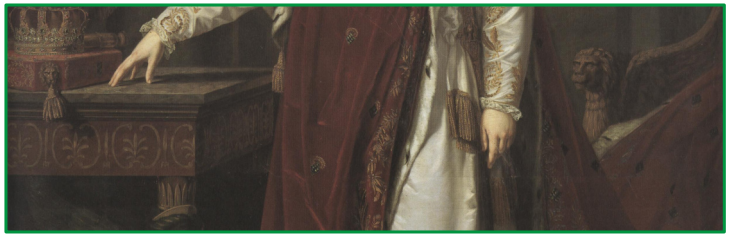
dCache Introduction Course

Overview
Chapters I, II and V

Christoph Anton Mitterer
christoph.anton.mitterer@lmu.de



V. Access Control

Access Control Systems In dCache

dCache contains several access control systems, used with different types of protocols and providing a varying set of features:

- gPlazma + door (NFS 4.1, WebDAV, gsiDCAP, GridFTP, xrootd, SRM)
  The NFS 4.1-, WebDAV-, gsiDCAP-, GridFTP-, xrootd- and SRM-doors implement their own (very similar) access control systems by using the services provided by gPlazma.
- xrootd-ALICE (xrootd)
  This system implements the so called "ALICE security model" and is only used for authorisation by xrootd-doors.
- NFS server export table (`/etc/exports`)
  Controls which and how clients can access NFS exports (both NFS 3 and NFS 4.1).
- Kerberos
  Used together with gPlazma in the case of NFS 4.1.

This course covers the access control systems based on gPlazma.

gPlazma

gPlazma (**G**rid-Aware **Pl**uggable **A**uthori**z**ation **Ma**nagement) is a service of dCache, providing functionalities for access control, which are used by doors in order to implement their respective access control system.

It should be noted, that not every type of door makes necessarily use of all the features provided by gPlazma.

It is used by NFS 4.1-, WebDAV-, gsiDCAP-, GridFTP-, xrootd- and SRM-doors.
To some extent it is further used by the `admin`- and `httpd`- services.

In order to serve different needs, gPlazma utilises plug-ins as back-end for its tasks and services.

This course gives only an overview of the plug-in system itself and some of the available plug-ins.

Requirements And Constraints

gPlazma has a number of requirements and constraints:

- Of course, the specific door must make use of gPlazma.
- gPlazma requires the CA- and VOMS-root-certificates, that it should trust, to be present (typically) in `/etc/grid-security/certificates/` and `/etc/grid-security/vomsdir` respectively.
- For some of the plug-ins, gPlazma requires X.509-host-certificates to be present (typically) in `/etc/grid-security/`.
- The configuration for gPlazma and its used plug-ins must be present on each host that runs a `gplazma`-service.
- Multiple DNs per client-certificate are currently generally not supported (but multiple FQANs are).
- `gplazma` must be restarted in order to notice changes to its configuration (not however to the certificates).
- In advanced setups, there might be more than one instance of `gplazma` per cluster. It is then usually desired to have gPlazma's configuration synchronised between all of its instances.

Enabling The gPlazma-Cell And General Configuration

In order to enable gPlazma for a cluster, the `gplazma`-service must be added to a domain in the "layout-configuration-file" on a node.

General configuration-parameters are:

- `gPlazmaNumberOfSimultaneousRequests`
  Specifies the number of requests gPlazma may process concurrently.
- `gplazma/cell.name`
  Specifies the name of the gPlazma-cell.
  Only important when running several instances of `gplazma` per cluster.
- `gplazma`
  Specifies the name of the gPlazma-cell that doors will use.
  Only important when running several instances of `gplazma` per cluster.
- `useGPlazmaAuthorizationModule` and `useGPlazmaAuthorizationCell`
  Can be used to have only "local" gPlazma-cells, run by the respective doors.
- `gplazma.configuration.file`
  The location of gPlazma's plug-in configuration file, per default `/etc/dcache/gplazma.conf`.

The gPlazma Plug-In System

gPlazma's plug-in system is similar to that of PAM:
There are different classes of plug-ins, with a number of such plug-ins of the same class being a phase.

The following classes/phases exist:

- `auth`
  Authenticating an accessing entity, for example via an X.509 certificate or a Kerberos ticket. This step leads to authentication information like a DN, FQANs or a Kerberos principal.
- `map`
  Mapping of authentication or mapping information (this is basically "recursive" mapping), for example a DN and FQANs to UNIX User IDs and Group IDs.
- `account`
  Determining the status of an account, for example whether it was banned.
- `session`
  Determining the session environment of the access, for example home- and root-directories.

The gPlazma Plug-In System

- `identity`
  Mapping of (UNIX) User IDs and Group IDs to user names and group names – and vice-versa.

The gPlazma Plug-In System

Each plug-in of a phase either succeeds or fails.
The control-type of the respective plug-in determines what happens then.

The following control-types exist:

- `optional`
  Continue with the “next” plug-in, regardless whether the “current” one succeeds or fails.
- `sufficient`
  If the “current” plug-in succeeds, finish the phase with success and do not process its “next” plug-ins.
- `requisite`
  If the “current” plug-in fails, finish the phase with failure and do not process its “next” plug-ins.
- `required`
  If the “current” plug-in fails, finish the phase with failure but process its “next” plug-ins nevertheless.

Currently Available gPlazma Plug-Ins

Currently the following plug-ins are available:

- for the `auth`-phase
  `x509`, `voms`, `xacml`, `jaas` and `kpwd`
- for the `map`-phase
  `vorolemap`, `authzdb`, `krb5`, `nsswitch`, `ldap`, `nis`, `gridmap`, `kpwd` and `mutator`
- for the `account`-phase
  `argus` and `kpwd`
- for the `session`-phase
  `authzdb`, `nsswitch`, `ldap`, `nis`, and `kpwd`
- for the `identity`-phase
  `nsswitch`, `ldap`, and `nis`

This course covers the plug-ins `x509`, `voms`, `vorolemap` and `authzdb`.

The "x509", "voms", "vorolemap" and "authzdb"-Plug-ins

The following is a typical combination of plug-ins that allows the usage of X.509 certificates (via the "x509" plug-in) with support for virtual organisations and their attributes, like `Role` and `Capability` (via the "voms" plug-in).

So called "grid-vorolemap-files" (via the "vorolemap" plug-in), which are similar to the "grid-mapfile-files", and "storage-authzdb-files" (via the "auhtzdb" plug-in) are used to map the certificate credentials to authentication information (especially UNIX User IDs and Group IDs).

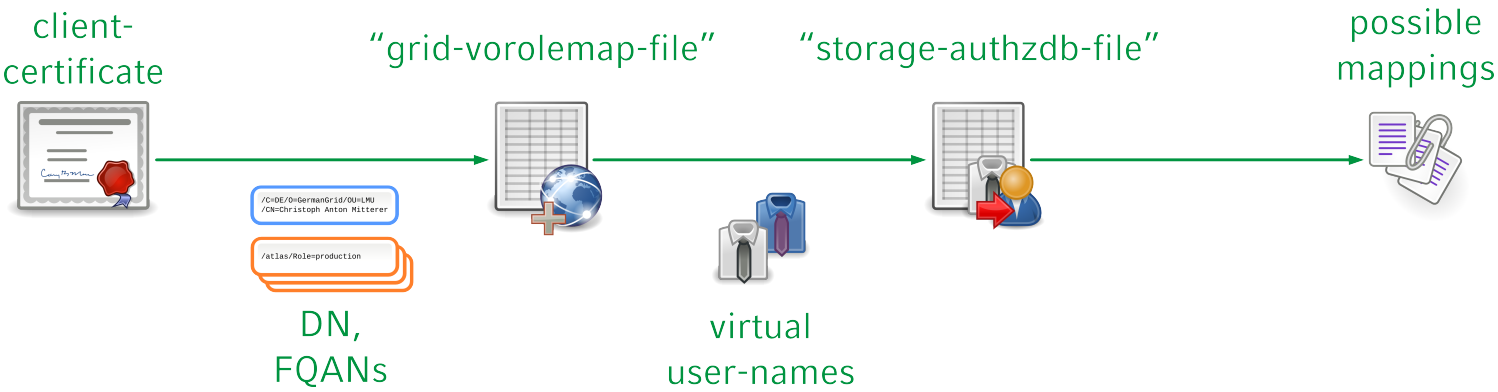
The plug-ins are configured via the following parameters:

- `gplazma.vomsdir.ca` and `gplazma.vomsdir.dir`
  Specify the pathnames to the CA- and VOMS-root-certificates that shall be trusted.
- `gplazma.vorolemap.file` and `gplazma.authzdb.file`
  Specify the pathnames of the "grid-vorolemap-file" and the "storage-authzdb-file" to be used by the plug-ins.
- `gplazma.authzdb.uid` and `gplazma.authzdb.gid`
  Advanced options which are not covered here.

The "x509", "voms", "vorolemap" and "authzdb"-Plug-ins

The following two-stage mapping-mechanism is applied:

1. Using the "grid-vorolemap-file", the client-certificate's DN and FQANs are mapped to some virtual user-names, which are not to be confused with actual UNIX user-names.
2. Using the "storage-authzdb-file", these virtual user-names are then mapped to the actual UNIX user-ID(s) and group-IDs (as they are also used by dCache's file hierarchy provider Chimera) as well as some other information.



The "x509", "voms", "vorolemap" and "authzdb"-Plug-ins

Example gPlazma plug-in configuration file:

```
auth    optional        x509
auth    optional        voms

map     optional        vorolemap
map     requisite       authzdb

session requisite       authzdb
```

"grid-vorolemap-File" Syntax And Semantics

Each line specifies a mapping from a client-certificate's DN and optionally FQAN to exactly one virtual user-name via the following syntax:
"*distinguished_name*" ["*fqan*"] *virtual_user-name*

distinguished_name and *fqan* must always be quoted using "".
Lines not starting with "" are currently ignored by dCache.
If the same DN occurs in multiple lines with the same FQAN then only the mapping from the last one is used. However, the same DN can be used multiple times with different FQANs and thus map to different virtual user-names.
If *fqan* is empty or "*fqan*" not specified at all, only client-certificates with an empty or no FQAN will match.

Examples:

- "/C=DE/O=GermanGrid/OU=LMU/CN=Christoph Anton Mitterer" "/atlas" atlas001
- "/C=DE/O=GermanGrid/OU=LMU/CN=Christoph Anton Mitterer" "/atlas/Role=production" prdatl01

"grid-vorolemap-File" Syntax And Semantics

distinguished_name can also be set to "*", which serves as a regular expression matching any character sequence ("wildcard character").
This is especially useful when mapping whole VOs.

Examples:

- `"*" "/atlas" atlas001`
- `"*" "/atlas/Role=production" prdatl01`

It is important to note, that when any mapping is found via an "explicit-DN-match", all mappings that would arise from a "wildcard-match" are ignored. This applies for "disabling entries", too.

"Disabling entries" (also called "revocation entries") can made by using "`-`" as virtual user-name.
It should be noted however, that other `map`-plug-ins may be still tried (and succeed).

"grid-vorolemap-File" Example Mappings

- "single wildcard-match"
  Certificate-DN: `/C=DE/O=GermanGrid/OU=LMU/CN=Christoph Anton Mitterer`
  Certificate-FQANs: `/atlas`
  "grid-vorolemap-file"-contents:
  ```
  "*" "/atlas" atlas001
  "*" "/atlas/de" atlas002
  ```
  Resulting mappings: `atlas001`
- "multiple wildcard-matches"
  Certificate-DN: `/C=DE/O=GermanGrid/OU=LMU/CN=Christoph Anton Mitterer`
  Certificate-FQANs: `/atlas`, `/atlas/de`, `/atlas/Role=production`
  "grid-vorolemap-file"-contents:
  ```
  "*" "/atlas" atlas001
  "*" "/atlas/de" atlas002
  "*" "/atlas/Role=production" prdatl01
  ```
  Resulting mappings: `atlas001`, `atlas002`, `prdatl01`

"grid-vorolemap-File" Example Mappings

- "overriding explicit-DN-match"
  Certificate-DN: `/C=DE/O=GermanGrid/OU=LMU/CN=Christoph Anton Mitterer`
  Certificate-FQANs: `/atlas`
  "grid-vorolemap-file"-contents:
  (the order of these lines does not matter)

```
"*" "/atlas" atlas001
"/C=DE/O=GermanGrid/OU=LMU/CN=Christoph Anton Mitterer" "/atlas" ops
```

  Resulting mappings: `ops`
- "disabling entry"
  Certificate-DN: `/C=DE/O=GermanGrid/OU=LMU/CN=Christoph Anton Mitterer`
  Certificate-FQANs: `/atlas`, `/atlas/de`, `/atlas/Role=production`
  "grid-vorolemap-file"-contents:
  (the order of these lines does not matter)

```
"*" "/atlas" atlas001
"/C=DE/O=GermanGrid/OU=LMU/CN=Christoph Anton Mitterer" "/atlas" -
```

  Resulting mappings: `-` (special "disabling" mapping)

"storage-authzdb-File" Syntax And Semantics

Each line starts with a keyword and is interpreted according to the currently set "storage-authzdb-file"-version or its default.
Keywords are followed by their respective arguments, which are all separated by white-space.
Lines not starting with a keyword are currently ignored.

The following keywords are recognised:

- `version`
  Sets the current "storage-authzdb-file"-version, that will be used when interpreting entries.
  It will be valid until the next occurrence of a `version`-keyword.

  The following versions are currently available:
  - `2.1`
    The default version.
  - `2.2`
    This version adds support for priorities of entries.

"storage-authzdb-File" Syntax And Semantics

- `authorize`
  Introduces a mapping with the static mapping-method, using the following syntax and semantics:
  `authorize` *`virtual_user-name access-mode`* {*`priority`*}[version 2.2] *`user-ID group-ID`*[`,`*`group-ID`*]* *`home-path root-path fs-root-path`*

  `access-mode` specifies the allowed access-mode for the respective virtual user-name and can be either set to `read-write` or to `read-only`.
  `priority` specifies the priority of the entry using non-negative integers, where a higher number means a higher priority. It cannot be set with the 2.1- but must be set with the 2.2-"storage-authzdb-file"-version.

  Multiple *`group-ID`*s are separated by "`,`".

"storage-authzdb-File" Syntax And Semantics

home-path is of legacy use and specified the home-directory for a virtual user-name. It is always interpreted relatively to *root-path*.

root-path is of legacy use and specified the root-directory for a virtual user-name. It is always interpreted as an absolute path.

fs-root-path is of legacy use and was needed for Kerberos.

These paths should not be used anymore, especially as they are not necessarily respected by all doors.
However, a value must be specified and it is suggested to set all three to "/".

"storage-authzdb-File" Syntax And Semantics

Example:

```
version 2.1

authorize atlas001 read-only 1000 100 / / /
authorize prdatl01 read-write 1001 101 / / /
```

Authentication- and Authorisation-Process

The following is a very brief description of the authentication- and authorisation-process:

1. A client makes an access request on some resource via a door using its protocol.
   Depending on the protocol, the access is either secured or not:
   - unsecured protocols (DCAP, HTTP)
   - secured protocols (NFS 4.1, WebDAV, gsiDCAP, GridFTP, xrootd, SRM)
     In this case, the client presents credentials, for example via a certificate with a DN and optionally one or more FQANs or via a Kerberos ticket.

   Although plain xrootd is unsecured, it is however possible to secure xrootd, too.
2. The access control system determines possible mappings (these include the actual UNIX user-ID(s) and group-IDs as well as some other information).
   - Depending on the door, gPlazma is used for this.
   - With unsecured protocols, the client receives usually a special limited mapping.
3. The access control system evaluates the found mappings against the policy data for the requested resource in order to determine the access decision.
4. The access decision is enforced and the access granted or denied.

Policies

Policies contain the rules that describe how resources might be accessed.

dCache can use two types of policies:

- Traditional POSIX file permission modes
- Access Control Lists
  dCache's "native" ACLs (not the ones from the "ALICE security model") are a subset of the NFS version 4 ACLs, providing nearly all of their features.

  They are evaluated in addition to the traditional POSIX file permission modes, which they generally outvote.

The data for both is stored within dCache's file hierarchy and thus the `chimera`-database.

This course does not cover ACLs.

Finis coronat opus.