FermiGrid
Site AuthoriZation Service (SAZ)
Overview

Keith Chadwick
09-Sep-2008

Abstract:

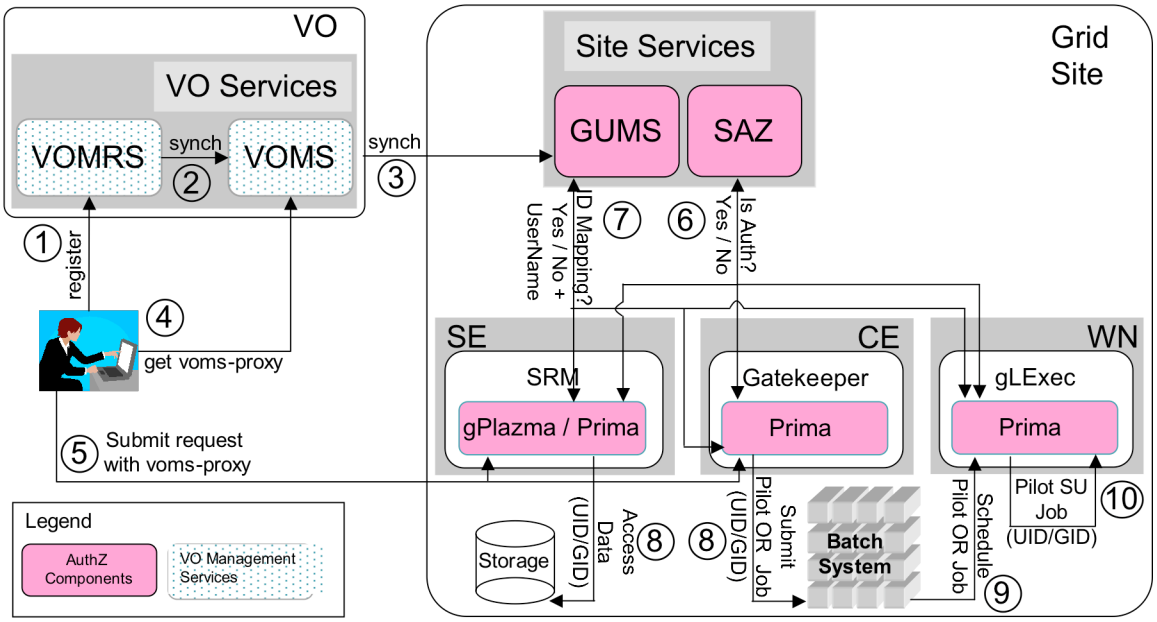
This document provides an overview description of the FermiGrid Site AuthoriZation Service (SAZ).

Document Revision History:

Version	Date	Author	Comments
0.1	04-Jun-2008	Keith Chadwick	Initial version.
0.2	06-Jun-2008	Keith Chadwick	Incorporate initial feedback.
0.3	13-Jun-2008	Keith Chadwick	Incorporate additional feedback.
0.4	09-Sep-2008	Keith Chadwick	Update document to reflect merger of sazclient and saz-check utilities

Introduction:

FermiGrid operates the Site AuthoriZation Service (SAZ) that serves to implement a Fermilab site wide whitelist and blacklist capability for Grid jobs. SAZ is a component of the VO services project (http://www.fnal.gov/docs/products/voprivilege/).



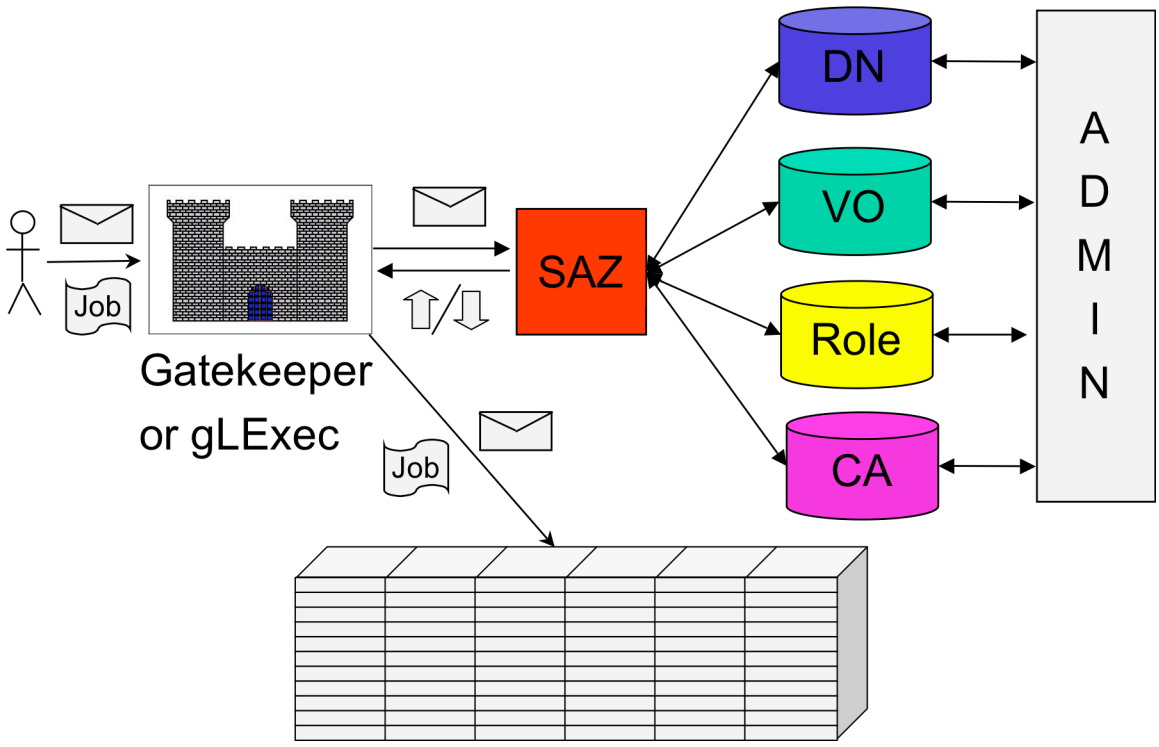
Components:

The Site AuthoriZation Service is comprised of the following components:

- The SAZ server with backend MySQL database.
- The SAZ client for Globus Gatekeepers.
- The SAZ client for gLExec (used within pilot jobs).
- The interactive sazclient utility.
- The SAZ client (plugin) for gPlazma.
- The SAZ command line administration script.
- The SAZ web administration interface (future).

Each of these components will be described in greater detail below.

The picture below shows a general diagram of how SAZ is deployed within FermiGrid (the SAZ server and backend database are the colored objects):



Communication between the SAZ Client(s) and the SAZ Server:

The SAZ clients and the SAZ server communicate via one of the three mechanisms detailed below:

Mechanism 1 – The public portions of the users certificate are transmitted from the SAZ client to the SAZ server. Any private key that may be embedded in the certificate is not transmitted to the SAZ server. The SAZ server then parses the desired information from the presented certificate (all current versions of SAZ).

Mechanism 2 – The SAZ client parses out the desired information (typically from the users certificate) and sends this information to the SAZ server in an XACML message. Upon receipt of the XACML message, the SAZ server parses out the desired information from within the XACML message (future XACML based SAZ clients with SAZ Server V3.0 and greater).

Mechanism 3 – The SAZ client parses out the desired information (typically from the users certificate) and submits this information to the SAZ server as a series of http requests. Upon receipt of the http request, the SAZ server performs the appropriate lookup of the corresponding information in the backend database and returns the information to the requestor (future http based SAZ clients with SAZ Server V3.1 and greater).

Note – As of June 2008, the current version of the SAZ Server is V2.0.0.

SAZ Server:

Upon receipt of an authorization request, the SAZ server parses out the following information elements from the authorization request:

- The user's Distinguished Name (DN).
- The user's Virtual Organization (VO).
- The user's Group and Role within the Virtual Organization (Role).
- The Certificate Authority (CA) that issued the user's initial certificate.

Using the above information, queries are performed against the SAZ backend database (currently MySQL). For each information element, the following fields (variables) are returned:

Information Element	Variable Name	Variable Enabled Value	Variable Trusted Value
DN	user	user.enabled	user.trusted
VO	vo	vo.enabled	vo.trusted
Role	role	role.enabled	role.trusted
CA	ca	ca.enabled	ca.trusted

The enabled field values are used to authorize or deny access to the resource (blacklist). The trusted field values are used to support (optional) whitelist authorization.

The SAZ server constructs the authorization decision (whitelist and/or blacklist) based on the enabled and trusted field values retrieved from the corresponding database queries as follows:

```
if [ grid-proxy ]
then
        if [ user.trusted ]
        then
                result = user.enabled && ca.enabled

        else
                result = user.enabled && vo.enabled && role.enabled && ca.enabled
        fi
elif [ "$saz.mode" = "pilot" ]
then
        result = user.enabled && vo.enabled && role.enabled && role.trusted && ca.enabled
else
        result = user.enabled && vo.enabled && role.enabled && ca.enabled
fi
```

Note 1 – The above pseudocode includes optional mode logic that will shortly be added to the SAZ client – server communication as part of the SAZ server V3.0 enhancements.

Note 2 – Within FermiGrid, the "NULL" VO is set enabled="N" and trusted="N". Users that submit jobs with legacy grid-proxy-init generated proxies will be unable to run unless they apply for (and are granted) trusted access to FermiGrid resources.

Note 3 – If any query returns "no such record", the SAZ server performs an insert of a corresponding record into the database table using the values for enabled and trusted variables that the SAZ administrator has configured in:

$VDT_LOCATION/saz/server/conf/sazserver.conf

The SAZ server installation automatically creates this file.

Here are the contents of the typical sazserver.conf:

```
<?xml version="1.0"?>
<saz>
    <SAZ_SERVER_PORT>8888</SAZ_SERVER_PORT>
    <SAZ_SERVER_CERT>/etc/grid-security/hostcert.pem</SAZ_SERVER_CERT>
    <SAZ_SERVER_KEY>/etc/grid-security/hostkey.pem</SAZ_SERVER_KEY>
    <CA_DIR>/etc/grid-security/certificates</CA_DIR>
    <SAZ_USER_ENABLED>Y</SAZ_USER_ENABLED>
    <SAZ_USER_TRUSTED>N</SAZ_USER_TRUSTED>
    <SAZ_CA_ENABLED>Y</SAZ_CA_ENABLED>
    <SAZ_CA_TRUSTED>N</SAZ_CA_TRUSTED>
    <SAZ_VO_ENABLED>Y</SAZ_VO_ENABLED>
    <SAZ_VO_TRUSTED>N</SAZ_VO_TRUSTED>
    <SAZ_ROLE_ENABLED>Y</SAZ_ROLE_ENABLED>
    <SAZ_ROLE_TRUSTED>N</SAZ_ROLE_TRUSTED>
</saz>
```

Communication between the SAZ Server and Backend Database:

Prior to SAZ V2.0, all communication between the SAZ server and the SAZ backend database occurred through direct MySQL calls.

For SAZ V2.0 and above, the SAZ server uses the hibernate method to interface between the SAZ server and the SAZ backend database.

The SAZ hibernate method configuration is stored in the following configuration file:

$VDT_LOCATION/saz/server/conf/hibernate.cfg.xml

The SAZ server installation automatically creates this file.

Here are the contents of the typical hibernate.cfg.xml:

```
<?xml version='1.0' encoding='utf-8'?>
<!DOCTYPE hibernate-configuration PUBLIC
      "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
      "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

   <session-factory>

      <!-- Database connection settings -->
      <property name="connection.driver_class">org.gjt.mm.mysql.Driver</property>
      <property name="connection.url">jdbc:mysql://fg-mysql.fnal.gov:3306/SAZDB</property>
      <property name="connection.username">sazdbuser</property>
      <property name="connection.password">12345</property>

      <!-- JDBC connection pool (use the built-in) -->
      <property name="hibernate.c3p0.min_size">5</property>
      <property name="hibernate.c3p0.max_size">50</property>
      <property name="hibernate.c3p0.timeout">60</property>
      <property name="hibernate.c3p0.max_statements">0</property>
      <!-- SQL dialect -->
      <property name="dialect">org.hibernate.dialect.MySQLDialect</property>

      <!-- Enable Hibernate's automatic session context management -->
      <property name="current_session_context_class">thread</property>

      <!-- Disable the second-level cache  -->
      <property name="cache.provider_class">org.hibernate.cache.NoCacheProvider</property>

      <!-- Echo all executed SQL to stdout -->
      <property name="show_sql">false</property>

      <!-- Drop and re-create the database schema on startup -->
      <!--Neha -Commenting below so that schema is not dropped -->
      <!--<property name="hbm2ddl.auto">create</property> -->
      <mapping resource="User.hbm.xml"/>
      <mapping resource="CA.hbm.xml"/>
      <mapping resource="Role.hbm.xml"/>
      <mapping resource="VO.hbm.xml"/>
   </session-factory>

</hibernate-configuration>
```

SAZ Server Logging:

The SAZ server uses the standard log4j method to log information, and the SAZ server logs are stored in: $VDT_LOCATION/saz/server/log (note that this directory may be optionally linked to: /var/log/saz). In addition, these logs are forwarded via syslog-ng to the central Fermilab splunk collector.

A typical SAZ server log entry where the authorization request was granted is:

```
2008-06-04 11:23:25 :INFO :Thread-152375: Received SAZclient authorization request from 131.225.204.224 : cmsosgce3.fnal.gov
2008-06-04 11:23:25 :INFO :Thread-152375: VO fermilab
2008-06-04 11:23:25 :INFO :Thread-152375: Role fermilab/Role=NULL/Capability=NULL
2008-06-04 11:23:25 :INFO :Thread-152375: User /DC=gov/DC=fnal/O=Fermilab/OU=Robots/CN=fermigrid0.fnal.gov/CN=cron/CN=Keith Chadwick/CN=UID:chadwick
2008-06-04 11:23:25 :INFO :Thread-152375: Cert /DC=gov/DC=fnal/O=Fermilab/OU=Certificate Authorities/CN=Kerberized CA
2008-06-04 11:23:25 :INFO :Thread-152375: Sending to client Y
```

A typical SAZ server log entry where the authorization request was rejected is:

```
2008-06-04 11:24:22 :INFO :Thread-152423: Received SAZclient authorization request from 131.225.167.42 : fngp-osg.fnal.gov
2008-06-04 11:24:23 :INFO :Thread-152423: VO NULL
2008-06-04 11:24:23 :INFO :Thread-152423: Role NULL
2008-06-04 11:24:23 :INFO :Thread-152423: User /DC=org/DC=doegrids/OU=People/CN=Mats Rynge 722233
2008-06-04 11:24:23 :INFO :Thread-152423: Cert /DC=org/DC=DOEGrids/OU=Certificate Authorities/CN=DOEGrids CA 1
2008-06-04 11:24:23 :INFO :Thread-152423: Sending to client N
```

It should be noted that the SAZ server is a multi-threaded java application, so the messages from multiple simultaneous authorization requests in the SAZ server log may be intermixed. The unique thread identifier may be used to extract the messages that correspond to a single authorization request.

SAZ Client for Globus Gatekeeper:

The Globus Gatekeeper is a process used by Grid Computing Elements (CEs) to take incoming job requests and check the security to make sure each is allowed to use the associated computing resource(s). The gatekeeper process starts up the job-manager process after successful authentication.

The SAZ client for Globus Gatekeepers is configured through the globus_authorization entry in /etc/grid-security/gsi-authz.conf configuration file:

```
globus_authorization /usr/local/vdt-1.8.1/sazclient-1.2/lib/libSAZ-gt3.2_gcc32dbg globus_saz_access_control_callout
globus_mapping /usr/local/vdt-1.8.1/prima/lib/libprima_authz_module_gcc32dbg globus_gridmap_callout
```

The host name, port number and DN of the SAZ server that the SAZ client contacts are configured in:

/etc/grid-security/sazc.conf

The typical contents of /etc/grid-security/sazc.conf are:

```
SAZ_SERVER_HOST saz.fnal.gov
SAZ_SERVER_PORT 8888
SAZ_SERVER_DN   /DC=org/DC=doegrids/OU=Services/CN=saz.fnal.gov
```

Multiple SAZ servers may be configured in sazc.conf. If more than one SAZ server is configured in sazc.conf, the SAZ servers are contacted in series for each and every authorization request.

The SAZ globus gatekeeper client stores the SAZ communication and decision information in the log files of the globus gatekeeper in:

$VDT_LOCATION/globus/var

A typical entry in the (pre web services) globus gatekeeper log is shown below:

```
TIME: Thu Jun  5 00:03:08 2008
 PID: 24196 -- Notice: 5: Authenticated globus user:
/DC=gov/DC=fnal/O=Fermilab/OU=Robots/CN=fermigrid0.fnal.gov/CN=cron/CN=Keith Chadwick/CN=UID:chadwick
TIME: Thu Jun  5 00:03:08 2008
 PID: 24196 -- Notice: 0: GATEKEEPER_JM_ID 2008-06-05.00:03:08.0000024196.0000000000 for
/DC=gov/DC=fnal/O=Fermilab/OU=Robots/CN=fermigrid0.fnal.gov/CN=cron/CN=Keith Chadwick/CN=UID:chadwick on
131.225.1
07.12
 PID: 24196 -- PRIMA INFO ts=2008-06-05T00:03:08-06:00 event=org.osg.prima.authz.start
DN="/DC=gov/DC=fnal/O=Fermilab/OU=Robots/CN=fermigrid0.fnal.gov/CN=cron/CN=Keith Chadwick/CN=UID:chadwick"
FQAN="/fe
rmilab/Role=NULL/Capability=NULL" FQAN_Issuer="/DC=org/DC=doegrids/OU=Services/CN=http/voms.fnal.gov"
Service_URL="https://gums.fnal.gov:8443/gums/services/GUMSAuthorizationServicePort"
 PID: 24196 -- PRIMA INFO ts=2008-06-05T00:03:08-06:00 event=org.osg.prima.authz.end status=0 decision=PERMIT
DN="/DC=gov/DC=fnal/O=Fermilab/OU=Robots/CN=fermigrid0.fnal.gov/CN=cron/CN=Keith Chadwick/CN=
UID:chadwick" FQAN="/fermilab/Role=NULL/Capability=NULL"
FQAN_Issuer="/DC=org/DC=doegrids/OU=Services/CN=http/voms.fnal.gov"
Service_URL="https://gums.fnal.gov:8443/gums/services/GUMSAuthorizationService
Port" local_user=fnalgrid
 PID: 24196 -- SAZ DEBUG Client.c:69 Successfully authorized this user
 PID: 24196 -- SAZ DEBUG SAZ-gt3.2.c:296 User Authorized by SAZ
TIME: Thu Jun  5 00:03:09 2008
 PID: 24196 -- Notice: 5: Requested service: jobmanager-fork [PING ONLY]
TIME: Thu Jun  5 00:03:09 2008
 PID: 24196 -- Notice: 5: Authorized as local user: fnalgrid
TIME: Thu Jun  5 00:03:09 2008
 PID: 24196 -- Notice: 5: Authorized as local uid: 13160
TIME: Thu Jun  5 00:03:09 2008
 PID: 24196 -- Notice: 5:          and local gid: 9767
Failure: ping successful
TIME: Thu Jun  5 00:03:09 2008
 PID: 24196 -- Failure: ping successful
TIME: Thu Jun  5 00:03:58 2008
 PID: 25822 -- Notice: 6: /usr/local/vdt-1.8.1/globus/sbin/globus-gatekeeper pid=25822 starting at Thu Jun  5 00:03:58 2008
```

A typical entry in the web services container log:

```
2008-03-18 10:19:17,533 INFO  gt4.VOMSUserInfo [RunQueueThread_10,retrieveVOMSUserInfo:132] PRIMA: No attributes
found ... assuming grid proxy.
2008-03-18 10:19:17,536 INFO  gt4.OSGAuthorization [RunQueueThread_10,isPermitted:134] PRIMA: Contacting  authz service
(https://gums.fnal.gov:8443/gums/services/GUMSAuthorizationServicePort) for for -
UserDN:/O=GermanGrid/OU=AEI/CN=Thomas Radke - HostDN/DC=org/DC=doegrids/OU=Services/CN=fermigrid1.fnal.gov -
FQAN:null - FQANIssuer:null
2008-03-18 10:19:17,537 WARN  client.SAMLAuthZClientBase [RunQueueThread_10,createFQANEvidenceFromString:218] fqan
and fqanIssuer information must be provided for evidence element to be created
2008-03-18 10:19:17,890 INFO  gt4.OSGAuthorization [RunQueueThread_10,isPermitted:154] PRIMA: PERMIT - for -
UserDN:/O=GermanGrid/OU=AEI/CN=Thomas Radke - HostDN/DC=org/DC=doegrids/OU=Services/CN=fermigrid1.fnal.gov -
FQAN:null - FQANIssuer:null from authz service (https://gums.fnal.gov:8443/gums/services/GUMSAuthorizationServicePort)  is
assigned local account (ligo)
2008-03-18 10:19:17,890 INFO  authorization.ServiceAuthorizationChain [RunQueueThread_10,authorize:285] Authorized
"/O=GermanGrid/OU=AEI/CN=Thomas Radke" to invoke
"{http://www.globus.org/namespaces/2004/10/rft}createReliableFileTransfer".
2008-03-18 10:19:18,589 DEBUG authorization.GridMapAuthorization [RunQueueThread_10,isPermitted:99] Grid map authz
2008-03-18 10:19:18,590 DEBUG authorization.GridMapAuthorization [RunQueueThread_10,isPermitted:124] Service
ReliableFileTransferService
2008-03-18 10:19:18,591 DEBUG authorization.GridMapAuthorization [RunQueueThread_10,isPermitted:163] Peer
"/O=GermanGrid/OU=AEI/CN=Thomas Radke" authorized as "ligo"
2008-03-18 10:19:18,591 INFO  authorization.ServiceAuthorizationChain [RunQueueThread_10,authorize:285] Authorized
"/O=GermanGrid/OU=AEI/CN=Thomas Radke" to invoke "{http://www.globus.org/namespaces/2004/10/rft}subscribe".
2008-03-18 10:19:19,015 DEBUG authorization.GridMapAuthorization [RunQueueThread_10,isPermitted:99] Grid map authz
2008-03-18 10:19:19,016 DEBUG authorization.GridMapAuthorization [RunQueueThread_10,isPermitted:124] Service null
```

```
2008-03-18 10:19:19,016 DEBUG authorization.GridMapAuthorization [RunQueueThread_10,isPermitted:163] Peer
"/O=GermanGrid/OU=AEI/CN=Thomas Radke" authorized as "ligo"
2008-03-18 10:19:19,017 INFO  authorization.ServiceAuthorizationChain [RunQueueThread_10,authorize:285] Authorized
"/O=GermanGrid/OU=AEI/CN=Thomas Radke" to invoke "{http://www.globus.org/namespaces/2004/10/rft}start".
2008-03-18 10:19:21,098 ERROR service.TransferWork [WorkThread-26,run:724] Terminal transfer error:
 [Caused by: Server refused performing the request. Custom message: Bad password. (error code 1) [Nested exception message:
Custom message: Unexpected reply: 530-Login incorrect. : gridmap.c:globus_gss_assist_map_and_authorize:1932:
530-Error invoking callout
530-globus_callout.c:globus_callout_handle_call_type:727:
530-The callout returned an error
530-SAZ-gt3.2.c:Globus Gridmap Callout:237:
530-Gridmap lookup failure: Could not map /O=GermanGrid/OU=AEI/CN=Thomas Radke
530-
530 End.]]
```

SAZ Client for gLExec:

The gLExec application is adapted from apache's suexec, and is designed to be used by VO's that run pilot jobs in order to provide a level of separation between the submitter of a pilot job (aka the pilot user) and the submitter of the workload job (aka the workload user) that is downloaded and run by the pilot job on behalf of the workload user. The typical job is a PANDA or Condor Glide-in that calls the gLexec application to authenticate the workload, verify that an authorized user from the VO submitted the workload, and then perform the appropriate su calls to launch the workload as the corresponding workload user.

The SAZ client for gLExec is the sazclient application. The syntax is:

```
sazclient –x <proxy>
```

or

```
sazclient -file <conffile> <certfile>
```

The former usage presumes that there is a configuration file in conf/sazc.conf

The SAZ gLExec client stores the SAZ communication and decision information in /var/log/glexec/gums_interface.log

<insert typical sazclient log entry>

SAZCLIENT utility:

The sazclient utility may also be used as an interactive utility to check and verify acceptance of Grid user credentials. The syntax is:

```
sazclient –s [server] –p [port] –x [proxy]
```

SAZ Client for gPlazma:

There is a SAZ client plugin available for gPlazma. Contact Ted Hesselroth for this client.

SAZ Command Line Administration Script:

The SAZ command line administration script is normally located in:

$VDT_LOCATION/saz/admin/saz-admin-script.sh

At Fermilab, access to this script is granted to members of FCIRT through entries in the .k5login for the saz-admin account on systems fg5x4 and fx6x4.

The saz-admin-script has the following capabilities:

- Browse the contents of the SAZ server MySQL database based on DN, VO, Role or CA.
- Modify the enabled or trusted flags in one or more records in the MySQL database.
- Insert new records into the MySQL database.

SAZ Web Administration Interface:

The SAZ web administration interface is currently in the process of development.