CERN IT Department

Changes to LFC & DPM for IPv6

David Smith on behalf of IT-DM/SMD

19 February 08

Joint EGEE/EUChinaGRID/ETICS meeting on IPv6 and gLite

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Outline

- Introduce LFC & DPM
- Explain changes made to LFC & DPM in order to allow for the use of IPv6
  - i.e. the LCG-DM CVS component
  - Plus couple of other modules, for the gridftp2 DPM plugin and cgsi gSOAP plugin
- Components are written in C
  - This presentation may be framed in the language and concepts of C
- IPv6 use is for the SL4 release:
  - globus 4 is used for the SL4 builds. Globus 4 includes IPv6 support, globus 2.4.3 does not.

What are the LFC and DPM?

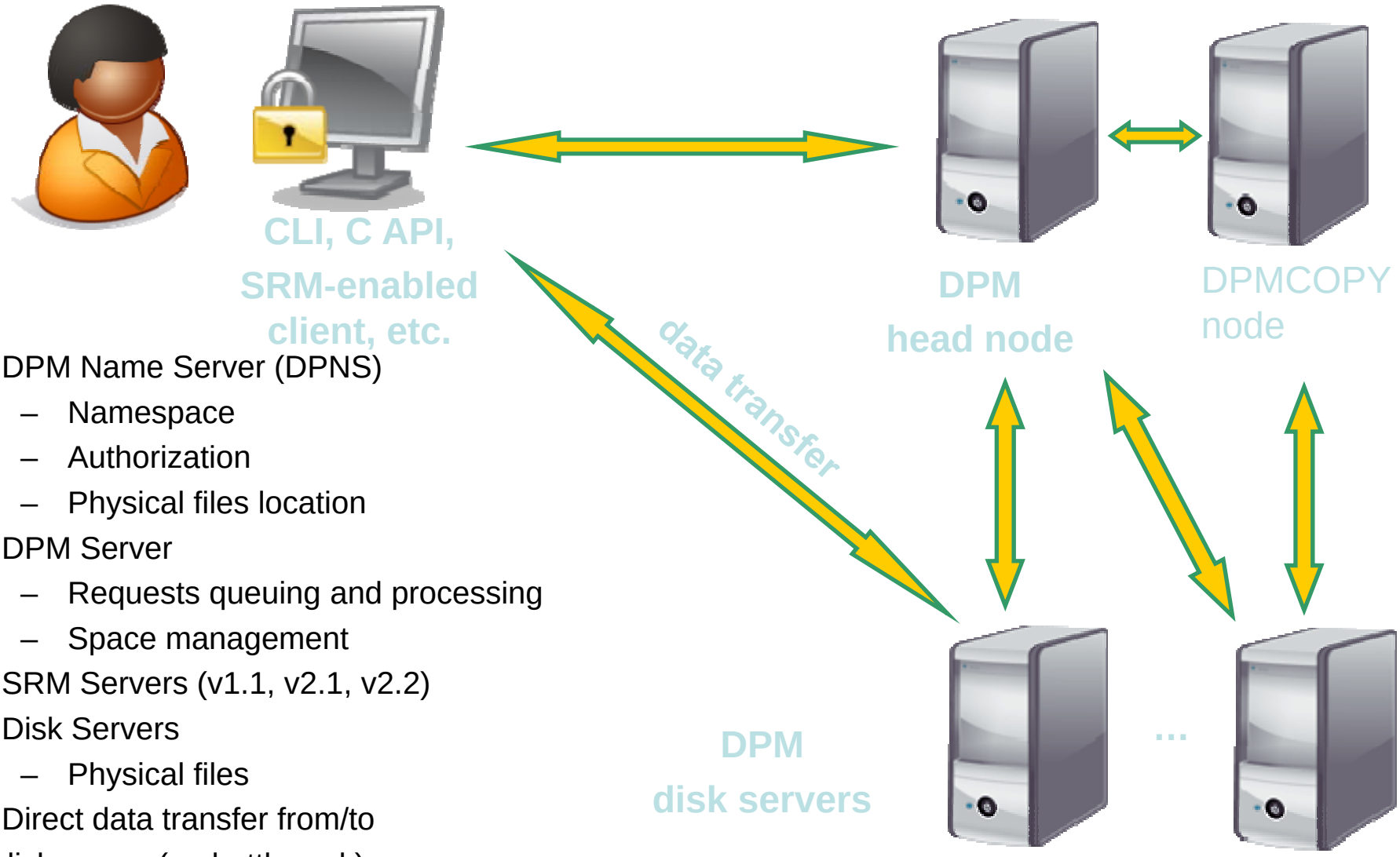
CERN IT Department

- LFC is a file catalogue
  - Generally used to lookup logical names to find one or more *replica* names.
  - Consists of a daemon, a database backend (mysql or oracle) and optionally a DLI - a service to answer requests via SOAP
- DPM is the Disc Pool Manager
  - Manage files in a number of pools (file systems on a number of disk servers)
  - One component of the DPM is the DPM Name server (DPNS) which is the same code base as the LFC
  - Other components include DPM, gridftp server, rfiod, srm versions 1, 2.1 & 2.2

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

CERN

DPM architecture



- DPM Name Server (DPNS)
  - Namespace
  - Authorization
  - Physical files location
- DPM Server
  - Requests queuing and processing
  - Space management
- SRM Servers (v1.1, v2.1, v2.2)
- Disk Servers
  - Physical files
- Direct data transfer from/to disk server (no bottleneck)
  - Access via e.g. gridftp & rfio

Software

- LFC
  - The daemons (LFC and DLI)
  - Client library (suitable for linking to C or C++ programs)
  - Command line tools - mostly providing access to selected API calls from the client library
  - Perl and python interfaces via swig wrapping of some of the client library functions
- DPM
  - The daemons (DPM, DPNS, rfiod, globus gridftp2 plugin, srmv1, srmv2, srmv2.2)
  - Client library, including DPM, DPNS and RFIO APIs
  - Command line tools for DPM, DPNS and RFIO
  - Perl and python interfaces via swig
  - Information provider information script
  - Some other data access protocols for optional deployment: xrootd and https
- Build regularly on linux but it is periodically also built on Mac OS X and Solaris

The IPv6 work

CERN IT Department

- Goal
  - Have the LFC or DPM services able to run on a suitably configured system using IPv6
  - Allow external and internal communication between components to use IPv6
  - mysql and Oracle don't support IPv6 yet
  - Don't want to degrade things for IPv4 systems
- Components to review:
  - Name server daemon & client (the LFC and DPNS)
  - DPM server daemon & client
  - SRM servers (3) & DLI server
  - RFIO server, client and command line utilities
  - Common routines (shared with Castor), 'Csec' security lib, and various plugins

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

CERN

The approach

- Make minimal changes, but
  - Try to preserve the functionality of all the functions
  - Avoid changes that are linux specific or exclusive
  - protocol family independence is desirable
  - Have IPv6 available by default if supported by the system. e.g. don't require special build with pre-processor macro.

- In general change:
  - gethostbyname() becomes getaddrinfo()
  - gethostbyaddr() becomes getnameinfo()
  - sockaddr_in structures are replaced by sockaddr_storage structures (for instance when allocating to use as arguments to functions such as getpeername(),getsockname())
  - Situations where one explicitly sets elements of sockaddr_in are replaced via calls to getaddrinfo() and a loop over a linked list

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

The approach II

CERN IT Department

- In the specific case of LCG-DM
  - Already have a common library with abstractions in packages such as Cmutex, Cthread, Cglobals, so:
  - Added new functions Cgetaddrinfo(), Cgetnameinfo(), Cgai_strerror() with the aim of working around any architecture specific quirks at the common level and allowing for consistent behaviour when used together with Cthreads
  - Also made a new utility function for the translation of addresses to hostnames
  - Decided to use separated stack approach of using IPV6_V6ONLY for the IPv6 listen sockets in those components which accept network connections

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

LFC and the DPNS

CERN IT Department

- LFC and the DPM Name Server share the same code base *ns* - built with some different options and defaults
- Cns_main.c
  - Add command line options to the daemon to allow selection of IPv4 only or IPv6 only service - default to both IPv4 & IPv6 if available
  - Hold a number of socket descriptors in an array: use those to listen for incoming connections on various socket domains (e.g. PF_INET and/or PF_INET6)
  - select() across the listen sockets and accept() when there are any connections
  - Replace Cgethostbyaddr() for finding peer's host name.

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

LFC / DPNS II

CERN IT Department

- Cns_main.c examples - get listen addresses:

```
 memset (&hints, 0, sizeof(struct addrinfo));
        if (listen_ipv4only)
                hints.ai family = PF INET;
        else if (listen_ipv6only)
                hints.ai_family = PF_INET6;
        else
                hints.ai_family = PF_UNSPEC;
        hints.ai socktype = SOCK STREAM;
        hints.ai_flags = AI_PASSIVE;
        if ((p = getenv (CNS_PORT_ENV)) || (p = getconfent (CNS_SCE, "PORT", 0))) {
                strncpy (strport, p, sizeof(strport));
                strport[sizeof(strport)-1] = '\0';
        } else if (sp = getservbyname (CNS_SVC, "tcp")) {
                snprintf (strport, sizeof(strport), "%u", ntohs (sp->s_port));
 } else {
                snprintf (strport, sizeof(strport), "%u", CNS_PORT);
        }

        if (gaierrno=Cgetaddrinfo (NULL, strport, &hints, &aitop)) {
…
```

- Produces linked list starting at aitop…

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

CERN

LFC / DPNS III

CERN IT Department

- Cns_main.c examples - listen on service sockets:

```
        num_listen_socks = 0;
        for (ai = aitop; ai; ai = ai->ai_next) {
                int fo = 0;
…
                if ((s = socket (ai->ai_family, ai->ai_socktype, ai->ai_protocol))<0)
…
                if (setsockopt (s, SOL_SOCKET, SO_REUSEADDR, (char *)&on, sizeof(on))) {
…
                if (ai->ai_family == PF_INET6) {
…
                        if (setsockopt (s, IPPROTO_IPV6, IPV6_V6ONLY,
                                        (char *)&on, sizeof(on))) {
…                        }
…
                }
                if (bind (s, ai->ai_addr, ai->ai_addrlen) < 0) {
…
                 listen_socks[num_listen_socks] = s;
                 ++num_listen_socks;
                 listen (s, 5);
         }
         freeaddrinfo (aitop);
…
```

- Then wait for connections by select()ing for read.

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

CERN

LFC / DPNS IV - making connections

CERN IT Department

- send2nsd.c contains the client function for connecting and communicating with the name server
  - Replace gethostbyname() with Cgetaddrinfo(): results in a linked list containing socket address structures for the desired peer, port & socket type - may return IPv6 or IPv4 addresses depending on the local system configuration and the DNS content.
  - Loop over resulting addresses attempting a synchronous connect() to each in turn in the order getaddrinfo() gave.
  - Have to rework existing wait/retry logic to retry the connect() sequence a number of times if all the address fail
- Other files involved in LFC/DPNS changes: Cns_chkperm.c, Cns_procreq.c, nsdaemon.man, Cns_server.h, Cns_api.h

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

DPM

- dpm_main.c & send2dpm.c had similar changes as the DPNS. Connection handling in send2dpm.c is actually simpler as it does not have the retry code.

- Other issues:
  - Some changes in a couple of routines that deal with machine names of pool servers. Check for ':', which would occur if the user tries to enter an IPv6 numeric address. In the case of the DPM these were simply disallowed.
  - IPv6 support has to be explicitly enabled when using the gridftp client - used by the DPM as a part of the replication tool.
- Other files involved: dpm.man, dpm_procreq.c, dpm_util.c, dpm_copyfile.c

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Globus gridftp client usage

CERN IT Department

- Globus describe IPv6 support as experimental. gridftp client can use EPRT, EPSV (RFC2428) which are available in the gridftp2 server
  - Gridftp client only used in the DPM by the replication tool to allow for the creation of file replicas within the storage element
  - Need to set ‘allow_ipv6’ operationattr in source and/or destination attribute-set for the gridftp client calls.
  - Appears not completely straight forward to setup for 3rd party transfers. For example if the source machine is IPv4 only the destination ipv6 attribute must not be set even if the destination is reachable on both IPv4 & IPv6: otherwise the 4 most sig. bytes of the destination IPv6 address are passed to the source as the target IPv4 address.
  - Approach adopted - for third party only set the ipv6 flag in both source and destination attribute-sets if both source and destination have AAAA records. For non 3rd party transfers set as expected.

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

CERN

Csec - background

CERN IT Department

- Csec is an API & plugin(s) which handle a network dialogue
  - To agree on an authentication scheme and perform the authentication exchange via a dynamically loaded library dedicated to a particular scheme.
  - GSI (globus' x509 and proxy handling infrastructure) together with VOMS is the usual method used
  - Internally a light-weight protocol called ID is used between DPM components
  - Kerberos 4 may optionally be compiled
  - Kerberos 5 may optionally be compiled (it uses the same gss-api interface as much of GSI does)

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Csec - changes

CERN IT Department

- Csec IPv6 changes:
  - Change name lookups and sockaddr_in storage types
  - Changed the server to not offer kerberos 4 authentication if the peer connects via IPv6, because:
  - Kerberos 4 includes a fixed 4-octet space in its protocol for addresses. The kerberos 4 API even includes sockaddr_in* in some prototypes and will attempt to verify the host address using that data. It may be technically possible to disable host checks in the KDC and application servers and have some level of functionality.

- Files involved: Csec_plugin_KRB4.c, Csec_protocol_policy.c, Csec_api.c (and various)

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

gSOAP based services

CERN IT Department

- The gSOAP based services are the SRM (called v1, v2 and v2.2) plus the LFC's data location index (DLI) service.
  - gSOAP handles the marshalling of various data types across the network and connection setup on both client and server.
  - IPv6 compliance may be selected with a compile time pre-processor macro 'WITH_IPV6' when compiling stdsoap2.c
  - Effectively switches to using getaddrinfo() etc.
  - The soap structure element 'ip' is no longer filled. Methods that need the peer's address must use the 'peer' element, which becomes a sockaddr_storage type.
  - Uses the dual stack approach - i.e. one IPv6 socket that may accept IPv6 or IPv4 connections

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

CERN

gSOAP GSI plugin

CERN IT Department

- A gSOAP plug-in called cgsi-gsoap is used to implement GSI authentication and encryption on soap connection. Cgsi-gsoap is a separate module from the LFC or DPM - in both CVS and in terms of release.
  - Had previously changed the plug-in to use sockaddr_storage type and getaddrinfo(), getnameinfo() for lookups.
  - No further changes made in recent work.

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

CERN

RFIO

- RFIO is used internally within the DPM and is also used by users
  - Is a data access protocol and server - also uses Csec for authentication on the control channel
  - IPv6 related changes needed in several places:

- Connection attempts in checkkey.c, connect.c, stream.c
  - Similar change as send2dpm: potentially attempt to connect to a number of addresses (IPv4 or IPv6)
  - stream.c attempts to open the data connection when using the v3 (rfio stream) protocol. The data connection is attempted only over the same address family as the control connection.

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

RFIO II

CERN IT Department

- Connections accepted or sockets bound/setup for listening in rfio_calls.c, rfio_serv.c
  - The server changes are similar to the dpm or dpns servers
  - The v3 data port setup in rfio_calls.c is performed by a series of bind() attempts finally followed by listen() - to allow a port range to be specified

```
        hints.ai_family = af;
        hints.ai_socktype = SOCK_STREAM;
        hints.ai_flags = AI_PASSIVE;
…
 while (1) {
            snprintf (strport, sizeof(strport), "%u", port);
            gaierrno = Cgetaddrinfo(NULL, strport, &hints, &aitop);
            if (gaierrno == 0) {
              ai = aitop;
              if(bind(data_s, ai->ai_addr, ai->ai_addrlen) == 0) {
                freeaddrinfo(aitop);
                return (port);
              }
              freeaddrinfo(aitop);
            }
            port++;
            if (port > port_max) port = port_min;
            if (port == start_port) break;
          }
```

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

RFIO III

- Various name lookups for logging or other purposes
  - Replace sockaddr_in with sockaddr_storage
  - Replace gethostbyaddr() with custom utility function (essentially getnameinfo())
- Allow for IPv6 style addresses in rfio_parse()
  - Usually in DPM we don't expect numeric IP addresses, but at the RFIO level wanted to be able to allow them
  - The ':' is usually used to delineate port number. For IPv6 require square brackets if a port number is to be specified
  - e.g. [1:2:3:4:aaaa::fff]:5001
- Files: open.c, open64.c, opendir.c, rfio_call64.c, rfio_fcalls.c, stream64.c, parse.c

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

RFIO IV

- (largely legacy) accounting log included two 4-octet fields for IP addresses
  - Created new package identifier and change ACCTRFIO64 to ACCTRFIO64IPV46 to indicate a new structure with more space for two IPv6 addresses
- Some structure types such as RFILE contain arrays which are supposed to be large enough for numeric IP addresses
  - Raise RESHOSTNAMELEN
- Files: rfioacct.c, rfio_constants.h, sacct.h

Common

CERN IT Department

- There are a set of common *Castor* utility functions. Some needed change or the addition of new functions:
  - Cnetdb.c: ensure existing IPv4 only functions do indicate an error if passed an IPv6 address. Addition of new functions: Cgetnameinfo(), Cgetaddrinfo(), Cgai_strerror() and new utility Cgetnetaddress()
  - Cdomainname.c, isTrustedHost.c, setnetio.c, use of appropriate storage type and name lookup functions

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Common II

CERN IT Department

- remote.c and getifnam.c were more complicated. e.g. remote.c contains:
  - is_remote(): should determine if an address and hostname is local to the site or not
  - The function will explicitly check environment variables or config files to match on hostname or host IP or IP prefix. Failing that it lists all the configured interface addresses and tries to match the network portion of the address
  - Used ioctl SIOCGIFCONF to list interfaces, attempts to match network part of each to the test address

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Common III

CERN IT Department

- remote.c continued:
  - Extend configuration file parsing to allow specification of IPv6 addresses with a prefix length indicator, e.g.
  - 1:2:3:4::/64
  - SIOCGIFCONF does not list IPv6 configuration on linux, does on Mac OS 10.5. Added use of getifaddrs() or ioctls SIOCGLIFCONF and SIOCGLIFADDR when available.
- getifnam.c
  - Should return an interface name given a socket descriptor.
  - Uses same interface scanning techniques as remote.c
- Scanning interfaces appears to be generally not very portable, this took some time to apply to IPv6.

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

CERN

Gridftp2 DPM plugin

- gridftp2 plugin was changed
  - Avoid cutting a numeric IPv6 address while trying to remove a trailing port number. (Essentially a logging issue)
  - Used gethostbyname() to canonicalise the machine name. Change to use getaddrinfo()

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Status

- No release of LFC or DPM yet with these changes
  - Build candidates which include these changes (and others unrelated to IPv6) have been made and are undergoing tests here: but these tests will only verify IPv4.
  - My own tests during development involved a couple of machines manually configured with IPv6 global unicast addresses
  - Mario (and maybe others) testing at nodes in Paris and Rome.

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Anticipation of problems

CERN IT Department

- No particular problems known, however some places were I'm paying particular attention:
- Don't want to impact on IPv4 performance (our tests exercise that)
- Have used AI_ADDRCONFIG option on getaddrinfo() calls to try to avoid lookups of IPv6 (i.e. AAAA DNS records) by clients not configured with IPv6 interfaces.
  - DNS lookups or connect() attempts on systems or DNS servers not well configured with IPv6 have the potential to introduce unacceptable delays.
- getaddrinfo() is relatively flexible, but consequently more complicated that gethostbyname(). Scope for performance issues or differences between standard library versions. Recent tests on e.g. Mac OS X may help in further exposing any platform or architecture peculiarities.

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Conclusion

- Have shown list of components or subsystems involved in making LFC/DPM IPv6 compliant.
  - Changes were generally localised and similar
  - However quite lot of files involved
  - About 2 weeks initial work (in December)
  - Then an extended period of review; coincides with other functional changes
  - Various changes or fixes also applied in the time since the initial work. Foresee release of 1.6.10 approx end of this month.
  - Other DM components - e.g. GFAL & lcg_utils will need some IPv6 changes; currently not in immediate plan
- Listed files in this presentation for information. Confirm in CVS if you really want to review changes:
  - Most changes added to tag LCG-DM_R_1_6_9_1, few further changes in LCG-DM_R_1_6_10_1

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Contacts

CERN IT Department

- Particulars relating to the IPv6 work David.Smith@cern.ch
- DPM online documentation
  - https://twiki.cern.ch/twiki/bin/view/LCG/DataManagementDocumentation
- Support
  - helpdesk@ggus.org
- General DPM questions
  - hep-service-dpm@cern.ch
- CVSROOT for gLIte, including LCG-DM (LFC & DPM) module:
  - :pserver:anonymous@**jra1mw.cvs.cern.ch:/cvs/jra1mw**
- Web browsing of CVS:
  - http://jra1mw.cvs.cern.ch:8180/cgi-bin/jra1mw.cgi/

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

CERN