Some gPlazma things

Paul Millar
Berlin, 2013.05.28
Quick intro/reminder* of gPlazma

* delete as appropriate
Need to identify users
Logging in: the four phases

- door
- gPlazma
- Authn
- Map
- Account
- Session
Logging in: first phase

Has the user proved who they are?
Similar to entering a country

- Is it a valid ID card?
- Does the person look like the photo?
- Pull out information

Nationality: German
Authenticate and extract principals

Name: Erika Mustermann
DoB: 1964-08-12
Place of Birth: Berlin
Authentication: door, both or gPlazma

- NFS door
- WebDAV door
- gPlazma

Usernames and passwords are required for both NFS and WebDAV doors. Authentication can be Kerberos or X.509 certificates.
Logging in: second phase

Who is this user, in dCache terms: uid, gids?
Mapping principals

DN:/C=UK/O=eScience/OU=Glasgow/L=Compserv/CN=paul millar

DN:/C=DE/O=GermanGrid/OU=DESY/CN=Paul Millar

Kerberos: paul@DESY.DE

uid: 15691

gid: 1000

FQAN: /atlas

Kerberos: atlas@DESY.DE
Logging in: third phase

Is this user allowed to use the service?
Logging in: final phase

Add extra information: root and home directory, ...
Each phase uses plugins

- Each plugin has a small, concrete job
  - Each gPlazma-1 file format is supported by a gPlazma-2 plugin
  - Plugins can support one or more phase

- Plugins succeed or fail
  - Not all failures are “bad”

- You get to configure:
  - which plugins run in each phase,
  - in which the order plugins run within a phase,
  - how gPlazma treats a plugin's success or failure.
Migrating from gPlazma-1 to gPlazma-2
You should act now

• In dCache v2.2
  • gPlazma-1 still exists, but only a gPlazma-2 plugin
  • Default gPlazma-2 configuration uses gPlazma-1 plugin
  • Same functionality now available via new plugins,
  • Can edit gPlazma-2 config file to use the new plugins.

• In dCache v2.6
  • gPlazma 1 has gone completely
  • Upgrading from 2.2 to 2.6 forces you to use new gPlazma-2 plugins,
  • Default configuration assumes a grid site (x509, voms, vorolemap, gridmap, authzdb)
The procedure for updating

- **Create** a new file: `gplazma-new.conf`
  
  details in a moment

- **Test** new configuration:
  
  if you don't have a test instance, start new domain running gPlazma (with distinct cell.name and using `gplazma-new.conf`) and a test door, configured to use this gPlazma.

- **Deploy** new configuration:
  
  - Copy `gplazma.conf` file as `gplazma-old.conf`
  - Register site “at risk” in GOC-DB, if you're paranoid
  - Copy `gplazma-new.conf` as `gplazma.conf`.

- **Note**: there's no need to restart dCache.
gPlazma-2 plugins for gPlazma1 “plugin”s

<table>
<thead>
<tr>
<th>Phase</th>
<th>gPlazma-1 “plugin”s</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>kpwd</td>
</tr>
<tr>
<td></td>
<td>grid-mapfile</td>
</tr>
<tr>
<td></td>
<td>Gplazmalite-vorole-mapping</td>
</tr>
<tr>
<td></td>
<td>xacml-vo-mapping</td>
</tr>
<tr>
<td>auth</td>
<td>opt: x509, opt: kpwd</td>
</tr>
<tr>
<td></td>
<td>opt: x509</td>
</tr>
<tr>
<td></td>
<td>opt: x509, opt: voms</td>
</tr>
<tr>
<td></td>
<td>opt: xacml</td>
</tr>
<tr>
<td>map</td>
<td>suf: kpwd</td>
</tr>
<tr>
<td></td>
<td>opt: gridmap, suf: authzdb</td>
</tr>
<tr>
<td></td>
<td>opt: vorolemap, suf: authzdb</td>
</tr>
<tr>
<td></td>
<td>suf: authzdb</td>
</tr>
<tr>
<td>account</td>
<td>req: kpwd</td>
</tr>
<tr>
<td></td>
<td>req: gridmap</td>
</tr>
<tr>
<td></td>
<td>req: vorolemap</td>
</tr>
<tr>
<td></td>
<td>req: authzdb</td>
</tr>
<tr>
<td>session</td>
<td>suf: kpwd</td>
</tr>
<tr>
<td></td>
<td>suf: authzdb</td>
</tr>
<tr>
<td></td>
<td>suf: authzdb</td>
</tr>
<tr>
<td></td>
<td>suf: authzdb</td>
</tr>
</tbody>
</table>

Key: opt = optional, suf = sufficient, req = requisite
Creating gplazma-new.conf file

• For each enabled gPlazma-1 “plugin” there is a set of gPlazma-2 plugins you need to configure

• Write down the gPlazma-2 plugins corresponding to each gPlazma-1 “plugin” in decreasing order of priority:
  
  For the auth phase:
  • Write down the auth phase gPlazma-2 plugins for the highest priority gPlazma-1 “plugin”
  • Write down the auth phase gPlazma-2 plugins for the next highest priority gPlazma-1 “plugin”
    ...
etc...
  and so on for map, account and session phases.

• If the same plugin appears multiple times in auth or session phase then you can keep the first plugin and remove the remainder.

• Do NOT remove duplicate plugins in map phase
Migration worked example
Worked example

This is an example

dcachesrm-gplazma.policy file.

# Switches
xacml-vo-mapping="OFF"
saml-vo-mapping="OFF"
kpwd="ON"
grid-mapfile="ON"
gplazmalite-vorole-mapping="ON"

# Priorities
xacml-vo-mapping-priority="5"
saml-vo-mapping-priority="1"
kpwd-priority="3"
grid-mapfile-priority="4"
Gplazmalite-vorole-mapping-priority="2"

# ... rest of configuration file would go here ...
Worked example

# Switches
xacml-vo-mapping="OFF"
saml-vo-mapping="OFF"
kpwd="ON"
grid-mapfile="ON"
gplazmalite-vorole-mapping="ON"

# Priorities
xacml-vo-mapping-priority="5"
saml-vo-mapping-priority="1"
kpwd-priority="3"
grid-mapfile-priority="4"
Gplazmalite-vorole-mapping-priority="2"

# ... rest of configuration file would go here ...
Worked example

Consider the remaining plugins in their execution order

```
# Switches
xacml-vo-mapping="OFF"
saml-vo-mapping="OFF"
kpwd="ON"
grid-mapfile="ON"
gplazmalite-vorole-mapping="ON"

# Priorities
xacml-vo-mapping-priority="5"
saml-vo-mapping-priority="1"
kpwd-priority="3"
grid-mapfile-priority="4"
gplazmalite-vorole-mapping-priority="2"
```
Worked example

First,
gplazma-vorole-map plugin
## gPlazma-2 plugins for gplazmalite-vorole...

<table>
<thead>
<tr>
<th>Phase</th>
<th>kpwd</th>
<th>grid-mapfile</th>
<th>gplazmalite-vorole-mapping</th>
<th>xacml-vo-mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>auth</td>
<td>opt: x509, opt: kpwd</td>
<td>opt: x509</td>
<td>opt: x509, opt: voms</td>
<td>opt: xacml</td>
</tr>
<tr>
<td>map</td>
<td>suf: kpwd</td>
<td>opt: gridmap, suf: authzdb</td>
<td>opt: vorolemap, suf: authzdb</td>
<td>suf: authzdb</td>
</tr>
<tr>
<td>account</td>
<td>req: kpwd</td>
<td>req: gridmap</td>
<td>req: vorolemap</td>
<td>req: authzdb</td>
</tr>
<tr>
<td>session</td>
<td>suf: kpwd</td>
<td>suf: authzdb</td>
<td>suf: authzdb</td>
<td>suf: authzdb</td>
</tr>
</tbody>
</table>

Key: opt = optional, suf = sufficient, req = requisite
Worked example

auth optional x509
auth optional voms

map optional vorole
map sufficient authdb

account requisite vorole

session sufficient authdb
Worked example

Next, the kpwd plugin

```
# Switches
xacml-vo-mapping="OFF"
saml-vo-mapping="OFF"
kpwd="ON"
grid-mapfile="ON"
gplazmalite-vorole-mapping="ON"

# Priorities
xacml-vo-mapping-priority="5"
saml-vo-mapping-priority="1"
kpwd-priority="3"
grid-mapfile-priority="4"
gplazmalite-vorole-mapping-priority="2"
```
# gPlazma-2 plugins for kpwd

<table>
<thead>
<tr>
<th>Phase</th>
<th>gPlazma-1 “plugin”s</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>kpwd</td>
</tr>
<tr>
<td></td>
<td>grid-mapfile</td>
</tr>
<tr>
<td></td>
<td>Gplazmalite-vorole-mapping</td>
</tr>
<tr>
<td></td>
<td>xacml-vo-mapping</td>
</tr>
<tr>
<td>auth</td>
<td>opt: x509,</td>
</tr>
<tr>
<td></td>
<td>opt: kpwd</td>
</tr>
<tr>
<td></td>
<td>opt: x509</td>
</tr>
<tr>
<td></td>
<td>opt: x509,</td>
</tr>
<tr>
<td></td>
<td>opt: voms</td>
</tr>
<tr>
<td></td>
<td>opt: xacml</td>
</tr>
<tr>
<td>map</td>
<td>suf: kpwd</td>
</tr>
<tr>
<td></td>
<td>opt: gridmap,</td>
</tr>
<tr>
<td></td>
<td>suf: gridmap</td>
</tr>
<tr>
<td></td>
<td>opt: vorolemap,</td>
</tr>
<tr>
<td></td>
<td>suf: vorolemap</td>
</tr>
<tr>
<td></td>
<td>suf: authzdb</td>
</tr>
<tr>
<td>account</td>
<td>req: kpwd</td>
</tr>
<tr>
<td></td>
<td>req: gridmap</td>
</tr>
<tr>
<td></td>
<td>req: vorolemap</td>
</tr>
<tr>
<td></td>
<td>req: authzdb</td>
</tr>
<tr>
<td>session</td>
<td>suf: kpwd</td>
</tr>
<tr>
<td></td>
<td>suf: authzdb</td>
</tr>
<tr>
<td></td>
<td>suf: authzdb</td>
</tr>
</tbody>
</table>

Key: opt = optional, suf = sufficient, req = requisite
Worked example

auth optional x509
auth optional voms
auth optional x509
auth optional kpwd

map optional vorole
map sufficient authdb
map sufficient kpwd

account requisite vorole
account requisite kpwd

session sufficient authdb
session sufficient kpwd
Worked example

```
auth optional x509
auth optional voms
auth optional x509
auth optional kpwd
auth optional x509

map optional vorole
map sufficient authdb
map sufficient kpwd
map optional gridmap
map sufficient authdb

account requisite vorole
account requisite kpwd
account requisite gridmap

session sufficient authdb
session sufficient kpwd
session sufficient authdb
```
Worked example

Authn

Map

Account

Session

auth optional x509
auth optional voms
auth optional x509
auth optional kpwd
auth optional x509

map optional vorole
map sufficient authdb
map sufficient kpwd
map optional gridmap
map sufficient authdb

account requisite vorole
account requisite kpwd
account requisite gridmap

session sufficient authdb
session sufficient kpwd
session sufficient authdb
Worked example

auth optional x509
auth optional voms
auth optional x509
auth optional kpwd
auth optional x509

map optional vorole
map sufficient authdb
map sufficient kpwd
map optional gridmap
map sufficient authdb

account requisite vorole
account requisite kpwd
account requisite gridmap

session sufficient authdb
session sufficient kpwd
session sufficient authdb
## Worked example

### Final result

- **auth**:
  - optional x509
  - optional voms
  - optional kpwd

- **map**:
  - optional vorole
  - sufficient authdb
  - sufficient kpwd
  - optional gridmap
  - sufficient admin

- **account**:
  - requisite vorole
  - requisite kpwd
  - requisite gridmap

- **session**:
  - sufficient authdb
  - sufficient kpwd
Customising plugin behaviour

- Some aspects of gPlazma-1 “plugins” may be customised in the `dcachesrm-gplazma.policy` file.
  (location of a file, GUMS server to contact, ...)
- With gPlazma-2, this is achieved with the normal dCache configuration
  - Properties for plugin foo start `gplazma.foo`
    e.g., `gplazma.vorolemap.file`, `gplazma.authzdb.uid`
  - Properties are documented in:
    `/usr/share/dcache/default/gplazma.properties`
  - Configure properties in `dcache.conf`, the layout file or in `gplazma.conf`
    if not sure, use `dcache.conf`
Example configuration

- In dcache.conf
  
gplazma.vorolemap.file=/etc/dcache/vorole.conf

- In layout file
  
  [testDomain/gPlazma]
  gplazma.vorolemap.file=/etc/dcache/vorole.conf

- In gplazma.conf
  
  map optional vorolemap gplazma.vorolemap.file=/etc/d[..]
Adding HTTP/WebDAV support
Just start WebDAV door

• Configure 'https-jglobus' for the webdavProtocol property
  (needed to support proxies)
• Start WebDAV door
• X.509 authenticated users should just work
Adding support for username+password

• Configure door to accept basic authn
  WebdavBasicAuthentication = enabled
  Make sure only https-jglobus is enabled.

• Add username+password support in gPlazma
  • Either use kpwd, jaas (with Kerberos) or jaas (external plugin activated)
  • See Ron's talk for details
Adding NFS mounted users
The simple approach: no Kerberos

- Start NFS v4.1 server
- Mount server on client machine
- uid/gid(s) on client machine are uid/gid(s) in dCache
  - If you have a mismatch then it’ll be painful
- “Trusted host” security.
Integrating with Kerberos

- Configure NFS door to use Kerberos
- Add the krb5 plugin to map phase
  maps 'paul@DESY.DE' to 'paul'
- Ensure user is mapped to a uid
  Either use static mapping (e.g., kpwd) or call out to local infrastructure (e.g., nis, ldap)
- On client machine, mount with Kerberos security and with rpc.idmapd, rpc.gssd running
Summary

• Sites should be upgrading to gPlazma-2 now(-ish)

• Upgrading is (relatively) easy, may be tested and executed without down-time

• gPlazma-2 allows NFS mounting using Kerberos security