EGEE

EGEE User's Guide

VOMS ADMIN SERVICE

Document identifier:	**EGEE-JRA3-TEC-572406-VOMS-ADMIN**
Date:	**May 11, 2006**
Activity:	**JRA3: Security**
Document status:	**DRAFT**
Document link:	**https://edms.cern.ch/document/572406/**

Abstract: This user's guide explains how to use the VOMS Admin service.

Delivery Slip

	Name	Partner	Date	Signature
From				
Reviewed by				
Approved by				

Document Change Log

Issue	Date	Comment	Author
1.0	2005-03-03	Initial version	Károly Lőrentey
1.1	2005-03-11	Typos fixed	Károly Lőrentey
1.2	2005-03-14	Added Problems & Caveats section and Acknowledgements	Károly Lőrentey
1.3	2005-03-21	Removed bogus `<Realm>` line in server.xml sample	Károly Lőrentey
1.4	2005-03-22	Enhanced section 4.2.	Károly Lőrentey
1.5	2005-03-31	Fix some typographical problems	Károly Lőrentey

Document Change Record

Issue	Item	Reason for Change

CONTENTS

1. INTRODUCTION

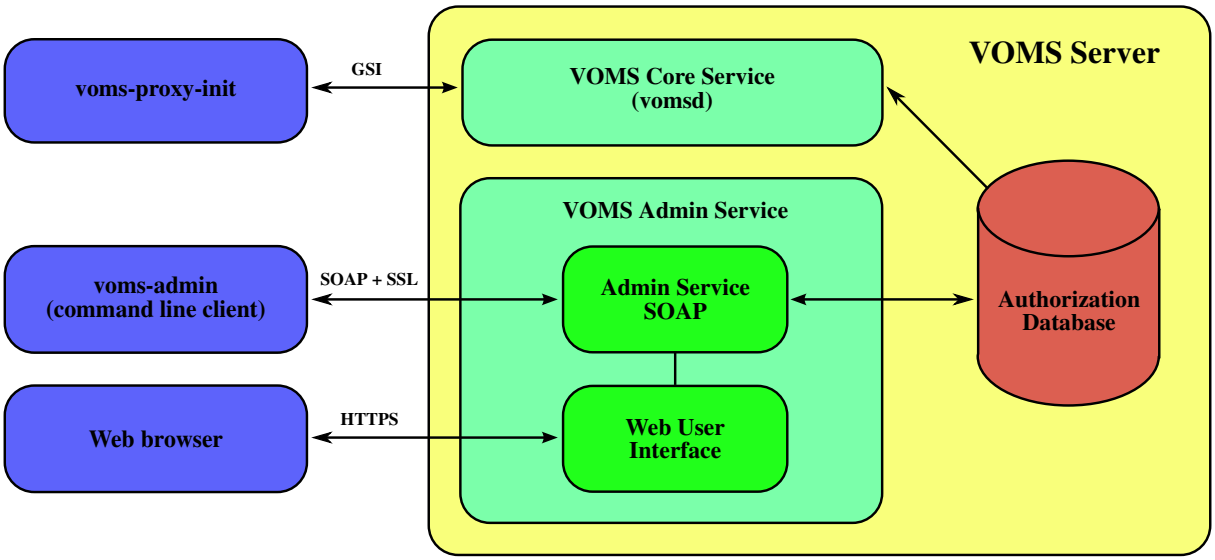
The VOMS Admin service is a web application providing tools for administering member databases for VOMS, the Virtual Organization Membership Service. VOMS serves as a central repository for user authorization information, providing support for sorting users into a general group hierarchy, keeping track of their roles, etc. Its functionality may be compared to that of a Kerberos KDC server.

VOMS Admin provides an intuitive web user interface for daily administration tasks, and a SOAP interface for remote clients. (The entire functionality of the VOMS Admin service is accessible via the SOAP interface.) The Admin package includes a simple command-line SOAP client that is useful for automating frequently occuring batch operations, or simply to serve as an alternative to the full-blown web interface. It is also useful for bootstrapping the service.

1.1. SERVICE ARCHITECTURE

The figure below show a high-level overview of the main components of a VOMS server. The Admin component implements a comprehensive SOAP application program interface for VO membership management. Both the command line client and the builtin web user interface use this API to perform their tasks.

The `voms-proxy-init` command contacts the standalone vomsd process (a service different from VOMS Admin) that queries the authorization database and generates the actual VOMS attribute certificates



1.2. INTERACTIONS WITH OTHER SERVICES

The VOMS Admin service is not capable of generating VOMS Attribute Certificates itself, so it is not able to provide VO login services. It relies on a separate service (*org.glite.security.voms*) to do this task. In this document, we will refer to this service as the *VOMS Core service*, or simply *core service* for brevity.

For small VOs without a heavy user load, it may be convenient to have the VOMS server functionality provided entirely by a single service; for this purpose, interfaces are defined in VOMS Admin that emulate the functionality of the core service. However, these interfaces are not yet implemented in the current release.

2. QUICKSTART GUIDE

This section provides a step-by-step guide to create a new VO server.

2.1. PREREQUISITES

2.1.1. TOMCAT

Make sure you have a Trustmanager-enabled Tomcat instance running. In your server.xml file, you should have a configuration similar to the following snippet:

```
<Server port="8055" shutdown="SHUTDOWN">
  ...
  <Service name="Catalina">

    <!-- Unauthenticated port, primarily for bootstrapping the VO service. -->
    <Connector port="8080"
               maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
               enableLookups="false" redirectPort="8443" acceptCount="100"
               debug="0" connectionTimeout="20000"
               disableUploadTimeout="true" />

    <!-- Client-authenticated port. -->
    <Connector port="8443"
               maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
               enableLookups="false" disableUploadTimeout="true"
               acceptCount="100" debug="0" scheme="https" secure="true"
               sSLImplementation="org.glite.security.trustmanager.tomcat.TMSSLImplementation"
               sslCAFiles="/etc/grid-security/certificates/*.0"
               crlFiles="/etc/grid-security/certificates/*.r0"
               sslCertFile="/etc/grid-security/hostcert.pem"
               sslKey="/etc/grid-security/hostkey.pem"
               log4jConfFile="/opt/glite/externals/tomcat-5.0.28//conf/log4j-trustmanager.properties"
               clientAuth="want" sslProtocol="TLS" />

    <Engine name="Catalina" defaultHost="localhost">
      <Logger className="org.apache.catalina.logger.FileLogger" />
      <Host name="localhost" appBase="webapps" unpackWARs="true" />
      ...
    </Engine>
  </Service>
</Server>
```

It is a good idea to set the `clientAuth` attribute to `"want"` in your secure connector, because it will make for better error reports if you (or your VO users) forget to import their client certificate in their browsers. The service will handle unauthenticated clients gracefully, and will also work fine if you leave this setting at its default `"true"` value.

2.1.2. MYSQL

The service requires a MySQL server running. You will need to know the MySQL database administrator password to create a new VO.

The MySQL service must be accessible via TCP, and must have InnoDB support enabled.

2.1.3. HOST CERTIFICATE

You must have a host or service certificate issued by a CA that is trusted by your VO in order to run a VOMS server. Tomcat must be given the rights to read the private key of this certificate.

2.2. CREATING A NEW VO

You must use the `voms-admin-configure` script to create a new VO.

```
$GLITE_LOCATION/sbin/voms-admin-configure install
        --vo ⟨VO name⟩
        --port ⟨core service port number⟩
        --dbapwd ⟨MySQL password⟩
        --smtp-host ⟨SMTP relay host⟩
        --mail-from ⟨Sender address for service-generated emails⟩
```

The command creates and initializes a VOMS database, and configures both the core service and the admin service for it. The required options are described below:

- The ⟨*VO name*⟩ must consist of alphanumeric characters, hyphens, dots and/or underscores. It may be of any length below 255 characters.
- The ⟨*core service port number*⟩ is the port number that the `vomsd` daemon should listen on. It must be different for each VO that you install on the same host; it is usually chosen sequentially from 15000.
- The ⟨*MySQL password*⟩ is the database password of the MySQL root user. (Note that this is not the same as the `root` login account.) You need to ask the person who installed the database for this password.
- The ⟨*SMTP relay host*⟩ parameter should be set to the hostname of your institution's SMTP submission service. The host must have TCP port 25 open and must behave as an open relay.
- The ⟨*sender address for service-generated emails*⟩ should be a valid email address that leads to a VO administrator.

Section 3.1.1. has more information about the available options to `voms-admin-configure`. To access this information online, run the script with the `--help` option, or run `perldoc voms-admin-configure`. (This trick works for every command-line tool in VOMS Admin.)

Be sure to review the Known Problems section on page 82 if you run into trouble during VO installation.

An example VO installation command is shown below:

```
$GLITE_LOCATION/sbin/voms-admin-configure install
        --vo my-first-vo
        --port 15000
        --dbapwd secretPassw0rd
        --smtp-host smtp.cern.ch
        --mail-from vo-admin@my-first-vo.cern.ch
```

(The above is entered as a single command; it has been broken up into multiple lines only for typographic reasons.)

2.3. STARTING UP THE CORE SERVICE

You are now ready to start up the VOMS core service for your new VO. Type in the following command:

```
$GLITE_LOCATION/etc/init.d/voms start
```

The command starts up a new `vomsd` process for each VO that listens for TCP connections on the port number that you specified on the `voms-admin-configure` command line.

The VO has no users yet, so you can not yet test the functionality of the core service.

Note that the Core service and the VOMS Admin service do not technically depend on each other; you can start and stop them in any order, whenever you want.

2.4. ENABLING YOUR NEW VOMS ADMIN INSTANCE

It is now time to start up an Admin service instance. Type the following command to deploy your new service in Tomcat:

```
$GLITE_LOCATION/etc/init.d/voms-admin start
```

(If you have created other VOs and want to start only some of them, list the VO names to start at the end of the command.)

Provided that Tomcat is running, you should now have a VOMS Admin service deployed, and ready to serve requests. If you forgot to start Tomcat, do it now.

You do not need to type in the above command again unless you explicitly undeploy the service later, or create new VOs. Tomcat will automatically remember to run your VOMS Admin service across server reboots.

2.5. TESTING THE SERVICE

This section assumes that you can run a browser on the server host. To test the service remotely from a UNIX or Linux workstation, you can use SSH port forwarding:

`ssh -R 8080:localhost:8080` ⟨*VOMS server host*⟩

Alternatively, you may replace the common `http://localhost:8080` URL prefix below with `https://`⟨*VOMS server host*⟩`:8443`. Note that some of the above URLs won't work in this case until you set yourself as a VO administrator.

1. Test if Tomcat is running with a web browser:
   `http://localhost:8080/`
2. Test if the service started up properly:
   `http://localhost:8080/voms/`⟨*VO name*⟩`/`
   (You should get a HTML welcome page.)
3. Try the admin interface:
   `http://localhost:8080/voms/`⟨*VO name*⟩`/webui/Admin`

4. Try a test SOAP method:
   `http://localhost:8080/voms/⟨VO name⟩/services/VOMSCompatibility?method=getGridmapUsers`
   (This should result in an XML file, not a HTML page. There should be no obvious error messages visible.)
5. Get a list of installed VOs:
   `http://localhost:8080/vomses/`

If something went wrong, the Tomcat and/or VOMS Admin logfiles will usually help you to discover the cause of the problem. They are located in the Tomcat log directory, `$CATALINA_HOME/logs`. The VOMS logfile is named after the VO name: `$CATALINA_HOME/logs/voms-admin.`⟨*VO Name*⟩`.log`.

2.6. ADDING YOURSELF AS A VO ADMINISTRATOR

In order to complete the VOMS setup procedure, you must add yourself as a VO administrator. Once you have done this, you will be able to administer your VO remotely by using the web interface or the command line client.

Simply copy your personal certificate file in PEM format to the VOMS server, and then type in the following command:

```
$GLITE_LOCATION/bin/voms-admin --vo ⟨VO name⟩
     create-user ⟨certificate.pem⟩
     assign-role VO VO-Admin ⟨certificate.pem⟩
```

(Again, the above must be typed as a single command.)

The command adds yourself as a VO user, and puts you in the VO-Admin role that is given VO administrator privileges by default.

If the command succeeds, you will be able to use your remote browser to continue administering the VO. Load your certificate into the browser, and go to the following address:

`https://`⟨*VOMS server host*⟩`:8443/voms/`⟨*VO name*⟩

Alternatively, you may also use the command line client remotely:

`$GLITE_LOCATION/bin/voms-admin --vo` ⟨*VO name*⟩ `--host` ⟨*VOMS server host*⟩ ⟨*commands*⟩ . . .

See section 3.1.2. on page 16 for further information on the available commands.

2.7. DISABLING VOMS ADMIN

If for whatever reason you want to stop a VOMS Admin instance, you may undeploy it from Tomcat by using the following command:

`$GLITE_LOCATION/etc/init.d/voms-admin stop` ⟨*VO name*⟩

(If you omit the ⟨*VO name*⟩ option, the command will undeploy all VOMS Admin instances.) Tomcat should automatically recognize the change and undeploy the service in a few seconds.

Alternatively, stopping Tomcat will stop all VOMS Admin services as well.

2.8. DELETING A VO

To delete a VO, you must run the `voms-admin-configure` command again, with slightly different parameters:

```
$GLITE_LOCATION/sbin/voms-admin-configure remove
        --vo ⟨VO name⟩
        --dbapwd ⟨MySQL password⟩
```

The command will delete the VO database and remove all VO configuration files for the given ⟨*VO name*⟩. Be careful not to delete important information; the command makes no backups.

3. REFERENCE GUIDE

3.1. COMMAND LINE INTERFACES

In this section, we list a collection of manual pages for the various command-line utilities of VOMS Admin. The following tools are described:

voms-admin The *voms-admin* script provides a basic command-line user interface for VO administration. It provides an alternative to the web interface that is built into the VOMS Admin service that is more convenient for simple batch operations.

voms-admin-configure The *voms-admin-configure* command is used for creating and deleting VOs. It is xusually the first tool that you will need to use after installing the VOMS Admin packages.

init-voms-admin The VOMS Admin init script (named *$GLITE_LOCATION/etc/init.d/voms-admin*) is responsible for deploying and undeploying the web applications that implement VOMS Admin. You will need to use it when you want to start up the service, or when you want to shut it down.

voms-ldap-sync The *voms-ldap-sync* script is useful if you already have a VO membership database in LDAP, and want to maintain a copy of it in the VOMS database.

cron-voms-ldap-sync This is a cron job that is used to automate the LDAP synchronization.

voms-db-dump You can use this script to make a backup of a VO database. It creates a single database dump file.

voms-db-load This script restores a VO database from a dump file created by *voms-db-dump*.

voms-db-upgrade If you have an VO database created by an ancient (2001-2002) VOMS version, you can use this script to upgrade it to a format that is understood by recent VOMS/VOMS Admin releases. You don't normally need to use this script at all.

3.1.1. VOMS-ADMIN-CONFIGURE

SYNOPSIS

```
voms-admin-configure install --vo foobar.example.org --dbtype mysql
                     --dbusername foobar --dbpassword secret
                     --dbname foobar

voms-admin-configure remove --vo NAME

voms-admin-configure update --vo NAME [OPTIONS]

voms-admin-configure upgrade
```

Set up a new VO or remove an existing one.

OPTIONS

General options

```
-h, --help            Print this help message and exit.
-V, --version         Print version string.
-v, --verbose         Print more messages.

--vo VONAME           Install or delete the named VO.
```

Options for configuring the admin service (ignored for "remove")

```
--admincert CERTFILE  The certificate file of an initial VO administrator.
                      The VO will be set up so that this user has full
                      VO administration privileges.

--mail-from FROM      Set the address of the VO administration mailbox.
                      (There is no default; you must specify a valid
                      email address that reaches the VO administrators.)

--smtp-host HOST      Submit service-generated emails to this host.  (There is
                      no default for this option.  Use "localhost" if you have
                      an fully configured SMTP server running on this host.
                      Otherwise specify the hostname of a working SMTP
                      submission service.)
```

Options for configuring the core service (ignored for "remove")

```
--port NUMBER         The port that the VOMS core service should use.  (There
                      is no default.  The port number must be different for
                      each VO.  By convention, port numbers are allocated
                      starting with 15000.)
```

Database options

```
--dbtype TYPE          Database type: "mysql" or "oracle".
                       (Default is "mysql".)

--dbname NAME          Database name for the VO's database account.
--dbusername NAME      Database user name for the VO's database account.
--dbpassword PWD       Database password for the VO's database account.

--dbhost HOST          Hostname of the database server.
                       (Default is "localhost".)

--dbport PORT          Port number of the database server.
                       (Default is 1521 (Oracle) or 3306 (MySQL).)

--deploy-database      For install, clean out and (re)create database schema.
                       The current database contents will be lost. (Default is
                       --deploy-database when the VO is created from scratch,
                       otherwise --skip-database.)

--undeploy-database    For remove, clean out database by dropping all
                       database tables.

--skip-database        Don't touch the database contents.  (For installation,
                       assume that the database already contains valid VO data.
                       For removal, leave the database installed.)
```

If you use the above options, then you must make sure that the database account is set up and accessible before running this script.

Old-style MySQL-specific options (not supported on Oracle)

```
--createdb             Automatically create database.

--dbauser USER         Database userid of the MySQL administrator account.
                       (Default is root.  Implies --createdb.)

--dbapwd PASSWORD      The password of the MySQL administrator account.
                       (Implies --createdb.)

--dbapwdfile FILE      The location of a one-line file containing the DB
                       administrator password.  (Implies --createdb.)

--mysql-command PATH   The path to the "mysql" executable.
                       (Default is "/usr/bin/mysql".)

--mysql-host HOST      Obsolete alias for --dbhost.
--mysql-port PORT      Obsolete alias for --dbport.
```

If you use the above options, then you don't need to specify any of the --db* options. The script will automatically create the database account using the supplied DBA password.

Additional options for special effects

```
--code CODE           An integer code for the core service that is
                      different for each VO installation using the
                      same server certificate.  Used for generating
                      the serial numbers of the attribute
                      certificates. (Default is the value of --port.)

--libdir PATH         The directory that contains the database access
                      libraries of the VOMS core service.
                      (Default is $GLITE_LOCATION/lib.)

--sqlloc FILE         Full path to the database access library for the VOMS
                      core service.  (In case --libdir is not enough.)

--config-owner USER   The UNIX user that should own all configuration files.
                      (Default is the effective userid of the script.)

--tomcat-group GROUP  The UNIX group that Tomcat is run under.
                      (Default is "tomcat5", "tomcat4", or "tomcat".)

--voms-group GROUP    The UNIX group that the VOMS core service is run under.
                      (Default is "voms", if it exists.)

--hostname FQDN       The fully qualified domain name of this host.
                      (Useful if you want to use an alias instead.)

--openssl COMMAND     The path to the openssl executable used to interpret
                      certificates. (Default is "$GLOBUS_LOCATION/bin/openssl"
                      or "/usr/bin/openssl".)

--cert FILENAME       Override $X509_USER_CERT.
--key FILENAME        Override $X509_USER_KEY.
--certdir DIR         Override $X509_CERT_DIR.
```

The script must have write access to the $GLITE_LOCATION_VAR directory.

DESCRIPTION

You can create new VOs and remove existing VOs using this program.

The input/templates directory is:

```
$GLITE_LOCATION/etc/voms-admin
```

The output/configuration directories are:

```
$GLITE_LOCATION_VAR/etc/voms-admin
$GLITE_LOCATION/etc/voms
```

The only exception is when the vomsd package is already installed and configured. In this case the necessary configuration settings are taken from the already configured VOMS instance.

COMMANDS

install

With the **install** command one can set up a new VO by configuring a VOMS instance for it. The command will create a new MySQL database, populate it with the database users and tables required by the VOMS service, and set up a new VOMS instance for it. Both the VOMS core service and the Admin service are configured.

As a special case, if there is already a VOMS service configured for the given VO name, then the script will only create configuration for the Admin service. It will not touch the VOMS database or the core service configuration in this case.

remove

The **remove** command deletes a VO, including service configuration and database contents. Be careful not to delete important data by using this command.

upgrade

Upgrade all VO configurations for a new VOMS Admin release. This command is usually automatically executed on RPM upgrade.

AUTHORS

Akos Frohner (*Akos.Frohner@cern.ch*)

Karoly Lorentey (*Karoly.Lorentey@cern.ch*)

MAINTAINERS

Karoly Lorentey (*Karoly.Lorentey@cern.ch*)

COPYRIGHT

SEE ALSO

voms-admin

3.1.2. VOMS-ADMIN

SYNOPSIS

```
voms-admin [OPTIONS] --vo=NAME [-h HOST] [-p PORT] COMMAND PARAM...
voms-admin [OPTIONS] --url=URL COMMAND PARAM...
```

OPTIONS

```
--help              Print this short help message.
--help-commands     Print a list of available commands, then exit.
--version           Print version string.

-v, --verbose       Print more messages.
-q, --quiet         Print less messages.

--nousercert        Don't extract DNs from supplied certificates.
--usercert FILE     Extract DN parameters from FILE.
--usercert myself   Extract DN parameters from ~/.globus/usercert.

--separator CHAR    Use CHAR for separating fields in the output.
                    (Default is '|' (pipe).)
--nullstring STRING Use STRING to describe null values in the output.
                    (Default is an empty string.)

--noheader          Don't print descriptive header line in output.
```

Service access parameters:

```
--vo NAME           Connect to the NAME VO.  (No default.)

-h, --host HOSTNAME Use the VOMS Admin service running on HOSTNAME.
                    (Default is localhost.)

-s, --ssl           Use secure connection to VOMS Admin. (Default.)
                    Requires a valid Grid certificate.
--nossl             Use insecure connection to VOMS Admin.
                    Requires a nonstandard Tomcat configuration.

-p, --port PORT     Use the VOMS Admin service running on PORT.
                    (Default is 8080 or 8443 depending on --nossl.)

-u, --url URL       Connect to the admin service runnig on URL.
                    Example: https://localhost:8443/voms/voname
                    (Overrides --nossl, --host, --port, and --vo.)
```

Examples:

voms-admin –vo MyFavouriteVO list-users

List the users in MyFavouriteVO running on localhost.

voms-admin –host foobar.cern.ch –vo Foobar list-members /Foobar

List the members of /Foobar in the Foobar VO running on foobar.cern.ch.

DESCRIPTION

The **voms-admin** command provides a simple command line interface for VOMS administrators. It is a simple front-end that calls the SOAP methods provided by the VOMS Admin server.

ALIASES

You may specify `/` or `VO` instead of the VO name for any groupname or container input parameter. The program will look up the VO's real name and replace it in the parameters.

You may specify `myself` instead of a user certificate file name. The program will replace `myself` with `$HOME/.globus/usercert.pem`.

COMMANDS

This client provides the following commands for interaction with the VOMS Admin service. The commands are available in their original mixed-case versions (e.g. getVOName) as well as the dash-separated names (e.g. get-vo-name) displayed below. Some people find the dash-separated versions easier to type.

If you use a dash (-) in place of a parameter on the command line, then the rest of the commands and parameters will be read from standard input. The lines from the standard input may contain comments (starting with # to the end of line) and quoted strings (between " characters). Quotation marks (") can be escaped by backslash: \".

If you use the –nousercert option to use DN and CA strings instead of real certificates, then all parameters designated USER below must be replaced with DN CA. Otherwise replace USER with the name of a file that contains the user's certificate.

get-vo-name

Returns the name of the Virtual Organization.

list-users

Lists all the registered users in this VO.

create-user CERTIFICATE.PEM

Registers a new user in VOMS. If you use the –nousercert option, then four parameters are required (DN CA CN MAIL) to create the user. Otherwise these parameters are extracted automatically from the certificate.

delete-user USER

Deletes a user from VOMS, including all their attributes and membership information.

list-cas

Lists the certificate authorities accepted by the VO.

list-roles

Lists the roles defined in the VO.

create-role ROLENAME

Creates a new role.

delete-role ROLENAME

Deletes a role.

list-capabilities

Lists the capabilities in this VO. Note that support for capabilities is limited at the moment.

create-capability CAPABILITY

Creates a new capability. Note that support for capabilities is limited at the moment.

delete-capability CAPABILITY

Deletes a capability. Note that support for capabilities is limited at the moment.

list-groups

List all the groups in the VO.

list-sub-groups GROUPNAME

List the subgroups of GROUPNAME.

create-group PARENT GROUPNAME

Creates a new GROUPNAME group under PARENT.

delete-group GROUPNAME

Deletes a group.

add-member GROUPNAME USER

Adds USER to the GROUPNAME group.

remove-member GROUPNAME USER

Removes USER from the GROUPNAME group.

list-members GROUPNAME

Lists all members of a group.

assign-role GROUPNAME ROLENAME USER

Assigns role ROLENAME to USER in group GROUPNAME.

dismiss-role GROUPNAME ROLENAME USER

Dismisses role ROLENAME from USER in group GROUPNAME.

list-users-with-role GROUPNAME ROLENAME

Lists all users with ROLENAME in GROUPNAME.

assign-capability CAPABILITY USER

Adds CAPABILITY to USER. Note that support for capabilities is limited at the moment.

dismiss-capability CAPABILITY USER

Removes CAPABILITY from USER. Note that support for capabilities is limited at the moment.

list-users-with-capability CAPABILITY

Lists all users with the given capability. Note that support for capabilities is limited at the moment.

get-acl CONTAINER

Lists the access control list of a group or role.

You must use the correct syntax for group and rolenames, e.g. "/VO/group", or "Role=role1".

get-default-acl GROUPNAME

Lists the ACL that is applied to newly created subgroups of a group.

add-acl-entry CONTAINER ALLOW OPERATION USER

Adds an ACL entry to the CONTAINER's access control list.

You must use explicit types in the container name ("VO_name/group", "Role=role1" or "Capability=capability2") otherwise the system will not be able to figure out which one is selected.

The entry can "allow" or "deny" (ALLOW) any of the following operations: "all", "create", "delete", "add", "remove", "set-acl", "get-acl", "set-default-acl", "get-default-acl" and "list".

add-default-acl-entry GROUPNAME ALLOW OPERATION USER

Adds an ACL entry to a group's default access control list.

The entry can "allow" or "deny" (ALLOW) any of the following operations: "all", "create", "delete", "add", "remove", "set-acl", "get-acl", "set-default-acl", "get-default-acl" and "list".

remove-acl-entry CONTAINER ALLOW OPERATION USER

Removes an ACL entry from the CONTAINER's access control list.

You must use explicit types in the container name ("VO_name/group", "Role=role1" or "Capability=capability2") otherwise the system will not be able to figure out which one is selected.

The entry can "allow" or "deny" (ALLOW) any of the following operations: "all", "create", "delete", "add", "remove", "set-acl", "get-acl", "set-default-acl", "get-default-acl" and "list".

remove-default-acl-entry GROUPNAME ALLOW OPERATION USER

Removes an ACL entry from the group GROUPNAME default access control list.

The entry can "allow" or "deny" (ALLOW) any of the following operations: "all", "create", "delete", "add", "remove", "set-acl", "get-acl", "set-default-acl", "get-default-acl" and "list".

list-user-groups USER

Lists the groups that USER is a member of.

list-user-roles USER

Lists the roles that USER is assigned.

list-user-capabilities USER

Lists the capabilities of USER. Note that support for capabilities is limited at the moment.

Unimplemented methods of the VOMSAdmin interface: setACL, setDefaultACL, getUser, setUser, getGroupPath.

AUTHORS

Akos Frohner (*Akos.Frohner@cern.ch*)

Karoly Lorentey (*Karoly.Lorentey@cern.ch*)

MAINTAINERS

Karoly Lorentey (*Karoly.Lorentey@cern.ch*)

COPYRIGHT

SEE ALSO

voms-ldap-sync, gLite::HTTPS

3.1.3. INIT-VOMS-ADMIN

SYNOPSIS

```
$GLITE_LOCATION/etc/init.d/voms-admin COMMAND [VONAMEx...]
```

COMMAND is one of **status**, **start**, **stop** or **restart**.

DESCRIPTION

The **voms-admin** init.d script is used to start and stop the administrative services for VOMS databases, and also to reload the services after a software upgrade.

Without any further arguments the command is applied on all local VOs. If there are VO names specified, then the command will only apply to them.

COMMANDS

status

Prints status information on the services. It is only static information, the script only checks if the appropriate config files are in place, but does not makes a call to the service to see if it is actually running.

start

Starts the service by copying its context file from `${GLITE_LOCATION_VAR}/etc/voms-admin/VONAME/` to the place where Tomcat expects to find it.

stop

Stops the service by removing the context file from the appropriate Tomcat-specific directory.

restart

Stops then restarts the service.

reload

Stops then restarts all VOs that are currently enabled. Useful after a software upgrade.

AUTHORS

Akos Frohner (*Akos.Frohner@cern.ch*)

Karoly Lorentey (*Karoly.Lorentey@cern.ch*)

MAINTAINERS

Karoly Lorentey (*Karoly.Lorentey@cern.ch*)

COPYRIGHT

SEE ALSO

voms-admin-configure

3.1.4. VOMS-LDAP-SYNC

SYNOPSIS

```
voms-ldap-sync --list
voms-ldap-sync [-v] [-n] --vo=NAME
```

OPTIONS

```
--help              Print this short help message.
-V, --version       Print version string.
-v, --verbose       Print more messages.
-q, --quiet         Print less messages.

-n, --dryrun        Do not modify anything, just pretend.

-c, --config FILE   Use the given config file instead of
                    $GLITE_LOCATION/etc/voms-admin/ldap-vo-config.xml

-l, --list          List known VOs.
```

Required parameters

```
--vo NAME           Synchronize the NAME vo.
```

Optional parameters

```
-h,--host HOSTNAME  Use the VOMS Admin service running on HOSTNAME.
                    (Default is localhost.)

-s, --ssl           Use secure connection to VOMS Admin.
                    Requires a valid Grid certificate.

--nossl             Use insecure connection to VOMS Admin
                    Requires a nonstandard Tomcat configuration.

-p, --port NUMBER   Use the VOMS Admin service running on port NUMBER.
                    (Default is 8080 or 8443 depending on --nossl)

-u, --url URL       Connect to the admin service running on URL.
                    Example: https://localhost:8443/voms/voname
                    (Overrides --nossl, --host, --port, and --vo.)
```

Examples

`voms-ldap-sync --list`

Lists known VOs.

`voms-ldap-sync --vo MyFavouriteVO --nossl`

Synchronizes MyFavouriteVO running on localhost.

DESCRIPTION

This program synchronizes an LDAP Authorization server to a VOMS server.

It connects to the LDAP server and to the VOMS administration server and executes the appropriate commands on the latter one to synchronize their states.

It will create and delete the users and groups in VOMS according to the state in the LDAP server. Deletion of non-LDAP groups has to be explicitly requested, since there might be updates made only on the VOMS server, which should not be lost.

The script will **not** modify the LDAP database and it can also be requested to skip modifications on the VOMS server as well. In this case (**-n**) the required modifications are printed to the standard output, so a sysadmin can execute them separately.

AUTHORS

Overhaul by Karoly Lorentey (*Karoly.Lorentey@cern.ch*)

MAINTAINERS

Karoly Lorentey (*Karoly.Lorentey@cern.ch*)

COPYRIGHT

3.1.5. VOMS-DB-DUMP

SYNOPSIS

voms-db-dump [VONAME]

DESCRIPTION

The **voms-db-dump** command is a simple wrapper around the *mysqldump* program. Its main purpose is to simplify the dump process by extracting the DB configuration from the service's config files and calling mysqldump with the appropriate parameters.

OPTIONS

VONAME

The name of the virtual organization that you want to dump.

If no VO is specified then this script will iterate through all the configured VOs.

The name of the database dump for each VO is:

```
voms-VONAME-YYYY-MM-DD.sql
```

AUTHORS

Akos Frohner (*Akos.Frohner@cern.ch*)

Karoly Lorentey (*Karoly.Lorentey@cern.ch*)

MAINTAINERS

Karoly Lorentey (*Karoly.Lorentey@cern.ch*)

COPYRIGHT

SEE ALSO

voms-db-load, voms-db-upgrade

3.1.6. VOMS-DB-LOAD

SYNOPSIS

voms-db-load VONAME DUMPFILE

DESCRIPTION

The **voms-db-load** command is a simple wrapper around the mysql program. Its main purpose is to simplify the load process by extracting the DB configuration from the service's config files and calling mysql with the appropriate parameters.

OPTIONS

VONAME

Mandatory attribute, there is no default value.

DUMPFILE

The name of the file containing the database dump. If no value is given, then the default value is the last filename in the current directory matching the following pattern:

```
voms-VONAME-????-??-??.sql
```

AUTHORS

Akos Frohner (*Akos.Frohner@cern.ch*)

Karoly Lorentey (*Karoly.Lorentey@cern.ch*)

MAINTAINERS

Karoly Lorentey (*Karoly.Lorentey@cern.ch*)

COPYRIGHT

SEE ALSO

voms-db-dump, voms-db-upgrade

3.1.7. VOMS-DB-UPGRADE

SYNOPSIS

voms-db-upgrade [VONAME]

DESCRIPTION

The **voms-db-upgrade** command is a simple wrapper around the *mysql* program. Its main purpose is to simplify the load process by extracting the DB configuration from the service's config files and calling mysql with the appropriate parameters to load in the database upgrade scripts.

The upgrade procedure first determines the database's version, then it invokes the incremental upgrade scripts to bring up the database to the latest version.

OPTIONS

VONAME

The name of the virtual organization, which you want to upgrade.

If no VO is specified then this script will iterate through all the configured VOs.

AUTHORS

Akos Frohner (*Akos.Frohner@cern.ch*)

Karoly Lorentey (*Karoly.Lorentey@cern.ch*)

MAINTAINER

Karoly Lorentey (*Karoly.Lorentey@cern.ch*)

COPYRIGHT

SEE ALSO

voms-db-dump, voms-db-load

3.2. APPLICATION PROGRAM INTERFACES

Package org.glite.security.voms.service

Package Contents *Page*

Classes

CLASSES

Class **ACLEntry**

Represents access control list entries within the VOMS database.

Access control lists (ACLs) provide authorization information within the VOMS database. They consists of a list of principal-operation-allow/deny triplets called ACL entries. An entry allows or denies an operation to a client principal based on its third element. A client is allowed to perform an operation if she has no matching deny entry but at least one allow entry in the relevant ACL.

The principal of an ACL entry may be a VO group or role in this or in another VO, in which case that entry matches a set of clients instead of a single individual client.

The following operations are defined: CREATE, DELETE, ADD, REMOVE, SET_ACL, GET_ACL, SET_DEFAULT_ACL, GET_DEFAULT_ACL, LIST, plus a special wildcard operation ALL, which is a shorthand for all operations.

DECLARATION

```
public class ACLEntry
```

CONSTRUCTOR SUMMARY

ACLEntry() Empty public constructor.

METHOD SUMMARY

getAdminCA() Returns the principal's CA for this ACL entry.
getAdminDN() Returns the principal's DN for this ACL entry.
getOperationName() Returns the operation field of this ACL entry.
isAllow() Returns the allow field of this ACL entry.
setAdminCA(String) Sets the principal's CA for this ACL entry.
setAdminDN(String) Sets the principal's DN for this ACL entry.
setAllow(boolean) Sets the allow field of this ACL entry.
setOperationName(String) Sets the operation field of this ACL entry.

CONSTRUCTORS

- *ACLEntry*
  public **ACLEntry**( )
  - **Description**
    Empty public constructor.

METHODS

- *getAdminCA*
  public String **getAdminCA**( )
  - **Description**
    Returns the principal's CA for this ACL entry.

- *getAdminDN*
  public String **getAdminDN**( )
  - **Description**
    Returns the principal's DN for this ACL entry.

- *getOperationName*
  public String **getOperationName**( )
  - **Description**
    Returns the operation field of this ACL entry.

- *isAllow*
  public boolean **isAllow**( )
  - **Description**
    Returns the allow field of this ACL entry.

- *setAdminCA*
  public void **setAdminCA**(String **ca**)
  - **Description**
    Sets the principal's CA for this ACL entry.

- *setAdminDN*
  public void **setAdminDN**(String **dn**)
  - **Description**
    Sets the principal's DN for this ACL entry.

- *setAllow*
  public void **setAllow**(boolean **allow**)
  - **Description**
    Sets the allow field of this ACL entry.

- *setOperationName*
  public void **setOperationName**(String **operation**)
  - **Description**
    Sets the operation field of this ACL entry.

Class User

Identifies a user in the VOMS database.

DECLARATION

public class User

CONSTRUCTOR SUMMARY

User() Empty default constructor.

METHOD SUMMARY

getCA() Get the distinguished name of the CA that issued the certificate of this user.
getCertUri() Set the common name of this user.
getCN() Get the common name of this user.
getDN() Get the distinguished name of this user.
getMail() Get the email address of the user.
setCA(String) Set the distinguished name of the CA that issued the certificate of this user.
setCertUri(String) Set the URL of the user's certificate.
setDN(String) Set the distinguished name of this user.
setMail(String) Set the email address of the user.

CONSTRUCTORS

- *User*
  public **User**( )
  - **Description**
    Empty default constructor.

METHODS

- *getCA*
  public String **getCA**( )
  - **Description**
    Get the distinguished name of the CA that issued the certificate of this user.
- *getCertUri*
  public String **getCertUri**( )
  - **Description**
    Set the common name of this user. public void setCN(String cn) { } /** Get the URL of the user's certificate.

- *getCN*
  public String **getCN**( )
  - **Description**
    Get the common name of this user.

- *getDN*
  public String **getDN**( )
  - **Description**
    Get the distinguished name of this user.

- *getMail*
  public String **getMail**( )
  - **Description**
    Get the email address of the user.

- *setCA*
  public void **setCA**(String **ca**)
  - **Description**
    Set the distinguished name of the CA that issued the certificate of this user.

- *setCertUri*
  public void **setCertUri**(String **certUri**)
  - **Description**
    Set the URL of the user's certificate.

- *setDN*
  public void **setDN**(String **dn**)
  - **Description**
    Set the distinguished name of this user.

- *setMail*
  public void **setMail**(String **mail**)
  - **Description**
    Set the email address of the user.

EXCEPTIONS

Class VOMSException

General VOMS exception (thrown back to the client by the VOMS interfaces).

DECLARATION

public class VOMSException
extends java.lang.Exception

CONSTRUCTOR SUMMARY

VOMSException()

CONSTRUCTORS

- *VOMSException*
  public **VOMSException**( )

MEMBERS INHERITED FROM CLASS java.lang.Exception

MEMBERS INHERITED FROM CLASS java.lang.Throwable

- public synchronized native Throwable **fillInStackTrace**( )
- public Throwable **getCause**( )
- public String **getLocalizedMessage**( )
- public String **getMessage**( )
- public StackTraceElement **getStackTrace**( )
- public synchronized Throwable **initCause**(Throwable)
- public void **printStackTrace**( )
- public void **printStackTrace**(java.io.PrintStream)
- public void **printStackTrace**(java.io.PrintWriter)
- public void **setStackTrace**(StackTraceElement[])
- public String **toString**( )

Package org.glite.security.voms.service.admin

Package Contents *Page*

Interfaces

INTERFACES

Interface VOMSAdmin

Virtual Organisation Membership Service Administration interface.

DECLARATION

public interface VOMSAdmin

METHOD SUMMARY

addACLEntry(String, ACLEntry) Adds a new entry to an ACL of a container.
addDefaultACLEntry(String, ACLEntry) Manipulates the default ACL, which is applied on every group created as a subgroup of this one.
addMember(String, String, String) Adds a new member to the group.
assignCapability(String, String, String) Assigns a new capability to the user.
assignRole(String, String, String, String) Assigns a new role to the user.
createCapability(String) Creates a new capability.
createGroup(String, String) Creates a new group as a subgroup of an existing group.
createRole(String) Creates a new role.
createUser(User) Creates a new user in the VOMS database.
deleteCapability(String) Deletes a capability.
deleteGroup(String) Deletes a group.
deleteRole(String) Deletes a role.
deleteUser(String, String) Removes a user from the VOMS database.
dismissCapability(String, String, String) Dismisses a capability of a user.
dismissRole(String, String, String, String) Dismisses a role of a user.
getACL(String) Returns the whole ACL associated with a container.
getDefaultACL(String) Manipulates the default ACL, which is applied on every group created as a subgroup of this one.
getGroupPath(String) Returns the absolute ”path” down to this group.
getMajorVersionNumber() Returns the major version number.
getMinorVersionNumber() Returns the minor version number.
getPatchVersionNumber() Returns the patch version number.
getUser(String, String) Returns information about a user in the VOMS database.
getVOName() Return the name of this VO.
listCapabilities() Lists capabilities.
listCapabilities(String, String) Lists capabilities of a user.
listCAs() Lists certificate authorities.
listGroups(String, String) Lists groups of a user.
listMembers(String) Lists members of a group.
listRoles() Lists roles.
listRoles(String, String) Lists roles of a user.
listSubGroups(String) Lists immediate sub-groups of a group.
listUsersWithCapability(String) Lists assigned users of a capability.
listUsersWithRole(String, String) Lists assigned users of a role associated with a group.
removeACLEntry(String, ACLEntry) Removes an existing entry from the ACL.

removeDefaultACLEntry(String, ACLEntry) Manipulates the default ACL, which is applied on every group created as a subgroup of this one.
removeMember(String, String, String) Removes a member of a group.
setACL(String, ACLEntry[]) Replaces the existing ACL on this container.
setDefaultACL(String, ACLEntry[]) Manipulates the default ACL, which is applied on every group created as a subgroup of this one.
setUser(User) Updates auxiliary information about a user in the VOMS database.

METHODS

- *addACLEntry*
  void **addACLEntry**(String **container**, org.glite.security.voms.service.ACLEntry **aclEntry**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Adds a new entry to an ACL of a container.
    Permission: SETACL on the container.
  - **Parameters**
    - container – The container's name.
    - aclEntry – The new access control list entry.

- *addDefaultACLEntry*
  void **addDefaultACLEntry**(String **groupname**, org.glite.security.voms.service.ACLEntry **aclEntry**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Manipulates the default ACL, which is applied on every group created as a subgroup of this one.
  - **Parameters**
    - groupname – The group's name.
    - aclEntry – The new access control list entry.
  - **See also**
    - VOMSAdmin.addACLEntry(java.lang.String, org.glite.security.voms.service.ACLEntry) (in 3.2., page 36)

- *addMember*
  void **addMember**(String **groupname**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Adds a new member to the group. The user must be a member of the parent group.
    Permission: ADD on the group.
  - **Parameters**
    - groupname – The group's name.
    - username – The user's DN.
    - userca – The user's CA.

- *assignCapability*
  void **assignCapability**(String **capability**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException

- **Description**
  Assigns a new capability to the user.
  Permission: ADD on the capability.
- **Parameters**
  - capability – The capability's name.
  - username – The user's DN.
  - userca – The user's CA.

- *assignRole*
  void **assignRole**(String **groupname**, String **rolename**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Assigns a new role to the user. The user must be a member of the parent group.
    Permission: ADD on the role.
  - **Parameters**
    - groupname – The name of the group associated with this assignment.
    - rolename – The role's name.
    - username – The name of the user to add.
    - userca – The CA of the user to add.

- *createCapability*
  void **createCapability**(String **capability**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Creates a new capability. Copies the default ACL list of the VO to the new capability and adds an extra entry for the administrator with full privileges.
    Permission: CREATE on the VO group.
  - **Parameters**
    - capability – The capability to be created.

- *createGroup*
  void **createGroup**(String **parentname**, String **groupname**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Creates a new group as a subgroup of an existing group. Copies the default ACL list of the parent to the new group and adds an extra entry for the administrator with full privileges.
    Permission: CREATE on parent group.
  - **Parameters**
    - parentname – The parent group's name.
    - groupname – The group's name.

- *createRole*
  void **createRole**(String **rolename**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Creates a new role. Copies the default ACL list of the VO Group to the new role and adds an extra entry for the administrator with full privileges.
    Permission: CREATE on the VO group.
  - **Parameters**

* rolename – The role to be added.

- *createUser*
  void **createUser**(org.glite.security.voms.service.User **user**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Creates a new user in the VOMS database.
    Permission: ADD on the VO group.
  - **Parameters**
    * user – The user to be added.

- *deleteCapability*
  void **deleteCapability**(String **capability**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Deletes a capability. Deletes the capability with all the membership information.
    Permission: DELETE on the capability.
  - **Parameters**
    * capability – The capability to be deleted.

- *deleteGroup*
  void **deleteGroup**(String **groupname**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Deletes a group. The operation deletes the group, all of its sub-groups and associated roles with all the membership information.
    Warning: Deleting the VO "group" effectively wipes out the whole database, so use with care!
    Permission: DELETE on the group.
  - **Parameters**
    * groupname – The group's name.

- *deleteRole*
  void **deleteRole**(String **rolename**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Deletes a role. The role is removed with all the membership information.
    Permission: DELETE on the role.
  - **Parameters**
    * rolename – The role to be deleted.

- *deleteUser*
  void **deleteUser**(String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Removes a user from the VOMS database. Deletes all the associated group, role membership information and corresponding ACL entries as well. It is basically a call to removeMember(VO, user).
    Permission: REMOVE on the VO group.

- **Parameters**
  - username – The user's DN.
  - userca – The user's CA.
- **See also**
  - VOMSAdmin.removeMember(java.lang.String,java.lang.String,java.lang.String) (in 3.2., page 43)

- *dismissCapability*
  void **dismissCapability**(String **capability**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Dismisses a capability of a user.
    Permission: REMOVE on the capability.
  - **Parameters**
    - capability – The capability's name.
    - username – The user's DN.
    - userca – The user's CA.

- *dismissRole*
  void **dismissRole**(String **parentname**, String **rolename**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Dismisses a role of a user.
    Permission: REMOVE on the role.
  - **Parameters**
    - parentname – The parent group's name.
    - rolename – The role's name.
    - username – The user's DN.
    - userca – The user's CA.

- *getACL*
  org.glite.security.voms.service.ACLEntry[] **getACL**(String **container**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Returns the whole ACL associated with a container.
    Permission: GETACL on the container.
  - **Parameters**
    - container – The container's name (null is the VO group).
  - **Returns** – The access control list.

- *getDefaultACL*
  org.glite.security.voms.service.ACLEntry[] **getDefaultACL**(String **groupname**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Manipulates the default ACL, which is applied on every group created as a subgroup of this one.
  - **Parameters**
    - groupname – The group's name.

- **Returns** – The access control list.
- **See also**
  - VOMSAdmin.getACL(java.lang.String) (in 3.2., page 39)

- *getGroupPath*
  String[] **getGroupPath**(String **groupname**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Returns the absolute "path" down to this group. The first element is the VO group and the last is the group itself. There is at least one element in this path if the group exists: the VO group.
    Permission:LIST on parent groups.
  - **Parameters**
    - groupname – The group's name (null is the VO group).
  - **Returns** – Path to the group.

- *getMajorVersionNumber*
  int **getMajorVersionNumber**( )
  - **Description**
    Returns the major version number.

- *getMinorVersionNumber*
  int **getMinorVersionNumber**( )
  - **Description**
    Returns the minor version number.

- *getPatchVersionNumber*
  int **getPatchVersionNumber**( )
  - **Description**
    Returns the patch version number.

- *getUser*
  org.glite.security.voms.service.User **getUser**(String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Returns information about a user in the VOMS database. The user attributes are returned in a User object.
    Permission: LIST on the VO group.
  - **Parameters**
    - username – The name of the user to look up.
    - userca – The certificate authority of the user.
  - **Returns** – All information about the user that is known to VOMS.
  - **See also**
    - org.glite.security.voms.service.User (in 3.2., page 31)

- *getVOName*
  String **getVOName**( ) throws org.glite.security.voms.service.VOMSException

  - **Description**
    Return the name of this VO.
    Permission:LIST on the VO group.
  - **Returns** – The name of this VO.

- *listCapabilities*
  String[] **listCapabilities**( ) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists capabilities.
    Permission:LIST on the VO group.
  - **Returns** – List of capabilities.

- *listCapabilities*
  String[] **listCapabilities**(String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists capabilities of a user.
    Permission:LIST on the VO group.
  - **Parameters**
    * username – The user's DN.
    * userca – The user's CA.
  - **Returns** – List of capabilities.

- *listCAs*
  String[] **listCAs**( ) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists certificate authorities.
    Permission:LIST on the VO group.
  - **Returns** – List of certificate authority DNs.

- *listGroups*
  String[] **listGroups**(String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists groups of a user.
    Permission:LIST on the VO group.
  - **Parameters**
    * username – The user's DN.
    * userca – The user's CA.
  - **Returns** – List of groups in this group.

- *listMembers*
  org.glite.security.voms.service.User[] **listMembers**(String **groupname**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists members of a group.
    Permission:LIST on the group.

- **Parameters**
  - groupname – The group's name (null is the VO group).
- **Returns** – List of users in this group.

- *listRoles*
  String[] **listRoles**( ) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists roles.
    Permission:LIST on the VO group.
  - **Returns** – List of roles in this VO.

- *listRoles*
  String[] **listRoles**(String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists roles of a user.
    Permission:LIST on the VO group.
  - **Parameters**
    - username – The user's DN.
    - userca – The user's CA.
  - **Returns** – List of roles in this group.

- *listSubGroups*
  String[] **listSubGroups**(String **groupname**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists immediate sub-groups of a group.
    Permission:LIST on the group.
  - **Parameters**
    - groupname – The group's name (null is the VO group).
  - **Returns** – List of groups in this group.

- *listUsersWithCapability*
  org.glite.security.voms.service.User[] **listUsersWithCapability**(String **capability**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists assigned users of a capability.
    Permission:LIST on the capability.
  - **Parameters**
    - capability – The capability's name.
  - **Returns** – List of users with this capability.

- *listUsersWithRole*
  org.glite.security.voms.service.User[] **listUsersWithRole**(String **groupname**, String **rolename**) throws org.glite.security.voms.service.VOMSException

- **Description**
  Lists assigned users of a role associated with a group.
  Permission:LIST on the role.
- **Parameters**
  - groupname – The group's name.
  - rolename – The role's name.
- **Returns** – List of users for this role.

- *removeACLEntry*
  void **removeACLEntry**(String **container**, org.glite.security.voms.service.ACLEntry **aclEntry**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Removes an existing entry from the ACL.
    Permission: SETACL on the container.
  - **Parameters**
    - container – The container's name.
    - aclEntry – The access control list entry to be removed.

- *removeDefaultACLEntry*
  void **removeDefaultACLEntry**(String **groupname**, org.glite.security.voms.service.ACLEntry **aclEntry**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Manipulates the default ACL, which is applied on every group created as a subgroup of this one.
  - **Parameters**
    - groupname – The group's name.
    - aclEntry – The access control list entry to be removed.
  - **See also**
    - VOMSAdmin.removeACLEntry(java.lang.String, org.glite.security.voms.service.ACLEntry) (in 3.2., page 43)

- *removeMember*
  void **removeMember**(String **groupname**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Removes a member of a group. Also removes the membership information from the group's sub-groups and associated roles of these groups. If it is the VO group, then it will also delete the user with all its ACL entries.
    Permission: REMOVE on the group.
  - **Parameters**
    - groupname – The group's name.
    - username – The user's DN.
    - userca – The user's CA.
  - **See also**
    - VOMSAdmin.deleteUser(java.lang.String,java.lang.String) (in 3.2., page 38)

- *setACL*
  void **setACL**(String **container**, org.glite.security.voms.service.ACLEntry[] **acl**) throws org.glite.security.voms.service.VOMSException

– **Description**
Replaces the existing ACL on this container.
Permission: SETACL on the container.
– **Parameters**
∗ container – The container's name.
∗ acl – The new access control list.

- *setDefaultACL*
void **setDefaultACL**(String **groupname**, org.glite.security.voms.service.ACLEntry[] **aclEntry**) throws org.glite.security.voms.service.VOMSException

  – **Description**
  Manipulates the default ACL, which is applied on every group created as a subgroup of this one.
  – **Parameters**
  ∗ groupname – The group's name.
  ∗ aclEntry – The new access control list.
  – **See also**
  ∗ VOMSAdmin.setACL(java.lang.String,org.glite.security.voms.service.ACLEntry[]) (in 3.2., page 43)

- *setUser*
void **setUser**(org.glite.security.voms.service.User **user**) throws org.glite.security.voms.service.VOMSException

  – **Description**
  Updates auxiliary information about a user in the VOMS database. The new attributes are passed in the User object.
  Permission: ADD on the VO group.
  – **Parameters**
  ∗ user – The user to update.
  – **See also**
  ∗ org.glite.security.voms.service.User (in 3.2., page 31)

Package org.glite.security.voms.service.compatibility

Package Contents *Page*

Interfaces

INTERFACES

Interface VOMSCompatibility

Virtual Organization Membership Service Compatibility service for the mkgridmap utility.

DECLARATION

public interface VOMSCompatibility

METHOD SUMMARY

getGridmapUsers() Returns the DN of the users in the VOMS database.
getGridmapUsers(String) Returns the DN of the users who have the given container in the VOMS database.
getMajorVersionNumber() Returns the major version number.
getMinorVersionNumber() Returns the minor version number.
getPatchVersionNumber() Returns the patch version number.

METHODS

- *getGridmapUsers*
  String[] **getGridmapUsers**( ) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Returns the DN of the users in the VOMS database. It is used by mkgridmap++ to provide compatibility layer with the VO-LDAP database. This method is equivalent to calling getGridmapUsers(String) (in 3.2., page 46)with the VO group name as its parameter.
  - **Returns** – list of DNs

- *getGridmapUsers*
  String[] **getGridmapUsers**(String **container**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Returns the DN of the users who have the given container in the VOMS database. It is used by mkgridmap++ to provide compatibility layer with the VO-LDAP database.
  - **Parameters**
    - container – A fully qualified container name.
  - **Returns** – list of DNs

- *getMajorVersionNumber*
  int **getMajorVersionNumber**( )
  - **Description**
    Returns the major version number.

- *getMinorVersionNumber*
  int **getMinorVersionNumber**( )
  - **Description**
    Returns the minor version number.

- *getPatchVersionNumber*
  int **getPatchVersionNumber**( )
  - **Description**
    Returns the patch version number.

Package org.glite.security.voms.service.core

Package Contents *Page*

Interfaces

INTERFACES

Interface VOMSCore

Virtual Organisation Membership Service Core interface.

DECLARATION

public interface VOMSCore

METHOD SUMMARY

getAttributes(String[]) Returns the user attributes as a list of strings.
getAttributesAsAC(String[]) Returns the user attributes as an Attribute Certificate.
getMajorVersionNumber() Returns the major version number.
getMinorVersionNumber() Returns the minor version number.
getPatchVersionNumber() Returns the patch version number.
getUser() Returns information about a user in the VOMS database.
listCapabilities() Returns a list of capabilities that the client has.
listGroups() Returns a list of groups that the client is a member of, including the VO group.
listRoles() Returns a list of roles that the client has, along with their associated groups.

METHODS

- *getAttributes*
  String[] **getAttributes**(String[] **roles**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Returns the user attributes as a list of strings.
  - **Parameters**
    - roles – Requested roles.
  - **Returns** – List of attributes.

- *getAttributesAsAC*
  byte[] **getAttributesAsAC**(String[] **roles**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Returns the user attributes as an Attribute Certificate.
  - **Parameters**
    - roles – Requested roles.
  - **Returns** – A signed Attribute Certificate

- *getMajorVersionNumber*
  int **getMajorVersionNumber**( )
  - **Description**
    Returns the major version number.

- *getMinorVersionNumber*
  int **getMinorVersionNumber**( )
  - **Description**
    Returns the minor version number.

- *getPatchVersionNumber*
  int **getPatchVersionNumber**( )
  - **Description**
    Returns the patch version number.

- *getUser*
  org.glite.security.voms.service.User **getUser**( ) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Returns information about a user in the VOMS database.
  - **Returns** – All information about the user that is known to VOMS.

- *listCapabilities*
  String[] **listCapabilities**( ) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Returns a list of capabilities that the client has.

- *listGroups*
  String[] **listGroups**( ) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Returns a list of groups that the client is a member of, including the VO group.

- *listRoles*
  String[] **listRoles**( ) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Returns a list of roles that the client has, along with their associated groups.

Package org.glite.security.voms.service.history

Package Contents *Page*

Interfaces

INTERFACES

Interface VOMSHistory

Virtual Organisation Membership Service History interface. This service provides auditing methods for "back-in-time" queries, which could describe the context of a situation based on the state of the database at a given point in time.

DECLARATION

public interface VOMSHistory

METHOD SUMMARY

getACL(String, long) Get the ACL for a container at a given transaction.
getDefaultACL(String, long) Get the default ACL for a group at a given transaction.
getMajorVersionNumber() Returns the major version number.
getMinorVersionNumber() Returns the minor version number.
getModificationsSince(long) Returns the database modifications as a HUGE XML document since the marked transaction.
getPatchVersionNumber() Returns the patch version number.
listMembers(String, long) List members of a container at a given transaction.
toTime(long) Converts a transaction number to time-mark.
toTransaction(Calendar) Converts a time-mark to transaction number.

METHODS

- *getACL*
  org.glite.security.voms.service.ACLEntry[] **getACL**(String **container**, long **transaction**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Get the ACL for a container at a given transaction.
  - **Parameters**
    - container – The tested container's name.
    - transaction – Transaction number.
  - **Returns** – ACL
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.getACL(java.lang.String) (in 3.2., page 39)

- *getDefaultACL*
  org.glite.security.voms.service.ACLEntry[] **getDefaultACL**(String **groupname**, long **transaction**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Get the default ACL for a group at a given transaction.

- **Parameters**
  - groupname – The tested group's name.
  - transaction – Transaction number.
- **Returns** – ACL
- **See also**
  - org.glite.security.voms.service.admin.VOMSAdmin.getDefaultACL(java.lang.String) (in 3.2., page 39)

- *getMajorVersionNumber*
  int **getMajorVersionNumber**( )
  - **Description**
    Returns the major version number.

- *getMinorVersionNumber*
  int **getMinorVersionNumber**( )
  - **Description**
    Returns the minor version number.

- *getModificationsSince*
  String **getModificationsSince**(long **transaction**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Returns the database modifications as a HUGE XML document since the marked transaction. The purpose of this function is to allow slave replicas for requesting incremental updates.
  - **Parameters**
    - transaction – The last known transaction.
  - **Returns** – XML document with all modifications.

- *getPatchVersionNumber*
  int **getPatchVersionNumber**( )
  - **Description**
    Returns the patch version number.

- *listMembers*
  org.glite.security.voms.service.User[] **listMembers**(String **container**, long **transaction**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    List members of a container at a given transaction.
  - **Parameters**
    - container – The tested container's name.
    - transaction – Transaction number.
  - **Returns** – Members of the container.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.listMembers(java.lang.String) (in 3.2., page 41)

- *toTime*
  java.util.Calendar **toTime**(long **transaction**) throws
  org.glite.security.voms.service.VOMSException
  - **Description**
    Converts a transaction number to time-mark. The time-marks are not recorded for every transaction, so the method will return the closest, earlierst time-mark.
  - **Parameters**
    * transaction – Transaction number.
  - **Returns** – Time-mark.

- *toTransaction*
  long **toTransaction**(java.util.Calendar **time**) throws
  org.glite.security.voms.service.VOMSException
  - **Description**
    Converts a time-mark to transaction number.
  - **Parameters**
    * time – Time-mark.
  - **Returns** – Transaction number.

Package org.glite.security.voms.service.request

Package Contents *Page*

Interfaces

Classes

INTERFACES

Interface VOMSRequest

Virtual Organization Membership Service user request interface.

DECLARATION

public interface VOMSRequest

METHOD SUMMARY

allowRequest(long, String) Accept the given request, i.e. do the requested operation.
confirmRequest(long, String, String) Confirm the email address given during the creation of the request.
deleteRequest(long, String) Delete the given request from the database.
denyRequest(long, String) Deny the given request, i.e. don't do the requested operation.
getAllRequests() Return a list of all requests in the database.
getIncompleteRequests() Return a list of all incomplete requests in the database.
getMajorVersionNumber() Returns the major version number.
getMinorVersionNumber() Returns the minor version number.
getMyRequests() Return a list of all requests created by the client.
getPatchVersionNumber() Returns the patch version number.
getPendingRequests() A shorthand for getRequestsInState (null, null, "Undecided").
getRequest(long) Return detailed information about a request.
getRequestsInState(String, String, String) Return a list of requests with the given type, with the given subject container and in the given state.
newAddMemberRequest(String, String, String[]) Create a new request for membership in the specified group.
newCreateUserRequest(User, String, String[]) Create a new request for adding a new user to the VO.
newCreateUserRequestFromContext(String, String, String[]) Create a new request for adding a new user to the VO.

METHODS

- *allowRequest*
  void **allowRequest**(long **id**, String **comment**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Accept the given request, i.e. do the requested operation. If the operation fails, a VOMSException is thrown and the request stays in the queue. If the operation succeeds, a notification is sent to the client which includes the given comment.
  - **Parameters**
    * id – The unique id of the request.
    * comment – A textual comment to the user who requested the operation.

- *confirmRequest*
  void **confirmRequest**(long **id**, String **cookie**, String **comment**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Confirm the email address given during the creation of the request. Only the requester may confirm the address.
  - **Parameters**
    - id – The unique id of the request.
    - cookie – The identifier that was sent to the given email address.
    - comment – A textual comment for this operation.

- *deleteRequest*
  void **deleteRequest**(long **id**, String **comment**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Delete the given request from the database. Only the requester or a VO admin may delete a request.
  - **Parameters**
    - id – The unique id of the request.
    - comment – A textual comment for this operation.

- *denyRequest*
  void **denyRequest**(long **id**, String **comment**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Deny the given request, i.e. don't do the requested operation. The request is closed and a notification is sent to the client which includes the given comment.
  - **Parameters**
    - id – The unique id of the request.
    - comment – A textual comment for the user who requested the operation.

- *getAllRequests*
  ShortRequest[] **getAllRequests**( ) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Return a list of all requests in the database. A shorthand for getRequestsInState (null, null, null).

- *getIncompleteRequests*
  ShortRequest[] **getIncompleteRequests**( ) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Return a list of all incomplete requests in the database.

- *getMajorVersionNumber*
  int **getMajorVersionNumber**( )
  - **Description**
    Returns the major version number.

- *getMinorVersionNumber*
  int **getMinorVersionNumber**( )
  - **Description**
    Returns the minor version number.

- *getMyRequests*
  ShortRequest[] **getMyRequests**( ) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Return a list of all requests created by the client.

- *getPatchVersionNumber*
  int **getPatchVersionNumber**( )
  - **Description**
    Returns the patch version number.

- *getPendingRequests*
  ShortRequest[] **getPendingRequests**( ) throws org.glite.security.voms.service.VOMSException
  - **Description**
    A shorthand for getRequestsInState (null, null, "Undecided").

- *getRequest*
  DetailedRequest **getRequest**(long **id**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Return detailed information about a request.
  - **Parameters**
    - id – The unique id of the request.
  - **Returns** – Detailed information about the request.

- *getRequestsInState*
  ShortRequest[] **getRequestsInState**(String **type**, String **container**, String **state**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Return a list of requests with the given type, with the given subject container and in the given state. Either parameter can be empty, which means return all types, all containers and/or all states, respectively.

- *newAddMemberRequest*
  long **newAddMemberRequest**(String **group**, String **comment**, String[] **parameters**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Create a new request for membership in the specified group. The member's DN and CA are retrieved from the security context (i.e. from the client's certificate).
  - **Parameters**
    - group – The group to request membership in.
    - comment – A free-form comment for the VO administrator about this request.

    - parameters – Optional parameters for the action in an even element String array (key at odd, value at even positions).
  - **Returns** – the unique id of the newly created request (for status queries).

- *newCreateUserRequest*
  long **newCreateUserRequest**(org.glite.security.voms.service.User **user**, String **comment**, String[] **parameters**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Create a new request for adding a new user to the VO. This general form requires an ADD privilege on the VO group.
  - **Parameters**
    - user – Additional info about the user (cn, email, certUri).
    - comment – A free-form comment for the VO administrator about this request.
    - parameters – Optional parameters for the action in an even element String array (key at odd, value at even positions).
  - **Returns** – the unique id of the newly created request (for status queries).

- *newCreateUserRequestFromContext*
  long **newCreateUserRequestFromContext**(String **email**, String **comment**, String[] **parameters**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Create a new request for adding a new user to the VO. The new user's DN and CA are retrieved from the security context (i.e. from the client's certificate).
  - **Parameters**
    - email – An email address to send allow/deny notifications to.
    - comment – A free-form comment for the VO administrator about this request.
    - parameters – Optional parameters for the action in an even element String array (key at odd, value at even positions).
  - **Returns** – the unique id of the newly created request (for status queries).

CLASSES

Class DetailedRequest

A detailed description of a request.

DECLARATION

public class DetailedRequest
extends org.glite.security.voms.service.request.ShortRequest (in 3.2., page 62)

CONSTRUCTOR SUMMARY

DetailedRequest()

METHOD SUMMARY

getAcceptedEvents() Return a list of events that this request accepts in this state.
getActionParameters() Get the parameters of the requested action as an array of strings.
getChronicle() Get the history of this request.
getStatusDescription() Get the status description of this request.
setAcceptedEvents(String[]) Set the list of events that this request accepts in this state.
setActionParameters(String[]) Set the parameters of the requested action as an array of strings.
setChronicle(SOAPChronicleEntry[]) Set the history of this request.
setStatusDescription(String) Set the status description of this request.

CONSTRUCTORS

- *DetailedRequest*
  public **DetailedRequest**( )

METHODS

- *getAcceptedEvents*
  public String[] **getAcceptedEvents**( )
  - **Description**
    Return a list of events that this request accepts in this state.
- *getActionParameters*
  public String[] **getActionParameters**( )
  - **Description**
    Get the parameters of the requested action as an array of strings. The array has an even number of members; even-numbered indices contain parameter names, odd-numbered indices contain parameter values.

- *getChronicle*
  public SOAPChronicleEntry[] **getChronicle**( )
  - **Description**
    Get the history of this request.

- *getStatusDescription*
  public String **getStatusDescription**( )
  - **Description**
    Get the status description of this request.

- *setAcceptedEvents*
  public void **setAcceptedEvents**(String[] **value**)
  - **Description**
    Set the list of events that this request accepts in this state. Only used on the service side.

- *setActionParameters*
  public void **setActionParameters**(String[] **value**)
  - **Description**
    Set the parameters of the requested action as an array of strings. Only used on the service side.

- *setChronicle*
  public void **setChronicle**(SOAPChronicleEntry[] **value**)
  - **Description**
    Set the history of this request. Only used on the service side.

- *setStatusDescription*
  public void **setStatusDescription**(String **value**)
  - **Description**
    Set the status description of this request. Only used on the service side.

MEMBERS INHERITED FROM CLASS org.glite.security.voms.service.request.ShortRequest (in 3.2., page 62)

- public String **getActionDescription**( )
- public String **getContainerName**( )
- public long **getId**( )
- public User **getRequester**( )
- public String **getStatus**( )
- public String **getType**( )
- public void **setActionDescription**(String **value**)
- public void **setContainerName**(String **value**)
- public void **setId**(long **value**)
- public void **setRequester**(org.glite.security.voms.service.User **value**)
- public void **setStatus**(String **value**)
- public void **setType**(String **value**)

Class ShortRequest

Contains a short description of a request.

DECLARATION

public class ShortRequest

ALL KNOWN SUBCLASSES

DetailedRequest (in 3.2., page 60)

CONSTRUCTOR SUMMARY

ShortRequest()

METHOD SUMMARY

getActionDescription() Get the description of the action of this request.
getContainerName() Get the FQCN of the container that this request wants to change.
getId() Get the unique id of this request.
getRequester() Get the requester of this request.
getStatus() Get the status of this request.
getType() Get the type of this request.
setActionDescription(String) Set the description of the action of this request.
setContainerName(String) Set the FQCN of the container that this request wants to change.
setId(long) Set the unique id of this request.
setRequester(User) Set the requester of this request.
setStatus(String) Set the status of this request.
setType(String) Set the type of this request.

CONSTRUCTORS

- *ShortRequest*
  public **ShortRequest**( )

METHODS

- *getActionDescription*
  public String **getActionDescription**( )

  – **Description**
  Get the description of the action of this request.

- *getContainerName*
  public String **getContainerName**( )
  - **Description**
    Get the FQCN of the container that this request wants to change.

- *getId*
  public long **getId**( )
  - **Description**
    Get the unique id of this request.

- *getRequester*
  public org.glite.security.voms.service.User **getRequester**( )
  - **Description**
    Get the requester of this request.

- *getStatus*
  public String **getStatus**( )
  - **Description**
    Get the status of this request.

- *getType*
  public String **getType**( )
  - **Description**
    Get the type of this request.

- *setActionDescription*
  public void **setActionDescription**(String **value**)
  - **Description**
    Set the description of the action of this request. Only used on the service side.

- *setContainerName*
  public void **setContainerName**(String **value**)
  - **Description**
    Set the FQCN of the container that this request wants to change. Only used on the service side.

- *setId*
  public void **setId**(long **value**)
  - **Description**
    Set the unique id of this request. Only used on the service side.

- *setRequester*
  public void **setRequester**(org.glite.security.voms.service.User **value**)
  - **Description**
    Set the requester of this request. Only used on the service side.

- *setStatus*
  public void **setStatus**(String **value**)
  - **Description**
    Set the status of this request.

- *setType*
  public void **setType**(String **value**)
  - **Description**
    Set the type of this request.

Class SOAPChronicleEntry

An entry in a request's past history.

DECLARATION

public class SOAPChronicleEntry

CONSTRUCTOR SUMMARY

SOAPChronicleEntry()

METHOD SUMMARY

getClientCA() Get the CA of the client who initiated the event.
getClientDN() Get the DN of the client who initiated the event.
getComment() Get the comment submitted by the client.
getDescription() Get the description of what happened to the request.
getTimestamp() Get the time the described event happened to the request.
setClientCA(String) Set the CA of the client who initiated the event.
setClientDN(String) Set the DN of the client who initiated the event.
setComment(String) Set the comment submitted by the client.
setDescription(String) Set the description of what happened to the request.
setTimestamp(Calendar) Set the time the described event happened to the request.

CONSTRUCTORS

- *SOAPChronicleEntry*
  public **SOAPChronicleEntry**( )

METHODS

- *getClientCA*
  public String **getClientCA**( )
  - **Description**
    Get the CA of the client who initiated the event.

- *getClientDN*
  public String **getClientDN**( )
  - **Description**
    Get the DN of the client who initiated the event.

- *getComment*
  public String **getComment**( )

– **Description**
Get the comment submitted by the client.

- *getDescription*
public String **getDescription**( )

– **Description**
Get the description of what happened to the request.

- *getTimestamp*
public java.util.Calendar **getTimestamp**( )

– **Description**
Get the time the described event happened to the request.

- *setClientCA*
public void **setClientCA**(String **value**)

– **Description**
Set the CA of the client who initiated the event. Only used on the service side.

- *setClientDN*
public void **setClientDN**(String **value**)

– **Description**
Set the DN of the client who initiated the event. Only used on the service side.

- *setComment*
public void **setComment**(String **value**)

– **Description**
Set the comment submitted by the client. Only used on the service side.

- *setDescription*
public void **setDescription**(String **value**)

– **Description**
Set the description of what happened to the request. Only used on the service side.

- *setTimestamp*
public void **setTimestamp**(java.util.Calendar **c**)

– **Description**
Set the time the described event happened to the request. Only used on the service side.

Package org.glite.security.voms.service.trustedadmin

Package Contents *Page*

Interfaces

INTERFACES

Interface VOMSTrustedAdmin

Virtual Organisation Membership Service Trusted Admin Interface.

This interface is to be used by frontends to VOMS that wish to use the voms-admin service as a backend, but are unable to delegate their clients' credentials to the edg-voms-admin service.

The access control through this interface is done in two levels: first, we check that the frontend (the credential in the SSL context) is a registered frontend. Then we check that the remote client given by the frontend has the necessary privilege to perform the requested operation. Access is denied if either of these checks fails.

This interface is disabled by default. Note that enabling and actively using this interface is a potential security problem, as clients with trusted access can masquarade as any other client. Trusted clients override the normal authentication mechanisms.

DECLARATION

```
public interface VOMSTrustedAdmin
```

METHOD SUMMARY

addACLEntry(String, String, String, ACLEntry) Adds a new entry to an ACL of a container.
addDefaultACLEntry(String, String, String, ACLEntry) Manipulates the default ACL, which is applied on every group created as a subgroup of this one.
addMember(String, String, String, String, String) Adds a new member to the group.
assignCapability(String, String, String, String, String) Assigns a new capability to the user.
assignRole(String, String, String, String, String, String) Assigns a new role to the user.
createCapability(String, String, String) Creates a new capability.
createGroup(String, String, String, String) Creates a new group as a subgroup of an existing group.
createRole(String, String, String) Creates a new role.
createUser(String, String, User) Creates a new user in the VOMS database.
deleteCapability(String, String, String) Deletes a capability.
deleteGroup(String, String, String) Deletes a group.
deleteRole(String, String, String) Deletes a role.
deleteUser(String, String, String, String) Removes a user from the VOMS database.
dismissCapability(String, String, String, String, String) Dismisses a capability of a user.
dismissRole(String, String, String, String, String, String) Dismisses a role of a user.
getACL(String, String, String) Returns the whole ACL associated with a container.
getDefaultACL(String, String, String) Manipulates the default ACL, which is applied on every group created as a subgroup of this one.
getGroupPath(String, String, String) Returns the absolute ”path” down to this group.
getMajorVersionNumber() Returns the major version number.
getMinorVersionNumber() Returns the minor version number.
getPatchVersionNumber() Returns the patch version number.

getUser(String, String, String, String) Returns information about a user in the VOMS database.
getVOName(String, String) Return the name of this VO.
listCapabilities(String, String) Lists capabilities.
listCapabilities(String, String, String, String) Lists capabilities of a user.
listCAs(String, String) Lists certificate authorities.
listGroups(String, String, String, String) Lists groups of a user.
listMembers(String, String, String) Lists members of a group.
listRoles(String, String) Lists roles.
listRoles(String, String, String, String) Lists roles of a user.
listSubGroups(String, String, String) Lists immediate sub-groups of a group.
listUsersWithCapability(String, String, String) Lists assigned users of a capability.
listUsersWithRole(String, String, String, String) Lists assigned users of a role associated with a group.
removeACLEntry(String, String, String, ACLEntry) Removes an existing entry from the ACL.
removeDefaultACLEntry(String, String, String, ACLEntry) Manipulates the default ACL, which is applied on every group created as a subgroup of this one.
removeMember(String, String, String, String, String) Removes a member of a group.
setACL(String, String, String, ACLEntry[]) Replaces the existing ACL on this container.
setDefaultACL(String, String, String, ACLEntry[]) Manipulates the default ACL, which is applied on every group created as a subgroup of this one.
setUser(String, String, User) Updates auxiliary information about a user in the VOMS database.

METHODS

- *addACLEntry*
  void **addACLEntry**(String **delegatedDN**, String **delegatedCA**, String **container**, org.glite.security.voms.service.ACLEntry **aclEntry**) throws org.glite.security.voms.service.VOMSException

  – **Description**
  Adds a new entry to an ACL of a container.
  Permission: SETACL on the container.

  – **Parameters**

  ∗ delegatedDN – The DN of the delegated client.
  ∗ delegatedCA – The CA of the delegated client.
  ∗ container – The container's name.
  ∗ aclEntry – The new access control list entry.

  – **See also**

  ∗ org.glite.security.voms.service.admin.VOMSAdmin.addACLEntry(java.lang.String, org.glite.security.voms.service.ACLEntry) (in 3.2., page 36)

- *addDefaultACLEntry*
  void **addDefaultACLEntry**(String **delegatedDN**, String **delegatedCA**, String **groupname**, org.glite.security.voms.service.ACLEntry **aclEntry**) throws org.glite.security.voms.service.VOMSException

- **Description**
  Manipulates the default ACL, which is applied on every group created as a subgroup of this one.
- **Parameters**
  - delegatedDN – The DN of the delegated client.
  - delegatedCA – The CA of the delegated client.
  - groupname – The group's name.
  - aclEntry – The new access control list entry.
- **See also**
  - VOMSTrustedAdmin.addACLEntry(java.lang.String,java.lang.String,java.lang.String, org.glite.security.voms.service.ACLEntry) (in 3.2., page 69)
  - org.glite.security.voms.service.admin.VOMSAdmin.addDefaultACLEntry(java.lang.String, org.glite.security.voms.service.ACLEntry) (in 3.2., page 36)

- *addMember*
  void **addMember**(String **delegatedDN**, String **delegatedCA**, String **groupname**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Adds a new member to the group. The user must be a member of the parent group.
    Permission: ADD on the group.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
    - groupname – The group's name.
    - username – The user's DN.
    - userca – The user's CA.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.addMember(java.lang.String, java.lang.String,java.lang.String) (in 3.2., page 36)

- *assignCapability*
  void **assignCapability**(String **delegatedDN**, String **delegatedCA**, String **capability**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Assigns a new capability to the user.
    Permission: ADD on the capability.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
    - capability – The capability's name.
    - username – The user's DN.
    - userca – The user's CA.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.assignCapability(java.lang.String, java.lang.String,java.lang.String) (in 3.2., page 36)

- *assignRole*
  void **assignRole**(String **delegatedDN**, String **delegatedCA**, String **groupname**, String **rolename**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException

- **Description**
  Assigns a new role to the user. The user must be a member of the parent group.
  Permission: ADD on the role.
- **Parameters**
  - delegatedDN – The DN of the delegated client.
  - delegatedCA – The CA of the delegated client.
  - groupname – The name of the group associated with this assignment.
  - rolename – The role's name.
  - username – The name of the user to add.
  - userca – The CA of the user to add.
- **See also**
  - org.glite.security.voms.service.admin.VOMSAdmin.assignRole(java.lang.String, java.lang.String,java.lang.String,java.lang.String) (in 3.2., page 37)

- *createCapability*
  void **createCapability**(String **delegatedDN**, String **delegatedCA**, String **capability**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Creates a new capability. Copies the default ACL list of the VO to the new capability and adds an extra entry for the administrator with full privileges.
    Permission: CREATE on the VO group.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
    - capability – The capability to be created.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.createCapability(java.lang.String) (in 3.2., page 37)

- *createGroup*
  void **createGroup**(String **delegatedDN**, String **delegatedCA**, String **parentname**, String **groupname**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Creates a new group as a subgroup of an existing group. Copies the default ACL list of the parent to the new group and adds an extra entry for the administrator with full privileges.
    Permission: CREATE on parent group.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
    - parentname – The parent group's name.
    - groupname – The group's name.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.createGroup(java.lang.String, java.lang.String) (in 3.2., page 37)

- *createRole*
  void **createRole**(String **delegatedDN**, String **delegatedCA**, String **rolename**) throws org.glite.security.voms.service.VOMSException

- **Description**
  Creates a new role. Copies the default ACL list of the VO Group to the new role and adds an extra entry for the administrator with full privileges.
  Permission: CREATE on the VO group.
- **Parameters**
  - delegatedDN – The DN of the delegated client.
  - delegatedCA – The CA of the delegated client.
  - rolename – The role to be added.
- **See also**
  - org.glite.security.voms.service.admin.VOMSAdmin.createRole(java.lang.String) (in 3.2., page 37)

- *createUser*
  void **createUser**(String **delegatedDN**, String **delegatedCA**, org.glite.security.voms.service.User **user**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Creates a new user in the VOMS database.
    Permission: ADD on the VO group.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
    - user – The user to be added.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.createUser(org.glite.security.voms.service.User) (in 3.2., page 38)

- *deleteCapability*
  void **deleteCapability**(String **delegatedDN**, String **delegatedCA**, String **capability**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Deletes a capability. Deletes the capability with all the membership information.
    Permission: DELETE on the capability.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
    - capability – The capability to be deleted.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.deleteCapability(java.lang.String) (in 3.2., page 38)

- *deleteGroup*
  void **deleteGroup**(String **delegatedDN**, String **delegatedCA**, String **groupname**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Deletes a group. The operation deletes the group, all of its sub-groups and associated roles with all the membership information.
    Warning: Deleting the VO "group" effectively wipes out the whole database, so use with care!
    Permission: DELETE on the group.

- **Parameters**
  - delegatedDN – The DN of the delegated client.
  - delegatedCA – The CA of the delegated client.
  - groupname – The group's name.
- **See also**
  - org.glite.security.voms.service.admin.VOMSAdmin.deleteGroup(java.lang.String) (in 3.2., page 38)

- *deleteRole*
  void **deleteRole**(String **delegatedDN**, String **delegatedCA**, String **rolename**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Deletes a role. The role is removed with all the membership information.
    Permission: DELETE on the role.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
    - rolename – The role to be deleted.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.deleteRole(java.lang.String) (in 3.2., page 38)

- *deleteUser*
  void **deleteUser**(String **delegatedDN**, String **delegatedCA**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Removes a user from the VOMS database. Deletes all the associated group, role membership information and corresponding ACL entries as well. It is basically a call to removeMember(VO, user).
    Permission: REMOVE on the VO group.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
    - username – The user's DN.
    - userca – The user's CA.
  - **See also**
    - VOMSTrustedAdmin.removeMember(java.lang.String,java.lang.String, java.lang.String,java.lang.String,java.lang.String) (in 3.2., page 80)
    - org.glite.security.voms.service.admin.VOMSAdmin.deleteUser(java.lang.String, java.lang.String) (in 3.2., page 38)

- *dismissCapability*
  void **dismissCapability**(String **delegatedDN**, String **delegatedCA**, String **capability**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Dismisses a capability of a user.
    Permission: REMOVE on the capability.
  - **Parameters**

    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
    - capability – The capability's name.
    - username – The user's DN.
    - userca – The user's CA.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.dismissCapability(java.lang.String, java.lang.String,java.lang.String) (in 3.2., page 39)

- *dismissRole*
  void **dismissRole**(String **delegatedDN**, String **delegatedCA**, String **parentname**, String **rolename**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Dismisses a role of a user.
    Permission: REMOVE on the role.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
    - parentname – The parent group's name.
    - rolename – The role's name.
    - username – The user's DN.
    - userca – The user's CA.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.dismissRole(java.lang.String, java.lang.String,java.lang.String,java.lang.String) (in 3.2., page 39)

- *getACL*
  org.glite.security.voms.service.ACLEntry[] **getACL**(String **delegatedDN**, String **delegatedCA**, String **container**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Returns the whole ACL associated with a container.
    Permission: GETACL on the container.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
    - container – The container's name (null is the VO group).
  - **Returns** – The access control list.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.getACL(java.lang.String) (in 3.2., page 39)

- *getDefaultACL*
  org.glite.security.voms.service.ACLEntry[] **getDefaultACL**(String **delegatedDN**, String **delegatedCA**, String **groupname**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Manipulates the default ACL, which is applied on every group created as a subgroup of this one.

- **Parameters**
  - delegatedDN – The DN of the delegated client.
  - delegatedCA – The CA of the delegated client.
  - groupname – The group's name.
- **Returns** – The access control list.
- **See also**
  - VOMSTrustedAdmin.getACL(java.lang.String,java.lang.String,java.lang.String) (in 3.2., page 74)
  - org.glite.security.voms.service.admin.VOMSAdmin.getDefaultACL(java.lang.String) (in 3.2., page 39)

- *getGroupPath*
  String[] **getGroupPath**(String **delegatedDN**, String **delegatedCA**, String **groupname**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Returns the absolute "path" down to this group. The first element is the VO group and the last is the group itself. There is at least one element in this path if the group exists: the VO group.
    Permission:LIST on parent groups.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
    - groupname – The group's name (null is the VO group).
  - **Returns** – Path to the group.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.getGroupPath(java.lang.String) (in 3.2., page 40)

- *getMajorVersionNumber*
  int **getMajorVersionNumber**( )
  - **Description**
    Returns the major version number.

- *getMinorVersionNumber*
  int **getMinorVersionNumber**( )
  - **Description**
    Returns the minor version number.

- *getPatchVersionNumber*
  int **getPatchVersionNumber**( )
  - **Description**
    Returns the patch version number.

- *getUser*
  org.glite.security.voms.service.User **getUser**(String **delegatedDN**, String **delegatedCA**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException

- **Description**
  Returns information about a user in the VOMS database.
  Permission: LIST on the VO group.
- **Parameters**
  - delegatedDN – The DN of the delegated client.
  - delegatedCA – The CA of the delegated client.
  - username – The name of the user to look up.
  - userca – The certificate authority of the user.
- **Returns** – All information about the user that is known to VOMS.
- **See also**
  - org.glite.security.voms.service.admin.VOMSAdmin.getUser(java.lang.String, java.lang.String) (in 3.2., page 40)

- *getVOName*
  String **getVOName**(String **delegatedDN**, String **delegatedCA**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Return the name of this VO.
    Permission:LIST on the VO group.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
  - **Returns** – The name of this VO.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.getVOName() (in 3.2., page 40)

- *listCapabilities*
  String[] **listCapabilities**(String **delegatedDN**, String **delegatedCA**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists capabilities.
    Permission:LIST on the VO group.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
  - **Returns** – List of capabilities.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.listCapabilities() (in 3.2., page 41)

- *listCapabilities*
  String[] **listCapabilities**(String **delegatedDN**, String **delegatedCA**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists capabilities of a user.
    Permission:LIST on the VO group.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.

    - delegatedCA – The CA of the delegated client.
    - username – The user's DN.
    - userca – The user's CA.
  - **Returns** – List of capabilities.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.listCapabilities() (in 3.2., page 41)

- *listCAs*
  String[] **listCAs**(String **delegatedDN**, String **delegatedCA**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists certificate authorities.
    Permission:LIST on the VO group.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
  - **Returns** – List of certificate authority DNs.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.listCAs() (in 3.2., page 41)

- *listGroups*
  String[] **listGroups**(String **delegatedDN**, String **delegatedCA**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists groups of a user.
    Permission:LIST on the VO group.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
    - username – The user's DN.
    - userca – The user's CA.
  - **Returns** – List of groups in this group.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.listGroups(java.lang.String, java.lang.String) (in 3.2., page 41)

- *listMembers*
  org.glite.security.voms.service.User[] **listMembers**(String **delegatedDN**, String **delegatedCA**, String **groupname**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists members of a group.
    Permission:LIST on the group.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
    - groupname – The group's name (null is the VO group).
  - **Returns** – List of users in this group.
  - **See also**

* org.glite.security.voms.service.admin.VOMSAdmin.listMembers(java.lang.String) (in 3.2., page 41)

- *listRoles*
  String[] **listRoles**(String **delegatedDN**, String **delegatedCA**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists roles.
    Permission:LIST on the VO group.
  - **Parameters**
    * delegatedDN – The DN of the delegated client.
    * delegatedCA – The CA of the delegated client.
  - **Returns** – List of roles in this VO.
  - **See also**
    * org.glite.security.voms.service.admin.VOMSAdmin.listRoles() (in 3.2., page 42)

- *listRoles*
  String[] **listRoles**(String **delegatedDN**, String **delegatedCA**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists roles of a user.
    Permission:LIST on the VO group.
  - **Parameters**
    * delegatedDN – The DN of the delegated client.
    * delegatedCA – The CA of the delegated client.
    * username – The user's DN.
    * userca – The user's CA.
  - **Returns** – List of roles of the user qualified with their group names.
  - **See also**
    * org.glite.security.voms.service.admin.VOMSAdmin.listRoles() (in 3.2., page 42)

- *listSubGroups*
  String[] **listSubGroups**(String **delegatedDN**, String **delegatedCA**, String **groupname**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists immediate sub-groups of a group.
    Permission:LIST on the group.
  - **Parameters**
    * delegatedDN – The DN of the delegated client.
    * delegatedCA – The CA of the delegated client.
    * groupname – The group's name (null is the VO group).
  - **Returns** – List of groups in this group.
  - **See also**
    * org.glite.security.voms.service.admin.VOMSAdmin.listSubGroups(java.lang.String) (in 3.2., page 42)

- *listUsersWithCapability*
  org.glite.security.voms.service.User[] **listUsersWithCapability**(String **delegatedDN**, String **delegatedCA**, String **capability**) throws org.glite.security.voms.service.VOMSException

- **Description**
  Lists assigned users of a capability.
  Permission:LIST on the capability.
- **Parameters**
  - delegatedDN – The DN of the delegated client.
  - delegatedCA – The CA of the delegated client.
  - capability – The capability's name.
- **Returns** – List of users with this capability.
- **See also**
  - org.glite.security.voms.service.admin.VOMSAdmin.listUsersWithCapability(java.lang.String) (in 3.2., page 42)

- *listUsersWithRole*
  org.glite.security.voms.service.User[] **listUsersWithRole**(String **delegatedDN**, String **delegatedCA**, String **groupname**, String **rolename**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Lists assigned users of a role associated with a group.
    Permission:LIST on the role.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
    - groupname – The group's name.
    - rolename – The role's name.
  - **Returns** – List of users for this role.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.listUsersWithRole(java.lang.String, java.lang.String) (in 3.2., page 42)

- *removeACLEntry*
  void **removeACLEntry**(String **delegatedDN**, String **delegatedCA**, String **container**, org.glite.security.voms.service.ACLEntry **aclEntry**) throws org.glite.security.voms.service.VOMSException
  - **Description**
    Removes an existing entry from the ACL.
    Permission: SETACL on the container.
  - **Parameters**
    - delegatedDN – The DN of the delegated client.
    - delegatedCA – The CA of the delegated client.
    - container – The container's name.
    - aclEntry – The access control list entry to be removed.
  - **See also**
    - org.glite.security.voms.service.admin.VOMSAdmin.removeACLEntry(java.lang.String, org.glite.security.voms.service.ACLEntry) (in 3.2., page 43)

- *removeDefaultACLEntry*
  void **removeDefaultACLEntry**(String **delegatedDN**, String **delegatedCA**, String **groupname**, org.glite.security.voms.service.ACLEntry **aclEntry**) throws org.glite.security.voms.service.VOMSException

  – **Description**
  Manipulates the default ACL, which is applied on every group created as a subgroup of this one.
  – **Parameters**
  * delegatedDN – The DN of the delegated client.
  * delegatedCA – The CA of the delegated client.
  * groupname – The group's name.
  * aclEntry – The access control list entry to be removed.
  – **See also**
  * VOMSTrustedAdmin.removeACLEntry(java.lang.String,java.lang.String, java.lang.String,org.glite.security.voms.service.ACLEntry) (in 3.2., page 79)
  * org.glite.security.voms.service.admin.VOMSAdmin.removeDefaultACLEntry(java.lang.String, org.glite.security.voms.service.ACLEntry) (in 3.2., page 43)

- *removeMember*
  void **removeMember**(String **delegatedDN**, String **delegatedCA**, String **groupname**, String **username**, String **userca**) throws org.glite.security.voms.service.VOMSException

  – **Description**
  Removes a member of a group. Also removes the membership information from the group's sub-groups and associated roles of these groups. If it is the VO group, then it will also delete the user with all its ACL entries.
  Permission: REMOVE on the group.
  – **Parameters**
  * delegatedDN – The DN of the delegated client.
  * delegatedCA – The CA of the delegated client.
  * groupname – The group's name.
  * username – The user's DN.
  * userca – The user's CA.
  – **See also**
  * VOMSTrustedAdmin.deleteUser(java.lang.String,java.lang.String,java.lang.String, java.lang.String) (in 3.2., page 73)
  * org.glite.security.voms.service.admin.VOMSAdmin.removeMember(java.lang.String, java.lang.String,java.lang.String) (in 3.2., page 43)

- *setACL*
  void **setACL**(String **delegatedDN**, String **delegatedCA**, String **container**, org.glite.security.voms.service.ACLEntry[] **acl**) throws org.glite.security.voms.service.VOMSException

  – **Description**
  Replaces the existing ACL on this container.
  Permission: SETACL on the container.
  – **Parameters**
  * delegatedDN – The DN of the delegated client.

* delegatedCA – The CA of the delegated client.
* container – The container's name.
* acl – The new access control list.

– **See also**

* org.glite.security.voms.service.admin.VOMSAdmin.setACL(java.lang.String, org.glite.security.voms.service.ACLEntry[]) (in 3.2., page 43)

- *setDefaultACL*
  void **setDefaultACL**(String **delegatedDN**, String **delegatedCA**, String **groupname**, org.glite.security.voms.service.ACLEntry[] **aclEntry**) throws org.glite.security.voms.service.VOMSException

  – **Description**
  Manipulates the default ACL, which is applied on every group created as a subgroup of this one.

  – **Parameters**

  * delegatedDN – The DN of the delegated client.
  * delegatedCA – The CA of the delegated client.
  * groupname – The group's name.
  * aclEntry – The new access control list.

  – **See also**

  * VOMSTrustedAdmin.setACL(java.lang.String,java.lang.String,java.lang.String, org.glite.security.voms.service.ACLEntry[]) (in 3.2., page 80)
  * org.glite.security.voms.service.admin.VOMSAdmin.setDefaultACL(java.lang.String, org.glite.security.voms.service.ACLEntry[]) (in 3.2., page 44)

- *setUser*
  void **setUser**(String **delegatedDN**, String **delegatedCA**, org.glite.security.voms.service.User **user**) throws org.glite.security.voms.service.VOMSException

  – **Description**
  Updates auxiliary information about a user in the VOMS database. The new attributes are passed in the User object.
  Permission: ADD on the VO group.

  – **Parameters**

  * delegatedDN – The DN of the delegated client.
  * delegatedCA – The CA of the delegated client.
  * user – The user to update.

  – **See also**

  * org.glite.security.voms.service.User (in 3.2., page 31)
  * org.glite.security.voms.service.admin.VOMSAdmin.setUser(org.glite.security.voms.service.User) (in 3.2., page 44)

4. KNOWN PROBLEMS AND CAVEATS

This section lists some of the bugs and problems that are still unresolved in VOMS Admnin. Some of these issues may have been resolved since the time this section was last updated; please refer to the Savannah bug tracking interface at *savannah.cern.ch* for a complete and up to date list of known bugs in the service.

4.1. VOMS ADMIN DOES NOT FUNCTION CORRECTLY WITH THE CORE SERVICE INSTALLL SCRIPT

There is an alternative install script supplied in VOMS Core called install_db that, in theory, should also be able to configure VOMS Admin for a new VO. Unfortunately at the moment it has several serious problems that prevent it from working correctly and/or safely with VOMS Admin. At this time, it is best to follow the instructions in this manual and use the script bundled in VOMS Admin (voms-admin-configure) for installing new VOs. The voms-admin-configure script should properly configure both the Core and the Admin service.

4.2. MySQL DATABASE CREATION GLITCH

Sometimes VO installation with the voms-admin-configure command fails with a MySQL database error 1045 (Access denied for user, with a VOMS-specific database username of the form ⟨*VO name*⟩_adm@localhost).

Creating the database...
    Loading schema
ERROR 1045: Access denied for user: '⟨*VO name*⟩_adm@localhost' (Using password: YES)

Whenever this happens, the first thing you should do is to run voms-admin-configure with the remove option to clean up the half-installed VO that you have just created.

The next step depends on your circumstances. Do you have a MySQL database that clashes with that of your new VO? (I.e., are you creating a whole new database, or do you already have a database whose name matches voms_⟨*VO name*⟩? If you have such a database, please remove it (or choose another VO name) and run voms-admin-configure again. You may also need to remove the following database users:

⟨*VO name*⟩_adm
⟨*VO name*⟩_que
⟨*VO name*⟩_upd
⟨*VO name*⟩_seq

There is a slight chance that you get the database error even if there is no conflicting database. In this case, just try again after removing your half-installed VO with the remove command of voms-admin-configure. There seems to be a bug in some versions of MySQL that affects newly created users. The bug rarely manifests itself, and mostly under certain special conditions (such as while running automated unit tests).

4.3. THE NOTIFICATION MECHANISM FOR VO ADMINISTRATORS IS BROKEN

VOMS Admin includes a simple workflow engine that allows VO administrators (with some Java programming abilities) to define standardised request interfaces for users to apply for VO membership

or to do other common tasks. The package comes with several example workflows for user creation, and includes support for notifying VO administrators when a new decision is to be made by them. Due to a design flaw in this notification mechanism, in certain cases notifications will not be sent to some of the administrators. The problem is further aggravated by a programming error that causes this flaw to affect all administrators in a VO in the default configuration. If you are affected this bug, the easiest way to work around it is to live without admin notification and regularly poll the pending request list on the web interface for new requests.

VO users who submit the requests are not affected by this bug and should be correctly notified by the service.

4.4. UNIMPLEMENTED FEATURES

Support for user expiration and suspension A real-life user registration service must provide support for requiring user re-registration after a certain period of time. It must also allow the administrator to temporarily suspend certain users' VO membership for reasons such as unacceptable behaviour. Currently VOMS Admin implements neither of these features.

Multiple database engine support Most institutions have their own supported relational database engine, in most cases with dedicated support personnel. It is therefore natural for the site running a VOMS server to expect that it would use the institute's official RDBMS solution. VOMS Admin uses a very portable subset of SQL, so it should be readily portable to other database engines than MySQL; however this needs to be tested and support scripts need to be written to handle VO creation on other database platforms.

The History interface for querying audit data VOMS Admin keeps a full change log inside the database in an easily accessible format. A History interface is defined to allow administrators to look at the VO database state at any given point in time and answer such questions as "Was this user a member of group foo on March 14th last year?" or "Which administrator removed Alice from the bar group?". Unfortunately at this time this interface lacks an implementation.

VOMS Admin is not capable of issuing VOMS certificates itself Smaller VOs with a small user base would find it useful to have the entire VOMS functionality provided by a single service. The Admin service defines a SOAP interface for this purpose, but it does not implement it yet.

5. ACKNOWLEDGEMENTS

Károly Lőrentey's work was supported by the National Office of Research and Technology, Budapest, Hungary.