# Setting up the CSP mode in a Gridengine production cluster

## Based on the experience at DESY

Andreas Haupt
DESY – DV –

Fall HEPiX 2012
Beijing, 2012-10-17

HELMHOLTZ GEMEINSCHAFT

DESY

# Outline

- Short overview of the Gridengine installations at DESY
- “Authentication” in the Gridengine default setup
- Why care?
- The “Certificate Security Protocol”
- Step by step guide to activate CSP in Gridengine

# Gridengine at DESY

- 4 independent batch systems
- Overview:

| Batch system | Implementation | Cores |
|---|---|---|
| Hamburg (BIRD) | SoGE 8.1 | 732 |
| NAF | OGS 6.2u5p2 | 2960 |
| Zeuthen (general use) | UGE 8.0.1p11 | 1584 |
| Zeuthen (parallel cluster PAX) | UGE 8.0.1p11 | 1024 |

# “Authentication” in the Gridengine default setup

- In short: there is none ...
- Client uses “geteuid()” system function call
  - This one is told to the Gridengine master server
  - This holds true for all currently available Gridengine implementations :-(
- … this is worse than e.g. NFSv3
- Other batch systems usually provide at least host-based authentication
  - e.g. munge

# Why care?

- Can easily be exploited
  - it allows arbitrary accounts to masquerade as other users at least
- As soon as users have access to an “administrative host”:
  - ability to change the Gridengine configuration
  - finally results in ability to become root on execution nodes
  - depending on configuration even further privilege escalation possible
- All in all it's a indefensible situation on clusters where you can't (or don't want to) trust all your users

# Possible workarounds

- Limit submit hosts to “gatekeeper nodes”
  - Users can't get in “direct” interaction with Gridengine
  - Probably only feasible for grid installations (CREAM)
- Limit administrative hosts to “secure” systems without user access
  - Fixes the root exploit (at least the direct one ...)
  - Doesn't fix the ability to masquerade accounts, though
- Provide client programs without the possibility to overwrite system functions
  - static binaries
  - doesn't protect from self-compiled & modified clients
- Provide modified clients capable of using privileged tcp ports only
  - ... and drop all traffic on master/execution hosts from unprivileged ports
  - very difficult to set up / fragile

# The “Certificate Security Protocol”

- (Useful) documentation at Oracle only
  - http://docs.oracle.com/cd/E24901_01/doc.62/e21973/chapter2.htm#BGBBFAHB
  - But still seems to be correct for all current implementations
- Based on X.509 certificates
- Users as well as Gridengine daemons do a mutual authentication
- Implemented as a “security mode”
  - Available security modes: afs, csp, (gss)
  - There can be just one active security mode ...
  - Workaround for AFS installations will be provided later

# The CSP directory structure

- $SGE_ROOT/$SGE_CELL/common/sgeCA
  - Common directory for all Gridengine hosts including subdirectories:
  - certs – contains daemon certificate
  - usercerts – contains user certificates
- /var/sgeCA/sge_qmaster/$SGE_CELL
  - If qmaster isn't running the default port, replace “sge_qmaster” with “port<tcp-port>” (e.g. “port1234”)
  - Subdirectory structure again
  - userkeys/<username> (only on master) – user keys / certs will be generated here
  - private – contains CA key and Gridengine daemon key
  - private/key.pem must be copied to all execution hosts

# Activating CSP

- Set up the CA:
  - Initializes basic directory structure and creates certificates for CA, daemons, admin users

```
[sgemaster] /root # cd $SGE_ROOT/util/sgeCA
[sgemaster] /usr/gridengine/util/sgeCA # ./sge_ca -init -days 3650
```

- You will be asked for:
  - Country code
  - State
  - Location
  - Organization
  - Organizational unit
  - CA email address

# Activating CSP

- Create credentials for users:

```
[sgemaster] /root # $SGE_ROOT/util/sgeCA/sge_ca -user <username>:<username>:<mail
address> -days 3650
[sgemaster] /root # ls -l /var/sgeCA/sge_qmaster/$SGE_CELL/userkeys/<username>/
total 32
-rw------- 1 user group 1464 Oct  8 10:14 cert.pem
-rw------- 1 user group  887 Oct  8 10:14 key.pem
-rw------- 1 user group 1024 Oct  8 10:14 rand.seed
-rw------- 1 user group  785 Oct  8 10:14 req.pem
```

- Credentials stored in /var/sgeCA/[sge_qmaster|port<port>]/$SGE_CELL/userkeys/<username>/
  - key as well as certificate

# CSP credentials on the client side

- $SGE_CERTFILE & $SGE_KEYFILE point to certificate & key
  - qsub, qstat, etc. use them directly
  - Gridengine profiles should set them for all users
- Directory structure beyond $HOME expected by qrsh:
  - directory .sge/[sge_qmaster|port<port>]/$SGE_CELL
  - cert/cert.pem: certificate
  - private/key.pem: key
  - private/rand.seed
  - execution node connects (authenticated) to client to open a shell on an execution node

# Bringing the certificates to the users

- So far credentials are only stored on the Gridengine master
  - The script provided with Gridengine ($SGE_ROOT/util/sgeCA/sge_ca -copy) relies on a shared filesystem containing the /var/sgeCA directory ...
- Set up a SPNEGO-authenticated (read: Kerberos5) cgi script
  - https web server with enabled mod_krb5
  - web server needs to have read access to the credentials directory
  - $REMOTE_USER contains the authenticated user
  - Returns a tarball containing the certificate / key
  - curl is currently the only command-line client capable of SPNEGO (at least to my knowledge ...)

# Bringing the certificates to the users (cont.)

- client script manages to setup the directory structure in the user's home directory
  - creates $HOME/.sge and makes it private for the user (AFS acl, directory permissions)
  - stores key & cert in $SGE_KEYFILE, $SGE_CERTFILE
  - creates directory structure to make qrsh work
- Provide wrapper scripts for the standard commands in a different path
  - Change Gridengine profiles so that they are used instead of the binaries
  - Wrapper scripts check for existence of credentials – if they are not in place, try to install them automagically
  - ... then start the real binary

# Workaround for existing AFS setups

- Security mode “afs” provides additional commands to support AFS
  - get_token_cmd: extract the AFS token on the client
  - set_token_cmd: started on execution node to store e.g. the AFS token
  - pag_cmd: sets up the AFS PAG (Process Authentication Group) for the job
- As mentioned, they cannot be used in CSP mode any more
- Workaround: replace standard “shepherd” process by a wrapper:

```
[client] ~ % qconf -sconf | grep shepherd
shepherd_cmd      /usr/gridengine/util/myshepherd
```

- myshepherd sets up a PAG and starts set_token_cmd as well as the original sge_shepherd process inside it
  - set_token_cmd is called regularly (e.g once a day) to renew credentials

# Finally: the real activation

- Check that all execution nodes have the daemon key installed
  - /var/sgeCA/sge_qmaster/$SGE_CELL/private/key.pem
- Check if all currently active user have their credentials stored correctly
  - ... and that the Gridengine profiles really establish $SGE_KEYFILE, $SGE_CERTFILE
- Modify $SGE_ROOT/$SGE_CELL/common/bootstrap
  - Set “security_mode csp”
- Restart all daemons
  - Preferably at the same time
  - Can be even done with a running cluster

# Pitfalls

- I just know of one currently ...
- When renewing certificates a CRL file is silently created / updated
  - $SGE_ROOT/$SGE_CELL/common/sgeCA/ca-crl.pem
  - Lifetime of 4 weeks only
  - If not removed / updated in time, all commands fail with a “certificate expired” error although all client and daemon certificates are still valid

# That's it folks!

Thank you for your attention