



Installing and configuring dCache

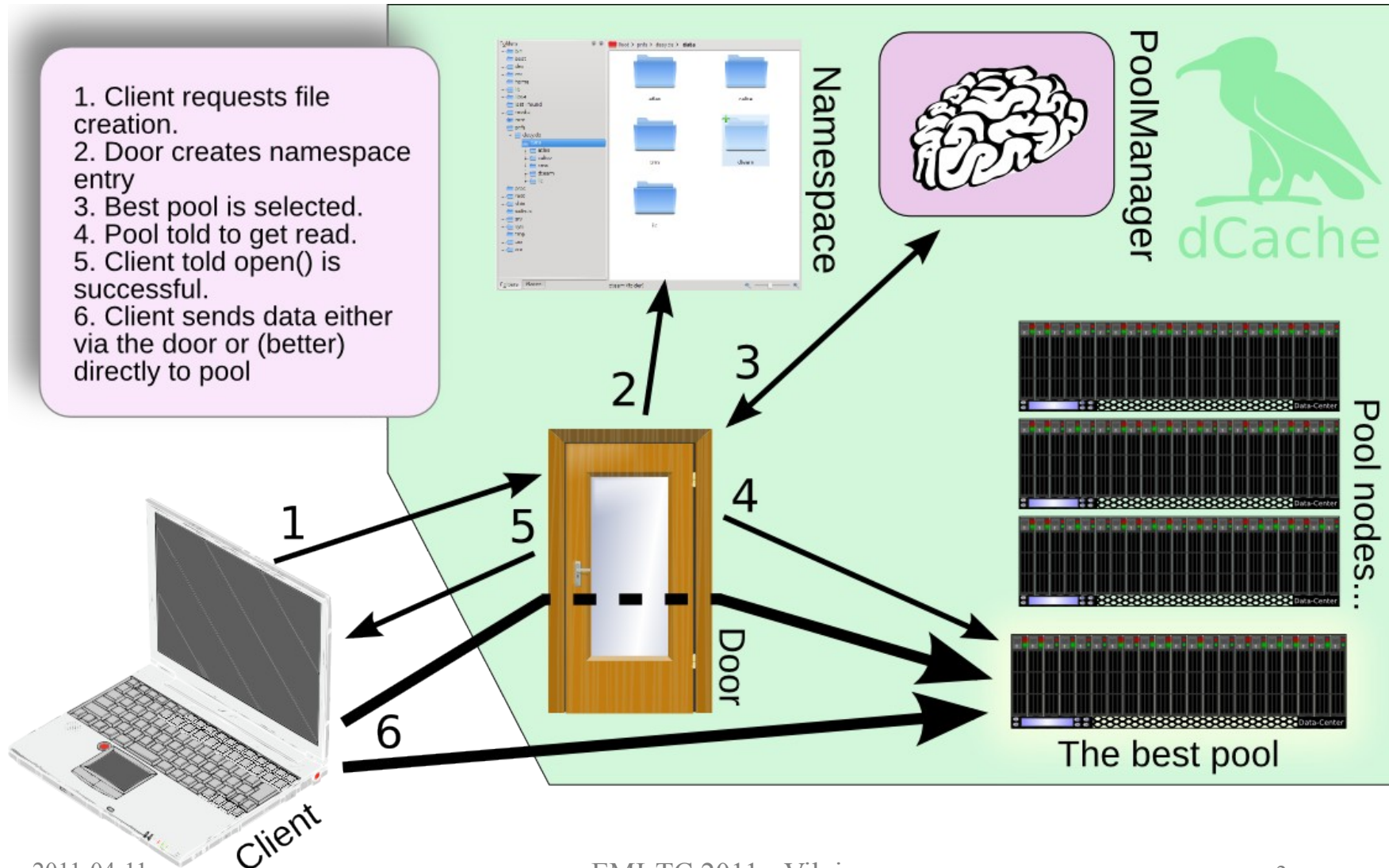
Paul Millar

About this workshop

- dCache is very flexible:
 - Configuring dCache is a *huge* topic
 - Can only give a sample in 30 minutes
- Focus on migrating a dCache v1.9.5 to v1.9.12
 - Expose you to the new configuration
 - Provide sites with worked examples in upgrading their site.
- If we have time, we can play with the configuration

dCache: writing a file

1. Client requests file creation.
2. Door creates namespace entry
3. Best pool is selected.
4. Pool told to get ready.
5. Client told open() is successful.
6. Client sends data either via the door or (better) directly to pool



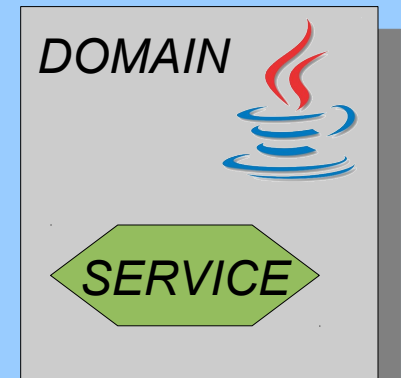
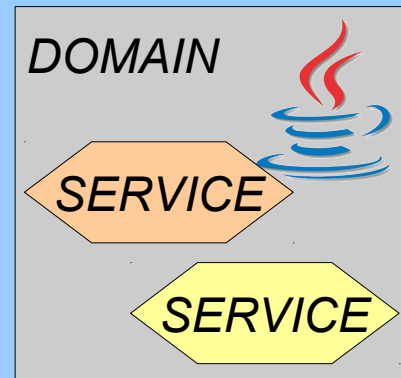
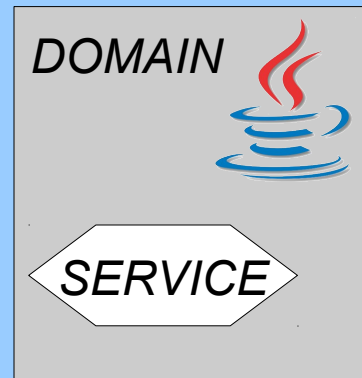
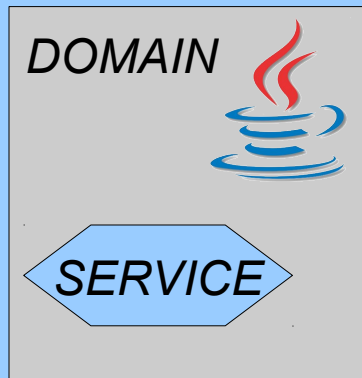
dCache v1.9.5



dCache instance



Processes running on this node:

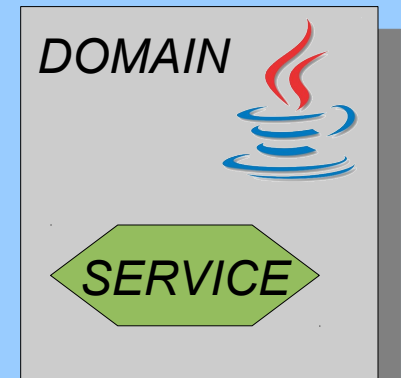
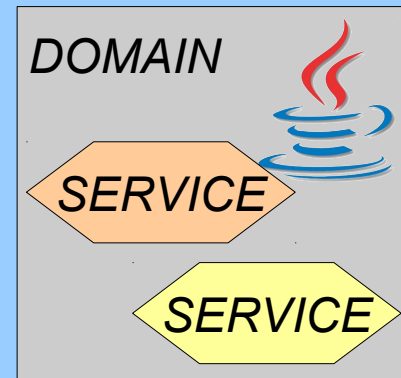
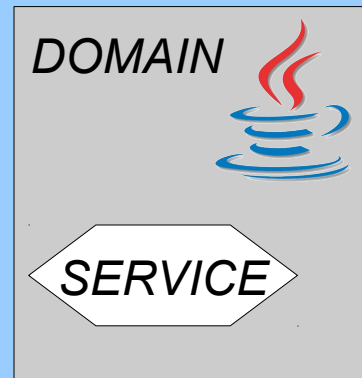
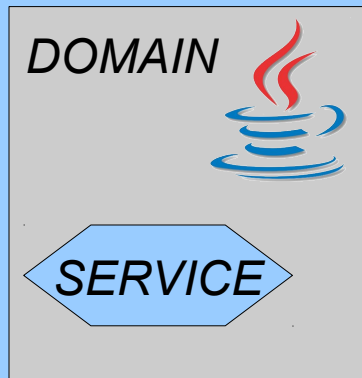


dCache v1.9.12 (after migration)



dCache instance

Processes running on this node:



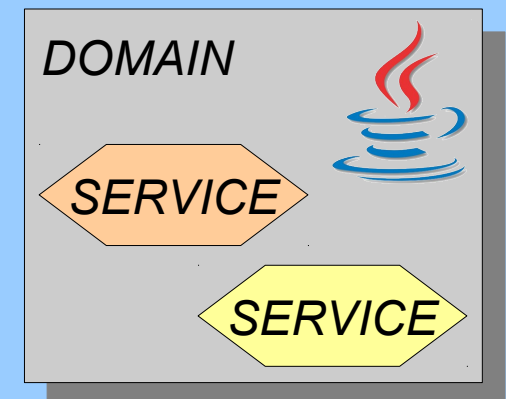
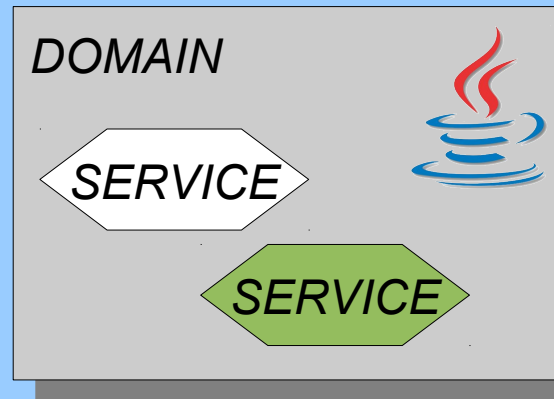
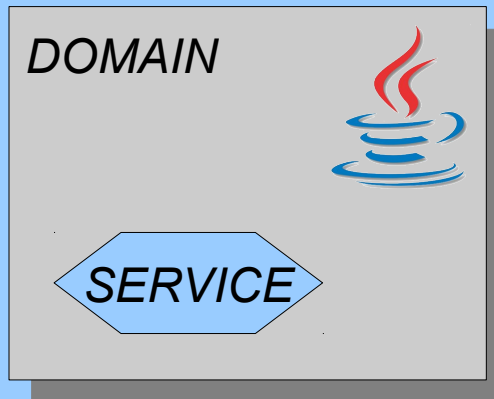
dCache v1.9.12 (consolidation)



dCache instance



Processes running on this node:



Advantages of new config. system

- An authoritative source of **default values**
- More **flexible**
- Configuration is shorter
- All nodes can share the **same set of configuration files**
- Managed life-cycle of a properties

Default values

- No template files!
- No commented-out “default” values
- Files located in `/opt/d-cache/share/defaults`
- These files provide:
 - Description for each property.
 - Authoritative statement on default values.
 - Annotations describing how property should be used.

Do not edit files in **defaults** directory. Your changes *will be lost* when you upgrade!

Fragment from a defaults file

```

keyStore=${dcache.paths.etc}/hostcert.p12

# ---- Password for SSL server certificate
#
# This parameter specifies the password with which the PKCS12 encoded
# server certificate is encrypted.
#
keyStorePassword=dcache

# ---- Trusted SSL CA certificates
#
# This parameter specifies the path to a Java Keystore containing
# the the trusted CA certificates for SSL. The CA certificates
# in /etc/grid-security/certificates/ must be converted into a
# Java Keystore file before being used. The default path is
# 'bin/dcache import cacert'.
# This is used in Webadmin.
#
# Notice that for GSI the certificates in
# /etc/grid-security/certificates/ are used directly.
#
trustStore=${dcache.paths.etc}/certificates.jks

# ---- Password for trustStore
#
# This parameter specifies the password with which the PKCS12 encoded
# containing the trusted CA certificates is encrypted.
#
trustStorePassword=dcache

```

Comments describing this property

Property `keyStorePassword` has a default value `dcache`. A site may alter this value with a similar line in `dcache.conf`

This default value contains a reference to property `dcache.paths.etc`. Changing `dcache.paths.etc` will automatically adjust `trustStore`.

/opt/d-cache/**etc/dcachelconf**

- Most properties are configured in this file
 - Simple list of property assignments
- It replaces dCacheSetup
- We recommend that this file is identical on all nodes in a dCache instance.
- There's no template file:
 - an empty dcachelconf is valid
- Most of the old parameters work as before
- The structure is *almost* the same as dCacheSetup

Layout file

- Each node uses a **single** layout file
- The file is located in `/opt/d-cache/etc/layouts`
 - If `dcache.conf` has `dcache.layout=foo` then:
`/opt/d-cache/etc/layouts/foo.conf`
 - Recommend using hostname for layout:
`dcache.layout=${host.name}`
- A layout file contains:
 - Which **domains** should be started on this node
 - Which **services** should run in these domains
 - (optionally) updated defaults for this node
 - (optionally) domain-specific configuration
 - (optionally) service-specific configuration

Example layout file

[doors]

Define the domain
doors

Define the domain
dCacheDomain

[dCacheDomain]

Defining a domain

Each domain declaration has form:

[<domain>]

The domain is called <domain>, which must be a unique name within the dCache instance. The name can be almost anything, but use only alphanumerical characters to be safe.

Example layout file

```
[doors]
```

```
[doors/dcap]
```

```
[doors/gsidcap]
```

```
[doors/gridftp]
```

```
[doors/webdav]
```

```
[doors/xrootd]
```

```
[dCacheDomain]
```

There are five services that should run inside domain doors: **dcap**, **gsidcap**, **gridftp**, **webdav** and **xrootd**.

Defining a service

Each service declaration has form:

```
[<domain> / <service>]
```

This says that **a** service of type **<service>** is started in domain **<domain>**. This may be repeated to start multiple instances of the same service.

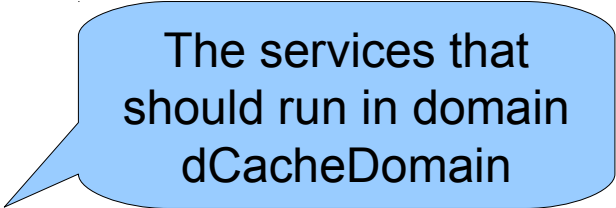
Example layout file

```
[doors]  
[doors/dcap]
```

```
[doors/gsidcap]  
[doors/gridftp]  
[doors/webdav]  
[doors/xrootd]
```

```
[dCacheDomain]
```

```
[dCacheDomain/poolmanager]  
[dCacheDomain/pinmanager]  
[dCacheDomain/cleaner]  
[dCacheDomain/pnfsmanager]  
[dCacheDomain/gplazma]
```



The services that
should run in domain
dCacheDomain

Example layout file

```
dcache.java.memory.heap = 512M
```

New default property values for this host

```
[doors]  
[doors/dcap]
```

```
[doors/gsidcap]  
[doors/gridftp]  
[doors/webdav]  
[doors/xrootd]
```

```
[dCacheDomain]
```

```
[dCacheDomain/poolmanager]  
[dCacheDomain/pinmanager]  
[dCacheDomain/cleaner]  
[dCacheDomain/pnfsmanager]  
[dCacheDomain/gplazma]
```

Updated default values for a node

Declarations at the beginning of are like a mini `dcache.conf` file. They declarations adjust the default values or those configured in `dcache.conf`. The new values will be used in all domains and all services running inside those domains.

Example layout file

```
dcache.java.memory.heap = 512M
```

```
[doors]  
[doors/dcap]
```

```
[doors/gsidcap]  
[doors/gridftp]  
[doors/webdav]  
[doors/xrootd]
```

```
[dCacheDomain]
```

```
  dcache.java.memory.heap = 2048M
```

```
[dCacheDomain/poolmanager]  
[dCacheDomain/pinmanager]  
[dCacheDomain/cleaner]  
[dCacheDomain/pnfsmanager]  
[dCacheDomain/gplazma]
```

Updated default values for a domain

Declarations immediately after a domain declaration affect only that domain and all services inside that domain. Other domains (and services running in those domains) are unaffected by the new configuration.

Updated property value that apply only to this domain

Example layout file

```
dcache.java.memory.heap = 512M
```

```
[doors]  
[doors/dcap]  
  port = 22126  
[doors/gsidcap]  
[doors/gridftp]  
[doors/webdav]  
[doors/xrootd]
```

Updated property value for this service only.

```
[dCacheDomain]  
  dcache.java.memory.he  
[dCacheDomain/poolmanag  
[dCacheDomain/pinmanage  
[dCacheDomain/cleaner]  
[dCacheDomain/pnfsmanag  
[dCacheDomain/gplazma]
```

Updated default values for a service

Declarations immediately after a service declaration affect only that service. The domain is unaffected by this configuration, as are all other domains and services running in those domains.

Pitfalls to watch out for

Duplicate declarations:

```
property.name = value 1  
property.name = value 2
```

```
# property.name = value 1  
property.name = value 2
```

Reference without braces:

```
cell.name = dcap-$host.name
```

```
cell.name = dcap-{host.name}
```

Recursive references:

```
cell.name = {cell.name}-1
```

In defaults file:

```
cell.name = dcap-{host.name}
```

In layouts file:

```
cell.name = dcap-{host.name}-1
```

Pitfalls to watch out for

Missing domain:

```
[domain/service]
```

```
[domain]  
[domain/service]
```

Service before domain:

```
[domain/service]  
[domain]
```

```
[domain]  
[domain/service]
```

Quotes are always significant:

```
cell.name = "foo bar"
```

```
cell.name = foo bar
```

Annotations

- A property's default value declaration also says how it may be used:
`srmPort = 8443`
(obsolete)`waitForRepositoryReady =`
(deprecated, not-for-services)`logArea =`

Annotation	Description	Effect when configured
deprecated	A property that will be retired in a future major version of dCache	Warning message is printed in log file.
obsolete	A property that is no longer supported, but isn't critical to dCache	Warning message is printed in log file.
forbidden	A property that is no longer support and was critical to dCache	Error message is printed and dCache refuses to start.
not-for-services	A property that has no effect if configured for a service.	Warning message is printed in log file if used in a service

check-config

- New command added in 1.9.12:
`/opt/d-cache/bin/dcache check-config`
`service dcache check-config`
- Checks:
 - Syntax of configuration files.
 - usage against annotations.
- Types of message:
 - Warning: a problem but dCache will start
 - Error: a problem dCache won't start.

A word of thanks to...

- Oleg Tsigenov
co-author of this tutorial
- Emidio Giorgio
coordinating facilities
- Salvatore (“Salvo”) Monforte
Heroic activity behind the scenes in
getting Xen to work

About the hands-on session:

- Using Xen virtual machines
 - generously provided by **National Institute of Nuclear Physics – Catania**
- 13 machines available:
 - vm[01..20].ct.infn.it but **NOT 03, 04, 07 and 16--20**
- SSH is available from port 2222
- The workshop instructions are here:
<http://bit.ly/eXNO9r>
<http://trac.dcache.org/projects/dcache/wiki/workshops/2011/vilnius>



Thank you!

EMI is partially funded by the European Commission under Grant Agreement RI-261611