If MyProxy is installed as root, this is `/var/myproxy`.

- If installed as under an alternative user account, `$GLOBUS_LOCATION/var/myproxy`.

It is possible to explicitly define an alternate location but this can only be done by providing a command line argument to `myproxy-server` . Note that this might be visible in the process list of the host machine as output from `ps` This could be avoided by running `myproxy-server` with `xinetd` (See 3.6.8.1).

Another factor to take into consideration is the available space on the file system for the repository location. There should be sufficient disk space on the partition where the directory is located to store credentials for all the users of the system at the target site.

This guide assumes installation under a dedicated user account. The username `globus` is used in the examples for convenience only. An alternative username is recommended.

As `root` user set up a local user account.

```
$ groupadd globus
$ useradd globus –g globus
```

Note that for security purposes, the globus user account is set up as a local rather NIS account so that access is restricted. Set the default home directory as necessary and default shell to bash. Set the password for `globus`:

```
$ passwd globus
```

Modify the relevant files and directories in the NDG installation area to be owned by the `globus` account:

```
$ chown -R globus:globus $NDGSEC_DIR/conf/ $NDGSEC_DIR/ndgSetup.sh
```

For convenience, the `ndgSetup.sh` file may be called from the globus account's `.bashrc` file so that the NDG environment is automatically initialised when a new globus shell is invoked.

Change to the `globus` account and edit `~/.bashrc` adding the following lines at the end:

```
# NDG set-up
. /usr/local/NDG/ndgSetup.sh
```

3.6.3Build Process

As `root`, create an installation directory for Globus within the NDG installation:

```
$ mkdir $NDGSEC_DIR/globus-4.0.1
$ chown globus:globus $NDGSEC_DIR/globus-4.0.1
```

Ensure that the setting for GLOBUS_LOCATION in `$NDGSEC_DIR/ndgSetup.sh` points to the new directory created `$NDGSEC_DIR/globus-4.0.1`.

Switch to the `globus` user account ready to download the globus installation.

Globus 4.0.1 distribution is recommended for use with the NDG Security software. This is available from http://www.globus.org/toolkit/downloads/4.0.1/

A binary version is available but it is recommended to install the source code version and build from scratch on the target machine. Note that it is possible to set a target for `make` so that only the MyProxy components of Globus are built. Click on the link for the `gt4.0.1-all-source-installer.tar.bz2`. Extract the files and change to the `gt4.0.1-all-source-installer/` directory created.

Configure the build settings compile and install MyProxy:

```
$ ./configure –prefix=$GLOBUS_LOCATION
$ make gsi-myproxy postinstall
```

When running `./configure` you may see an error if the JAVA_HOME environment variable is not set. This can be ignored because Java is not required for the MyProxy build.

3.6.4NDG SimpleCA Client Package

This configures the target machine to trust the NDG CA.

Login as the `globus` user. To install first initialise the environment settings (The following line should be included in `ndgSetup.sh` Check and amend as necessary).

```
$ . $GLOBUS_LOCATION/etc/globus-user-env.sh
```

Install the client package. <CA Hash> below is a unique identifier for the CA. Note that the `–nonroot` option ensures that the configuration files are installed in `$GLOBUS_LOCATION/etc` rather than the default location used with the root user: `/etc/grid-security`. If you are installing as `root`, this option may be omitted if required.

Also note that for 64 bit architectures the `gcc32dbg` argument to `gpt-build` should be substituted with `gcc64dbg`.

```
$ gpt-build globus_simple_ca_<CA hash>_setup-0.18.tar.gz gcc32dbg
$ gpt-postinstall
$ $GLOBUS_LOCATION/setup/globus_simple_ca_<CA hash>_setup/setup-gsi
–default –nonroot
```

When running `gpt-postinstall`, you may see a warning:

```
WARNING: The following packages were not set up correctly:
        globus_simple_ca_<CA hash>_setup-noflavor-pgm
Check the package documentation or run postinstall -verbose to see what
happened
```

This can be ignored.

3.6.4.1Modifications to Configuration File Settings

The configuration files installed require some minor modifications before proceeding:

Under the directory $GLOBUS_LOCATION/etc, edit `globus-host-ssl.conf` and under the section `[ req_distinguished_name ]`, edit the setting for `0.organizationalUnitName_default` and change the default BADC to the name of the organisation where this NDG security software is being installed. This name will be used as the default for the OU field of certificates held in the MyProxy server.

```
[ req_distinguished_name ]
# BEGIN CONFIG
0.organizationName               = Level 0 Organization
0.organizationName_default       = NDG
0.organizationalUnitName          = Level 0 Organizational Unit
0.organizationalUnitName_default = BADC
commonName                      = Name (e.g., John M. Smith)
commonName_max                  = 64
# END CONFIG
```

Under the same directory, edit the file `globus-user-ssl.conf` and carry out the same modification as above but also comment out the two lines below `1.organizationalUnitName` and `1.organizationalUnitName_default`:

```
[ req_distinguished_name ]
# BEGIN CONFIG
0.organizationName               = Level 0 Organization
0.organizationName_default       = NDG
0.organizationalUnitName          = Level 0 Organizational Unit
0.organizationalUnitName_default = BADC

#1.organizationalUnitName          = Level 1 Organizational Unit
#1.organizationalUnitName_default = badc.rl.ac.uk
commonName                      = Name (e.g., John M. Smith)
commonName_max                  = 64
# END CONFIG
```

Edit $GLOBUS_LOCATION/share/certificates/<CA Hash>.signing_policy and change the setting of `OU` in the line:

```
cond_subjects     globus       '"/O=NDG/OU=BADC/*"'
```

Replacing 'BADC' with the name of the Organisational Unit for your organisation. This should be the same as 0.organizationalUnitName_default set above for globus-host-ssl.conf and globus-user-ssl.conf.

Having completed these steps, a host certificate for the target machine can be made in order to identify it.

3.6.5Host Certificate Creation

Login as globus user to carry out these steps. ndgSetup.shshould configure the PATH variable to have included the Globus executable directories $GLOBUS_LOCATION/bin and $GLOBUS_LOCATION/sbin. Check the path to the command grid-cert-request:

```
$ which grid-cert-request
```

Should return something like: /usr/local/NDG/globus-4.0.1/bin/grid-cert-request

To generate a host certificate request, change to the certificates directory:

```
$ cd $GLOBUS_LOCATION/etc
```

Nb. If you installed MyProxy as root , as root user change to /etc/grid-securitywhere the certificates should be held.

```
$ grid-cert-request –host <machine hostname> -dir .
```

This creates the files hostcert.pem , hostkey.pemand hostcert_request.pem. hostcert.pem is empty.

In order to obtain the certificate it must be signed by the NDG CA. Contact the NDG CA forwarding hostcert_request.pem . The CA will issue a hostcert.pemfile. Copy this file into this directory i.e. $GLOBUS_LOCATION/etc. hostcert_request.pem is no longer needed and may be deleted if desired.

3.6.6MyProxy Configuration File

A MyProxy configuration file is normally kept in the Globus installation under the etc directory. If this file is not already present, copy the sample file:

```
$ cp $GLOBUS_LOCATION/share/myproxy/myproxy-server.config
$GLOBUS_LOCATION/etc
```

As the globus user edit $GLOBUS_LOCATION/etc/myproxy-server.config

Modify the entries under the section Complete Sample Policy so that they are all uncommented (remove leading # character):

```
#
# Complete Sample Policy
#
# The following lines define a sample policy that enables all
# myproxy-server features.  See below for more examples.
accepted_credentials  "*"
authorized_retrievers "*"
default_retrievers    "*"
authorized_renewers   "*"
default_renewers      "none"
authorized_key_retrievers "*"
default_key_retrievers "none"
```

Note that the wildcards for these fields may be modified such that only Distinguished Names of a given format may be accepted e.g. "/O=NDG/OU=BADC/*"

3.6.7Repository Directory

A directory needs to be specified on the file system to store the user credentials generated by MyProxy. This should be owned by the account that runs myproxy-server . In the examples given this would be the globus user and the expected location, $GLOBUS_LOCATION/var . See section 2.3.2 *MyProxy user account and repository location*.

Login as the globus user and change directory to the location for the repository:

```
$ cd $GLOBUS_LOCATION/var
$ mkdir myproxy
$ chmod 700 myproxy
```

The chmod command ensures that only the globususer has read/write access for the directory. Note also that the directory need not be called myproxy.

3.6.8Adding MyProxy Server to the system start up

Any of the standard mechanisms may be used such as adding a SysV style init script or using inetd or xinetd inetd/xinetd are preferred:

- myproxy-server process will not show on ps command listing
- It's more efficient since it's only invoked when a request from a MyProxy client is received.
- It's easy to configure so that myproxy-server runs as an alternative user to root

3.6.8.1inetd / xinetd

To run the myproxy server using inetd or xinetd as root user:

- Add the entries in $GLOBUS_LOCATION/share/myproxy/etc.services.modifications to the

```
myproxy-server  7512/tcp                        # Myproxy server
```

- Add the entries from

  - For inetd add to /etc/inetd.conf or /etc/inet/inetd.conf.
  - for xinetd , copy $GLOBUS_LOCATION/share/myproxy/etc.xinetd.myproxy to /etc/xinetd.d/myproxy. Modify the paths in the file according to your installation and set the user to the correct user name for running myproxy-server e.g.

```
service myproxy-server
{
  socket_type  = stream
  protocol     = tcp
  wait         = no
  user         = globus
  server       = /usr/local/NDG/globus-4.0.1/sbin/myproxy-server
  env           = GLOBUS_LOCATION=/usr/local/NDG/globus-4.0.1
LD_LIBRARY_PATH=/usr/local/NDG/globus-4.0.1/lib
  disable      = no
  only_from    = localhost.localdomain <hostAddress1> <hostAddress2>
}
```

- Note also, the additional setting in this example for only_from . This a limit to be placed on which hosts clients can connect from to the server. In the above, clients can connect from the local machine (note the fully qualified name including localdomain) and from the hosts <hostAddress1> and <hostAddress2>.

- Reactivate the inetd / xinetd This is typically accomplished by sending the SIGHUP signal to the server process. Redhat Linux machines include the GUI tool redhat-config-services to allow convenient management of services. Refer to the inetd or xinetd man page for your system.

3.6.8.2SysV-style boot script

A sample SysV-style boot script for is available in the Globus installation at, $GLOBUS_LOCATION/share/myproxy/etc.init.d.myproxy.

To install:

```
$ cp $GLOBUS_LOCATION/share/myproxy/etc.init.d.myproxy /etc/rc.d/init.d/myproxy
$ chkconfig --add myproxy
```

Edit the file to set the GLOBUS_LOCATION environment variable correctly.

4. APPENDICES

4.1 MySQL Installation

MySQL is required for the Credential Repository used by the SessionManager to stored user credentials as cached in their Credential Wallet held in their session.

This section describes how to make an installation from the MySQL binary package tarball. System administrators may wish to use an existing installation of MySQL or use an alternative installation method such as rpm. Installing from the binary package has the advantage that it doesn't interfere with any existing MySQL installation on the target machine. The instructions are adapted from the file `INSTALL-BINARY` provided in the tarball.

4.1.1 Version

Version 3.23 or later is recommended. These instructions are for version 5.0.20a, the latest stable release at time of writing.

4.1.2 Getting the Binaries

The package can be obtained from the MySQL web site (http://dev.mysql.com/downloads/mysql/5.0.html). Scroll to the correct version - Linux (non RPM, Intel C/C++ compiled, glibc-X.X) downloads. The version of glibc on the target machine can be checked using same machine as the web server.

```
$ ls /lib/libc-*
```

4.1.3 New `mysql` User Account

Make a new account to run MySQL if it doesn't already exist:

```
$ groupadd mysql
$ useradd -g mysql mysql
```

4.1.4 Unpacking the tarball

As root copy the tarball to the target directory for installation e.g. /usr/local, unpack the file:

```
$ cd /usr/local
$ tar zxvf mysql-standard-5.0.20a-linux-i686-icc-glibc23.tar.gz
```

Make a symbolic link to the new directory and '`cd`' to it:

```
$ ln -s /usr/local/mysql-standard-5.0.20a-linux-i686-icc-glibc23 mysql
$ cd mysql
```

The bin directory contains client programs and the server. You should add the full pathname of this directory to your PATH environment variable so that your shell finds the MySQL programs properly.

4.1.5Configuration File

Create a configuration file called my.cnf in the target directory (/usr/local/mysql in this example) to enable custom settings to be made for this installation. Note that if there is an existing installation of MySQL, there may be settings existing settings in a file /etc/my.cnf. To use the settings from this file, *ignore* this step.

```
[mysqld]
datadir=/usr/local/mysql-standard-5.0.20a-linux-i686-icc-glibc23/data
socket=/tmp/mysql.sock
# Default to using old password format for compatibility with mysql 3.x
# clients (those using the mysqlclient10 compatibility package).
old_passwords=1

[mysql.server]
user=mysql
basedir=/usr/local/mysql-standard-5.0.20a-linux-i686-icc-glibc23

[mysqld_safe]
err-log=/var/log/mysqld.log
pid-file=/tmp/mysql.pid
```

The settings above will mean that MySQL's tables and the Credential Repository database will be stored under /usr/local/mysql/data.

4.1.6Create the Grant Tables

The scripts directory contains the mysql_install_db script used to initialize the mysql database containing the grant tables that store the server access permissions. If you have not installed MySQL before, you must create the MySQL grant tables:

```
$ scripts/mysql_install_db --user=mysql
```

If you run the command as root, you must use the --user option as shown. The value of the option should be the name of the login account that you created in the first step to use for running the server. If you run the command while logged in as that user, you can omit the -user option. After creating or updating the grant tables, you need to restart the server manually.

4.1.7File and Directory Permissions

Change the ownership of program binaries to root and ownership of the data directory mysql. Assuming that you are located in the installation directory (/usr/local/mysql), the commands look like this:

```
$ chown -R root  .
$ chown -R mysql data
$ chgrp -R mysql .
```

The first command changes the owner attribute of the files to the root user. The second changes the owner attribute of the data directory to the mysql user. The third changes the group attribute to the mysql group.

4.1.8Starting the Server

If you want MySQL to start automatically when you boot your machine, you can copy support-files/mysql.server to the location where your system has its startup files. More information can be found in the support-files/mysql.server script itself.

To start the MySQL server, use the following command:

```
$ bin/mysqld_safe --user=mysql &
```

If that command fails immediately and prints mysqld ended, you can find some information in the <hostname>.err file in the data directory.

4.1.9Securing MySQL Accounts

To delete the anonymous accounts:

```
$ mysql -u root
mysql> DELETE FROM mysql.user WHERE User = '';
mysql> FLUSH PRIVILEGES;
```

Set the password for the root account:

```
mysql> SET PASSWORD FOR 'root'@'localhost' = PASSWORD('newpwd');
mysql> SET PASSWORD FOR 'root'@'hostname' = PASSWORD('newpwd');
```

The hostname can be checked using the query:

```
mysql> SELECT Host, User FROM mysql.user;
```

Add a new account for use with the Credential Repository database e.g.

```
mysql> GRANT SELECT, UPDATE, INSERT, DELETE ON ndgCredRepos.* TO
'ndgUser'@'localhost' IDENTIFIED BY 'password' WITH GRANT OPTION;
```

The above statement grants the user, ndgUser with password, password, select, update and insert privileges on the tables of database ndgCredRepos. The user may only connect from the localhost. Hence, in this case the Session Manager and Credential Repository must be installed on the same machine. To allow the Credential Repository to run on a separate machine to the Session Manager, the account must have permission to connect remotely. This can be achieved by altering the GRANT statement above to:

```
mysql> GRANT SELECT, UPDATE, INSERT, DELETE ON ndgCredRepos.* TO
'ndgUser'@'%' IDENTIFIED BY 'password' WITH GRANT OPTION;
```

You also can set up new accounts using the bin/mysql_setpermission script if you install the `DBI' and `DBD::mysql' Perl modules.

See section 3.3.1 for details about creation of the Credential Repository database.

4.1.10Server Automated Start up

<todo: >

4.2HTTPS set-up with Apache Web Server

NDG security requires HTTPS for the transfer of user credentials across cookie domains between a data provider web page requesting user credentials and a user's NDG home login page.

<todo: full explanation - incl. mod_ssl must be installed>

4.2.1Web Server Host Certificate Generation

```
$ grid-cert-request -prefix <hostname> -dir . -cn <hostname> -nopw
```

4.2.2Apache Configuration File Settings

4.3Apache Web Server Proxy Settings Configuration for Web Services

Apache provides a convenient mechanism to re-route web service ports through port 80 and so make them available to the outside world. This may be helpful if when deploying NDG Security you do not wish to open additional ports in your site firewall settings.

Edit the Apache configuration file. This should be located at `/etc/httpd/conf`

Add `ProxyPass` and `ProxyPassReverse` entries for the Session Manager and Attribute Authority web services. The first argument after the directive name itself is the directory that the service will be served from relative to the web server URL. So below, if the URL of the web server is http://www.badc.rl.ac.uk, then the Session Manager would be available at https://www.badc.rl.ac.uk/sessionMgr. The second argument is the actual location where the web service is running locally. In the example below, the Session Manager is running on port 5700 on the same machine as the web server.

```
# Session Manager and Attribute Authority settings
ProxyPass        /sessionMgr    https://localhost:5700/
ProxyPassReverse /sessionMgr    https://localhost:5700/

ProxyPass        /attAuthority  http://localhost:5000/
ProxyPassReverse /attAuthority  http://localhost:5000/
```

Restart the Apache web server. This can be done in a variety of ways. As root user:

1. On Redhat machines, using the command `redhat-config-services` or `system-config-services`In the GUI, click on httpd in the list and press the Restart button

```
$ redhat-config-services
```

2. `service` command

```
$ /sbin/service httpd restart
```

3. apache command

```
$ apachectl restart
```

4. Using `kill`

```
$ kill -HUP `cat /etc/httpd/run/httpd.pid`
```

Note in the last case that the location of the pid file will depend on your installation.

Once the changes have been made, ensure that sessionMgr.wsdl and attAuthority.wsdl contain the new locations for the web services in the tag <soap:address location="..."/>

4.4An Example Attribute Authority AAUserRoles interface class

This interface is required in order to link the Attribute Authority to the data centre's system for identifying registered users and managing their roles. The installation comes with a simple test class which illustrates this:

```
"""NDG Attribute Authority User Roles class - acts as an interface between
the data centre's user roles configuration and the Attribute Authority

NERC Data Grid Project

P J Kershaw 29/07/05

Copyright (C) 2005 CCLRC & NERC

This software may be distributed under the terms of the Q Public License,
version 1.0 or later.
"""
cvsID = '$Id'


from AttAuthority import AAUserRoles


class TestUserRoles(AAUserRoles):
    """Test User Roles class dynamic import for Attribute Authority"""

    def __init__(self, propertiesFilePath=None):
        pass


    def userIsRegistered(self, dn):
        return True


    def getRoles(self, dn):
        return ['staff', 'postdoc', 'undergrad']
```

The class must inherit from the AAUserRoles interface class. It must override the userIsRegistered and getRoles methods:

- userIsRegistered() – returns True if the user with the given input Distinguished Name is registered at the site. This method might contain an SQL query to the site's user database for example. This method is *optional*.

- getRoles() – returns a list of roles to which the user with the given input Distinguished Name is enrolled. Again, this method could be implemented with an SQL query to retrieve the roles for a given user. Note, that if not roles are found, the method should return [].

- `__init__()` – optionally, the initialisation method may be overridden to enable for example the setting up of a database connection. The path to a properties file may be passed in. This could contain database connection settings.

The custom class used by the BODC is a more detailed example:

```
"""NDG Attribute Authority User Roles class for the BODC - acts as an
interface
between BODC user database and the Attribute Authority

NERC Data Grid Project

P J Kershaw 09/09/05

Copyright (C) 2005 CCLRC & NERC

This software may be distributed under the terms of the Q Public License,
version 1.0 or later.
"""
from DCOracle2 import *

# For parsing of properties file
import cElementTree as ElementTree

from NDG.X509 import *
from NDG.AttAuthority import AAUserRoles
from NDG.AttAuthority import AAUserRolesError


class BODCUserRoles(AAUserRoles):
    """User Roles class dynamic import for BODC Attribute Authority"""

    # valid configuration property keywords
    __validKeys = [ 'userName', 'dbAddr']


    def __init__(self, propFilePath=None):

      self.__db = None

      if propFilePath:
          prop = self.readProperties(propFilePath)
          self.connect(prop['userName'], prop['dbAddr'])


    def readProperties(self, propFilePath):

        """Read the configuration properties for the Attribute Authority

        propFilePath: file path to properties file
        """

        try:
            tree = ElementTree.parse(propFilePath)

        except IOError, ioErr:
            raise AAUserRolesError(\
                            "Error parsing properties file \"%s\": %s" % \
```

```
                            (ioErr.filename, ioErr.strerror))


        prop = tree.getroot()

        # Copy properties from file as member variables
        userRolesProp = \
            dict([(elem.tag, elem.text.strip()) for elem in prop])


        # Check for missing properties
        propKeys = userRolesProp.keys()
        missingKeys = [key for key in BODCUserRoles.__validKeys \
                       if key not in propKeys]
        if missingKeys != []:
            raise AAUserRolesError("The following properties are " + \
                                   "missing from the properties file: " + \
                                   ', '.join(missingKeys))

      return userRolesProp




    def connect(self,
            userName,
            dbAddr,
            passPhrase=None,
            prompt=None):
      """Connect to database

      If no passphrase is given prompt from stdin"""


      if not passPhrase:
          if not prompt:
            prompt = "Database Passphrase: "

          import getpass
            passPhrase = getpass.getpass(prompt=prompt)


        try:
          self.__db = connect("%s/%s@%s" % (userName, passPhrase, dbAddr))
          self.__cursor = self.__db.cursor()

      except Exception, e:
          raise AAUserRolesError(\
            "Error connecting to database \"%s\": %s" % (dbAddr, e))


    def userIsRegistered(self, dn):
      """Check user with given Distinguished Name is registered with
      BODC database"""

      try:
          emailAddr = X500DN(dn)['CN']
          query = "<BODC Database query>"
            self.__cursor.execute(query, emailAddr)
```

```
        if self.__cursor.fetchall():
          return True
        else:
            return False

    except Exception, e:
        raise AAUserRolesError(\
          "Error checking user \"%s\" is registered: %s" % (dn, e))


  def getRoles(self, dn):
      """Retrieve roles from user with given Distinguished Name"""
    try:
        emailAddr = X500DN(dn)['CN']
        query = "<BODC Database query>"
          self.__cursor.execute(query, emailAddr)
        roles = self.__cursor.fetchall()
        return [i[0] for i in roles]

    except Exception, e:
        raise AAUserRolesError(\
        "Error getting roles for user \"%s\" is registered: %s" % (dn, e))
```

Note:

- It use the Python library `DCOracle2` to connect to an Oracle database.
- `ElementTree` Python library is used to parse an XML properties file.

NDG.X509security python library is used to parse the user Distinguished Name passed into `getRoles` and userIsRegistered methods.

4.5Troubleshooting

4.5.1M2Crypto SWIG Build Error

M2Crypto uses SWIG to bind C OpenSSL library code to the Python interface. Compilation errors with swig `.i` files in the M2Crypto tar bundle can be caused by using an earlier version of swig. This has been seen with the default swig on Redhat EL4. This comes with swig version 1.1. To check the SWIG version number type:

```
$ swig -version
```

To fix update to a version > 1.1 and re-run the installation script. SWIG is available from http://www.swig.org/

4.5.2PyXML

error: Could not find suitable distribution for Requirement.parse('PyXML>=0.8.3')

$ easy_install –f http://sourceforge.net/project/showfiles.php?group_id=6473 PyXML

or –f option with ndg-security-install.py

4.5.34Suite-XML Build error

Ft/Xml/src/expat/lib/xmlparse.c:89:2: #error memmove does not exist on this platform, nor is a substitute available

4Suite-XML 1.0.2

$ cat /proc/version

Linux version 2.4.21-32.0.1.ELsmp (bhcompile@bugs.build.redhat.com) (gcc version

3.2.3 20030502 (Red Hat Linux 3.2.3-52)) #1 SMP Tue May 17 17:52:23 EDT 2005

$ uname –a

Linux glue.badc.rl.ac.uk 2.4.21-32.0.1.ELsmp #1 SMP Tue May 17 17:52:23 EDT 2005 i686 i686 i386 GNU/Linux

Solution

$ echo -e "[build_ext]\ndefine=HAVE_MMEMOVE" > ~/.pydistutils.cfg

$ easy_install 4Suite-XML