


IST-2001-32603	Deliverable D 5.12	
----------------	--------------------	--

Project Number:	IST-2001-32603
Project Title:	6NET
CEC Deliverable Number:	32603/IBM/DS/D5.12/A1
Contractual Date of Delivery to the CEC:	30 June 2004
Actual Date of Delivery to the CEC:	30 June 2004
Title of Deliverable:	IPv6-enabled Globus Toolkit
Work package contributing to Deliverable:	WP5
Type of Deliverable*:	R
Deliverable Security Class**:	PU
Editors:	UCL
Contributors:	UCL, UoS, IBM

* Type: P - Prototype, R - Report, D - Demonstrator, O - Other

** Security Class: PU- Public, PP – Restricted to other programme participants (including the Commission), RE – Restricted to a group defined by the consortium (including the Commission), CO – Confidential, only for members of the consortium (including the Commission)

Abstract:

Deliverable D5.12 describes the 6net activity to combine IPv6 and Grid. UCL is leading the activity, which is focusing mainly on porting Globus Toolkit version 3 to an IPv6-enabled form. The University of Southampton is also involved deeply in the activity. The activity is also carried out in collaboration with the Globus development team in Argonne. The porting activity will be validated by specific demonstrators during the next phase of the project.

Keywords:

IPv6, Grid, Globus, Globus Toolkit

Table of Contents

1	Introduction	5
2	Globus Toolkit 3	6
2.1	Globus Components	6
2.1.1	Security	6
2.1.2	Resource Management	6
2.1.3	Information Services	6
2.1.4	Data Management	7
2.2	Required Tools Used by the Globus Toolkit	7
2.2.1	Java Development Kit (JDK)	7
2.2.2	Jakarta Ant	7
2.2.3	Java DataBase Connectivity (JDBC) database	8
2.2.4	Apache Axis SOAP (distributed with GT3)	8
2.2.5	IBM WSDL4J (distributed with GT3)	8
2.2.6	Globus Java CoG Kit (distributed with GT3)	8
2.2.7	Public-key Infrastructure (PKI)	9
2.2.8	Xindice (distributed with GT 3.0)	9
2.3	Optional Web Service Containers	9
2.3.1	Jakarta Tomcat	9
2.3.2	IBM Websphere	10
2.3.3	Microsoft .NET	10
2.4	GT3 API Analysis	10
2.4.1	Grid Service	10
2.4.2	Grid Security Infrastructure (GSI)	11
2.4.3	Grid Replica Management (GRM)	11
2.4.4	Grid Index Information Service (GIIS)	12
2.4.5	Grid Resource Allocation Manager (GRAM)	12
2.5	GT3 Specific Protocol Analyses	13
2.5.1	GridFTP/RFT	13
2.5.2	Secure Grid Naming Protocol (SGNP)	14
2.5.3	Grid Resource Allocation Agreement Protocol (GRAAP)	14
2.5.4	Grid Resource Registration Protocol (GRRP) and Grid Resource Inquiry Protocol (GRIP)	14
2.5.5	Service Negotiation and Acquisition Protocol (SNAP)	14
2.5.6	eXtensible Input Output library for GT (XIO)	15
2.6	GT3 Generic Protocol Analyses	15
2.6.1	HyperText Transfer Protocol (HTTP)	15
2.6.2	Lightweight Directory Access Protocol (LDAP)	15
2.6.3	Extensible Markup Language (XML)	16
2.6.4	Simple Object Access Protocol (SOAP)	16

2.6.5	Web Services Description Language (WSDL).....	17
2.6.6	Transport Layer Security (TLS).....	17
3	The Experimental Environment and Methodology	18
3.1	Set up of Test-bed.....	18
3.2	Choice of Platforms.....	18
3.3	Methods of Finding IP Dependencies.....	18
3.4	Test Scenarios and Stages	19
3.4.1	IPv6 and IPv4 Testing.....	19
3.4.2	IPv6-only Testing.....	20
3.4.3	IPv6 and IPv4 Operation in Heterogeneous IPv4/IPv6 Networks.....	20
3.4.4	IPv4-only Testing after IPv6-enabled.....	21
3.5	IPv6 Test Services	21
4	Current Status.....	21
4.1	General.....	21
4.2	GGF Standardisation.....	22
4.3	Solved Issues	22
4.4	Setting up IPv6-enabled GT3.....	22
4.5	Issues Still under Investigation or Implementation.....	23
4.5.1	IPv6 tests on GridFTP.....	23
5	Globus Demonstrators.....	23
6	Conclusion and Further Plan.....	23
7	References.....	24
	Appendix A: Details of Solved Issues	26
	Appendix B: IPv6 Relevant Bugs	30
	Appendix C: Non-IPv6 Relevant Bugs.....	32
	Appendix D: Online Guide Documentation “How-to IPv6 in Globus Toolkit 3”	32
	Appendix E: Proposal for a Reusable Grid Service Demonstration Component.....	38

Document history

Date	Action	Author	Comments
10 June 2004	Initial version	Sheng Jiang	Draft 1.0
15 June 2004	Version 1.1	Sheng Jiang	Comments from Peter Kirstein and Piers O'Hanlon
16 June 2004	Version 1.2	Sheng Jiang	Corrections from Peter Kirstein
25 June 2004	Version 1.3	David Mills	Corrections and added Websphere information
30 June 2004	Version 1.4	Tim Chown	Various minor edits/corrections
30 June 2004	Version 1.5	David Mills Tim Chown	Added text from IBM (appendix E). Various minor edits/corrections
30 June 2004	Version 1.6	Sheng Jiang	Update to the latest transition test result, and add eProtein information
1 July 2004	Version 1.7	Piers O'Hanlon	General updates.
2 July 2004	Version 1.8	J.Saint-Blancat	Approved version delivered.

Reviewers: Peter Kirstein, Sheng Jiang, Piers O'Hanlon, (UCL), Tim Chown, David Mills (UoS), Denis Demaugre, François-Etienne Rey, Jacques Saint-Blancat (IBM)

1 Introduction

During the last few years, Grid systems [GridS] have emerged to perform large-scale computation and data storage over IP-enabled data communication networks. They use distributed, potentially remote, resources to optimise computation and storage resources.

Grid systems are normally considered as network middleware [RFC 2768], since they lie between applications and network resources. The data in Grid systems is transported over TCP/IP [TCPIP] – currently using Internet Protocol IPv4 [RFC 791], now twenty years old. The next generation IP - IPv6 [RFC 2460], is expected to replace IPv4 with a number of improvements.

The Globus Toolkit [GlobusW, Globus00], developed mainly in the Argonne National Laboratory (ANL), provides the libraries and services for Grid computing. The current edition of Globus Toolkit – Version 3 (GT3) is based on the latest Grid standards – the Open Grid Services Infrastructure (OGSI) [OGSI10].

GT3 was designed to work with IPv4, though many aspects are compatible with IPv6. Porting Globus Toolkit to be IPv6-enabled will bring considerable advantage into the Grid. The much bigger address space enlarges Grid scaling potential; the mobility support could enable more Grid collaboration applications; auto-configuration mechanisms would simplify the management of the distributed Grid networks; and the better group communication is particularly appropriate for Grid technology. Moreover newer technologies will be deployed over IPv6.

University College London (UCL) and University of Southampton (UoS) started porting Globus Toolkit version 2 to an IPv6-enabled form during the middle of 2002. This almost reached the state of a working system by the end of the year. However, in January 2003 an alpha release of GT3 was produced. This was a radical departure from GT2. The GT3 is so different from GT2 that UCL decided to terminate its work on GT2 forthwith, even without a fully working system, and concentrate on GT3. This activity is being carried out in collaboration with UoS and the Globus development team in ANL. We worked together to include the IPv6 modifications in the official release.

In this document, we begin with the analysis of the Globus components, relevant tools, GT3 APIs and relevant protocols according to the IPv6 consideration. Following that, there is a description of our experimental environment and methodology. Then, we give a brief overview of our current status. The IPv6 issues that we have met and solved during our work are listed in the appendices. At the end of this document, we mention our online support web page and introduce our further plans.

The authors are aware of the latest developments of Globus in respect to the evolution of the Open Grid Services Architecture [phyGrid] to the Web Services Resource Framework [WSRF]. The proposed Globus toolkit version 4 is scheduled to implement these new standards from Organization for the Advancement of Structured Information Standards (OASIS). The timescales for this activity are not aligned with 6net and therefore GT4 will not be covered within 6net during 2004. Should 6net be extended into 2005, then GT4 may be analysed. It is obviously important that the mainstream GT development has IPv6 support in the production release code, and thus 6net has an important role to play in enabling this capability.

2 Globus Toolkit 3

2.1 Globus Components

The following figure provides a general view of Globus protocols and their relationship to external systems:

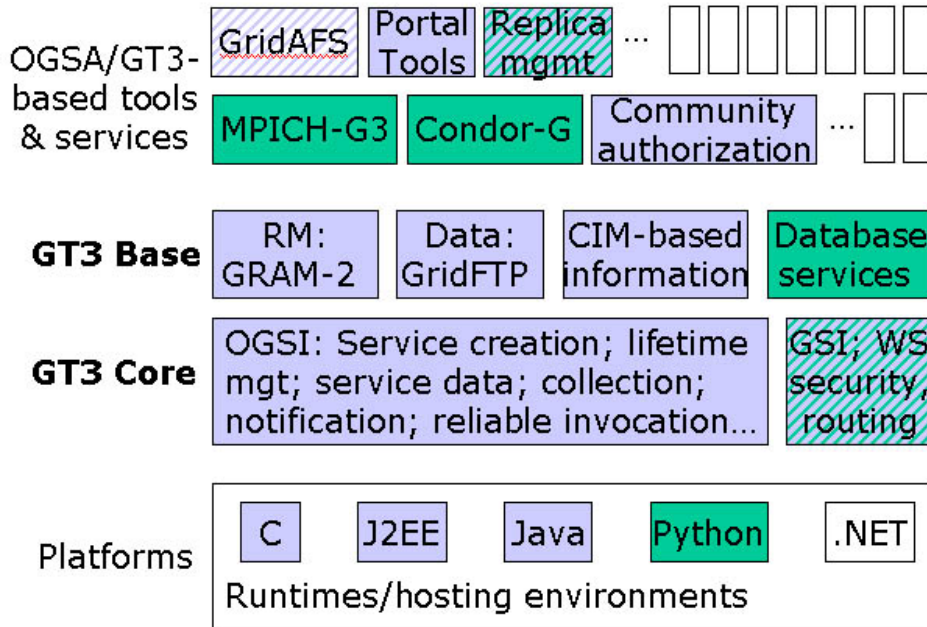


Figure 1: Globus Toolkit hierarchy

2.1.1 Security

Security mechanisms underlie the operation of Grid. These security mechanisms are provided by the Grid Security Infrastructure (GSI), which enables the use of certificates to provide authentication and authorization services.

GSI is a library for providing generic security services for applications that run on the Grid. GSI provides programs to facilitate login to a variety of sites, while each site has its own flavour of security measures.

2.1.2 Resource Management

The Globus Toolkit provides Resource Management, which involves the allocation of Grid resources. It includes such packages as the Globus Resource Allocation Manager (GRAM) and Globus Access to Secondary Storage (GASS).

2.1.3 Information Services

The Globus Toolkit Information Services provide information about Grid resources. Such utilities include the GT3 Index Service.

2.1.4 Data Management

The Globus Toolkit Data Management provides the ability to access and manage data in a Grid environment. This involves such utilities as GridFTP and the Reliable File Transfer (RFT) service, which are used to move files between Grid-enabled devices.

2.2 Required Tools Used by the Globus Toolkit

2.2.1 Java Development Kit (JDK)

2.2.1.1 Description

The main bulk of the Globus toolkit is written in Java, though there are interfaces to other languages available via the Commodity Grid (CoG) Kits.

GT3 was designed based on Java SDK 1.3.1. The JRE works if the user does not need to build any source also users can use the IBM JDK. The JAAS library is required as a separate download if using JDK 1.3.

Most of this work has been performed using the JDK 1.4.X, which provides IPv6 functionality on Linux and Solaris. The IPv6 support in JDK1.4.X provides for most of the required IPv6 functionalities.

Java SDK 1.5beta1 was released in early 2004 with more comprehensive IPv6 support, and support for Windows XP/2003. We recommend the use of Java SDK 1.5 for this work.

2.2.1.2 IPv6 Considerations

There is no IPv6 support within Java SDK 1.3.

JDK1.4 or later provides IPv6 support [jdkv6]. Within Java SDK 1.4 or later, the class `java.net.InetAddress` has two direct subclasses: `java.net.Inet4Address` and `java.net.Inet6Address`. They provide the support for IPv4 and IPv6 addresses. The `InetAddress` class uses the Host Name Resolution mechanisms to resolve host names to their appropriate host address type. However a bug was discovered in Domain Name Service (DNS) IPv6 reverse lookup functionality, which has been fixed in the latest JDK1.5beta release. The IPv6 reverse lookup functionality is necessary for correct operation of GRAM of GT3.

Additionally there are various system preferences that can influence protocol preferences which may be set when JDK initialises. These may be used to influence the start-up behaviour of GT3.

The earlier release, GT 3.0, is not compatible with JDK 1.5. The latest GT 3.2 has fixed these bugs and works well with JDK 1.5. We recommend the use of GT3.2 at the present time.

2.2.2 Jakarta Ant

Apache Ant is a Java-based build tool. It is optional for running GT3 but required for development. Jakarta Ant 1.5 with `optional.jar` is required for source distribution, and recommended for binary distribution. For installation of GT3, `Jakarta-oro.jar` is needed. If want to run tests from source code, Junit must be installed into `$ANT_HOME/lib`.

Jakarta ant is a high level build tool, which is IP protocol independent.

2.2.3 Java DataBase Connectivity (JDBC) database

2.2.3.1 Description

A JDBC compliant database is required for use of the Reliable Transfer Service (RFT) and Master Managed Job Factory Service (MMJFS). Currently Postgresql is recommended. We use Postgresql 7.3 in our test-bed.

2.2.3.2 GT3 Implementation Details

Since the release of Globus Toolkit 3.0, ManagedJob services do not use JDBC for the database connection any more. Instead, Xindice (an embedded database) is used for persisting ManagedJob state. The Reliable File Transfer (RFT) is the only service that uses JDBC for storing the state information.

2.2.3.3 IPv6 Considerations

The currently employed version 7.3 of Postgresql, on the test-bed, doesn't include IPv6 functionality, however an IPv6 patch [PSQLv6] is available. The patched version has been successfully tested for IPv6 operation on the test-bed. The latest version 7.4 of Postgresql includes IPv6 functionality.

2.2.4 Apache Axis SOAP (distributed with GT3)

Apache AXIS [AXIS] is an implementation of the SOAP ("Simple Object Access Protocol") submission to W3C. It is essentially a SOAP engine - a framework for constructing SOAP processors such as clients, servers, gateways, etc. It also includes a simple stand-alone server, a server which plugs into servlet engines, tools for conversion between WSDL and java, and a SOAP monitor tool.

2.2.5 IBM WSDL4J (distributed with GT3)

The Web Services Description Language for Java Toolkit [WSDL4J] allows the creation, representation, and manipulation of WSDL documents describing services.

2.2.6 Globus Java CoG Kit (distributed with GT3)

2.2.6.1 Description

The Java Commodity Grid (CoG) Kit [Jcog] provides access to Grid services through the Java framework. It has underlying support for GT3. There are a number of CoG kits becoming available providing other interfaces such as Python and CORBA.

There are two parts of Java CoG Kit. jglobus contains just the basic components and API's to interface with GT2 and GT3. OGCE contains possible future enhancements and showcases that use of some of the features of jglobus.

2.2.6.2 GT3 Implementation Details

The GT3 implementation includes jglobus as a background library. The GT3 core coding is largely based on the jglobus library.

2.2.6.3 *IPv6 Considerations*

Globus Java CoG Kit's design is based on Java, which can be IP version independent. However, the modification of source code is needed in order to support dual IP version at the same time. (See Solved Issues in Appendix A)

2.2.7 **Public-key Infrastructure (PKI)**

2.2.7.1 *Description*

A Public-key Infrastructure (PKI) is the combination of software, encryption technologies, and services that enables enterprises to protect the security of their communications and business transactions over the Internet. It can be used to provide digital credentials and verify their validity.

However GT3 does not currently interact with a PKI. The Globus Toolkit only checks certificates against a static list of Certification Authorities, and does not query the CA for certification revocation information.

2.2.7.2 *IPv6 Considerations*

The PKI from University of Murcia [UMPKI] supports IPv6. However Grid interfaces to it are currently not implemented. UCL and Murcia are collaborating on PKI as a joint 6net and Euro6IX project venture.

2.2.8 **Xindice (distributed with GT 3.0)**

2.2.8.1 *Description*

Apache Xindice [XIND] is a database designed from the ground up to store XML data or what is more commonly referred to as a native XML database.

2.2.8.2 *GT3 Implementation Details*

Xindice is used for persisting ManagedJob State. Some higher-level services like the Index Service and Execution Services use Xindice as well.

2.2.8.3 *IPv6 Considerations*

As an embedded database, Xindice is not involved with TCP/IP communication. Therefore, it is IP independent.

2.3 **Optional Web Service Containers**

GT3 provides a stand-alone web service container. It is only for test purpose. Jakarta Tomcat has been recommended by the Globus implementation group. A few other web service containers may be used as well.

2.3.1 **Jakarta Tomcat**

Apache Tomcat [TOM] is the servlet container that is used in the official Reference Implementation for the Java Servlet and JavaServer Pages technologies. Since the stand-alone web container is designed only for test purposes, we are currently deploying GT3 core services on Tomcat.

Since we used JDK 1.4 for IPv6 support, the "lightweight edition" of Tomcat - which just excludes several libraries that are already included in the JDK1.4 distribution - has to be used instead of the "full edition" of Tomcat. A bug was discovered that prevented Tomcat responding correctly to IPv6

literal address calls. However the bug has now been fixed in Tomcat 5. Additionally the new JDK1.5beta allows for use of the full version of Tomcat (see Appendix A). Both the lightweight and full versions of Tomcat provide sufficient functionality for GT3.

2.3.2 IBM Websphere

IBM's WebSphere [IBMWS] e-infrastructure software will provide a robust reference implementation for the OGSA. IBM and Globus are working together to retool the Globus Toolkit to be Java 2 Enterprise Edition (J2EE) compliant using IBM WebSphere as the reference application server.

IBM eServer hardware products are all being OGSA-enabled and will take advantage of WebSphere's cross-platform capabilities for executing distributed applications on a Grid.

UoS are investigating the deployment of IPv6-enabled GT3 core services on IBM Websphere and have had reasonable success [6GWebsphere]. Globus services have been deployed and tested within the Websphere environment, although the work involved initially was cumbersome and complex. The processes involved have been developed so that now, with later versions of GT3, it is easier to create a 'Websphere friendly' package of core services that can be readily deployed to run over the Websphere Application Server (WAS).

The next step would be to provide the same integration that Globus affords Tomcat for the Websphere environment. Where GT3 provides built in commands for deploying within Tomcat, so too should a similar deployment method be written for Websphere. Only through collaboration between the Globus team and IBM would this be possible.

The IBM Grid Toolbox (currently version 3) contains a snapshot of GT3 built within their own Grid/Websphere environment.

2.3.3 Microsoft .NET

Microsoft's .NET [MSNET] has been argued as an important potential hosting environment for OGSA-compliant services. It is a set of Microsoft software technologies for connecting information, people, systems, and devices. It enables a high level of software integration through the use of XML Web services - small, discrete, building-block applications that connect to each other as well as to other, larger applications over the Internet.

This has not been analysed by 6net as an IPv6-capable .NET framework will not be available until Q2 2005.

2.4 GT3 API Analysis

2.4.1 Grid Service

2.4.1.1 Description

A Grid service [OGSI10] is a Web service that conforms to a set of conventions (interfaces and behaviours) that define how a client interacts with a Grid service. A client gains access to a Grid service instance through Grid Service Handles (GSH) and Grid Service References (GSR). OGSI provides a mechanism, the Handle Resolver, to support client resolution of a GSH into a GSR.

2.4.1.2 GT3 Implementation Details

GT3 core is a Grid service container, which is built on top of the OGSI primitives and protocols. It provides a run-time environment capable of hosting Grid services.

2.4.1.3 IPv6 Considerations

Grid services are based on Web services. They are upper-layer and IP independent. IP addresses or hostnames are included in handles as part of URIs. However, the handles get URIs from lower-layer functions and invoke the lower-layer functions to build network communication.

2.4.2 Grid Security Infrastructure (GSI)

2.4.2.1 Description

The Globus Toolkit uses the Grid Security Infrastructure (GSI) for enabling secure authentication and communication over an open network [GSI]. GSI provides a number of useful services for Grids, including mutual authentication and single sign-on. GSI is based on public key encryption, X.509 certificates, and the Secure Sockets Layer (SSL) communication protocol.

GT2 used the Secure Socket Layer (SSL) protocol for its authentication and message protection. GT3, which is based on the emerging Web Services technologies and the Open Grid Services Architecture, leverages Web Services for these functions. GT3 implements a session based security service similar to what is described in the WS-Trust and WS-SecureConversation documents. The GT3 implementation (GSI-SecureConversation) allows for GSI's SSL-based authentication to take place over standard Web Services SOAP messages, which in turn allows for the use of Web Services security specifications for message protection (WS-Security, XML-Encryption and XML-Signature). In addition to the session based security mechanism GT3 also provides WS-Security and XML-Signature based per-message security using standard public key cryptography (GSI-SecureMessage).

2.4.2.2 GT3 Implementation Details

The Globus Toolkit is an implementation of the GSI that adheres to the Generic Security Service API (GSSAPI), which is a standard API for security systems promoted by the Internet Engineering Task Force (IETF). The GT3 security library is still accessible through the GSSAPI, as defined by RFC 2743 with extensions as defined by the Global Grid Forum GSS-extensions document. The following Java package provides most of the GSI functionality in Java CoG Kit library:

org.globus.gsi

2.4.2.3 IPv6 Considerations

In RFC2743 the GSAPI uses the concept of channel bindings which are protocol independent as far as GSSAPI is concerned. The implementation of the communication channel lies in the GSI package at:

org.globus.gsi.gssapi.net

This utilises java socket calls, which are protocol independent.

2.4.3 Grid Replica Management (GRM)

2.4.3.1 Description

Replica management is an important issue for a number of scientific applications. As data collections grow up to the petabyte level, it is likely that few universities, research laboratories or individual researchers will have sufficient storage to hold a complete copy. Instead, they will store copies of the most relevant portions of the data set on local storage for faster access. Replica Management is the process of keeping track of where portions of the data set can be found. It

maintains a mapping between logical names for files and collections and one or more physical locations.

2.4.3.2 GT3 Implementation Details

So far, we do not see a GRM implementation in GT3, but we believe it will be included in later versions of GT3 since replica management is important in high-performance data grids. In GT2, replica management implementation involved a software API to an associated library, and a command-line tool providing the same functionality. All of these were in standard C.

2.4.4 Grid Index Information Service (GIIS)

2.4.4.1 Description

The Grid Index Information Service (GIIS) provides a means of knitting together arbitrary GRIS services to provide a coherent system image that can be explored or searched by grid applications. GIISs thus provide a mechanism for identifying "interesting" resources, where "interesting" can be defined arbitrarily. For example, a GIIS could list all of the computational resources available within a confederation of laboratories, or all of the distributed data storage systems owned by a particular agency. A GIIS could pool information about all of the grid resources (computation engines, data, networks, instruments) in a particular research consortium, thus providing a coherent system image of that consortium's computational grid.

As part of this information infrastructure, the Index Service uses an extensible framework for managing static and dynamic data for Grids built using the Globus Toolkit 3.0. The functionality provided includes the following:

- Dynamic service data creation and management via information provider programs
- Aggregation of service data from multiple instances
- Registration of Grid service instances

There are two ways to access the service data aggregated by the Index Service: the GT3 Service Data Browser and the `findServiceData` command.

2.4.4.2 GT3 Implementation Details

The interfaces and functions most relevant to the Index Service can be summarized as follows: Factory, Grid Service Handle (GSH), Grid Service Reference (GSR), Query, Registry, and Notification.

2.4.4.3 IPv6 Considerations

Since GIIS is implemented as one of the upper layer Grid Service in GT3, its IP support is based on the IP support for general Grid Services.

2.4.5 Grid Resource Allocation Manager (GRAM)

2.4.5.1 Description

The Grid Resource Allocation Manager (GRAM) is the lowest level of Globus resource management architecture. GRAM allows user to run jobs remotely, using a set of WSDL/OGSI client interfaces for submitting, monitoring, and terminating a job in GT3. It allows the submitting

and cancelling of jobs, polling for job status and sending signals to a job. The library enables a user to ‘ping’ a gatekeeper to verify if the user can authenticate to it.

2.4.5.2 *GT3 Implementation Details*

GRAM in GT3 is outside of the main GT3 core. Under package *program_execution*, package *mjs* (the Managed Job Service) and *mmjfs* (the Master Managed Job Factory Service) provide the main GRAM service.

The following Java package in Java CoG Kit library also provides support for the GRAM functionality:

org.globus.gram

2.4.5.3 *IPv6 Considerations*

GRAM uses URIs to locate the remote host for jobs. It invokes the lower-layer functions to get URIs and build network communications.

2.5 **GT3 Specific Protocol Analyses**

This section contains an assessment of GT3 protocols. Work has been carried out to analyse all of the Global Grid Forum protocols [GGF] within their IPv6 Working group, which is led by 6net members.

2.5.1 **GridFTP/RFT**

2.5.1.1 *Description*

The Reliable File Transfer (RFT) is an OGSA-based service that provides interfaces for controlling and monitoring 3rd party file transfers using GridFTP servers. GridFTP is based on the FTP protocol [RFC 959] and provides a file transfer service linked with grid security mechanisms.

2.5.1.2 *GT3 Implementation Details*

GT2 had a standard C version GridFTP, which is not OGSA-compliant. So far, GridFTP, which is included in GT3 packages, is still written in standard C. ANL is developing an eXtensible Input Output library (XIO), which will be used to rewrite GridFTP. The Globus implementation group is currently working on the Java GridFTP APIs. The following Java package provides most of the FTP functionality currently available in Java CoG Kit library:

org.globus.ftp

2.5.1.3 *Protocol Dependencies*

The specification for the File Transfer Protocol assumes that the underlying network protocol uses a 32-bit network address (specifically IPv4).

2.5.1.4 *IPv6 Considerations*

Since GridFTP is largely based on FTP [RFC959] the considerations in [RFC2428] should be taken into account. In this specification, the FTP commands PORT and PASV are replaced with EPRT and EPSV, respectively. These dependencies are noted in the GridFTP working drafts [gftp].

2.5.2 Secure Grid Naming Protocol (SGNP)

2.5.2.1 Description

SGNP provides a mechanism for the "Naming" of Grid resources. It defines a scheme that assigns "logical" and thus location-independent names to Grid resources. The scheme obviates the need for authentication of two Grid resources via a trusted third party. The logical name is a combination of the identity and security information of a Grid resource.

2.5.2.2 GT3 Implementation Details

The Secure Grid Naming Protocol (SGNP) has been proposed to the Grid Forum to alleviate the location-dependency and security problems that arise with GSH and the HandleMap service of OGSA.

2.5.3 Grid Resource Allocation Agreement Protocol (GRAAP)

2.5.3.1 Description

The Grid Resource Allocation Agreement Protocol addresses the issues between a Super-Scheduler (Grid Level Scheduler) and local Schedulers necessary to reserve and allocate resources in the Grid as a building block for this service.

2.5.3.2 GT3 Implementation Details

As it is still a working item within the GGF Grid Resource Allocation Agreement Protocol Working Group, we cannot give implementation details or investigate IPv6 considerations here.

2.5.4 Grid Resource Registration Protocol (GRRP) and Grid Resource Inquiry Protocol (GRIP)

2.5.4.1 Description

The Grid Resource Registration Protocol (GRRP) and the Grid Resource Inquiry Protocol (GRIP) provide for soft-state registration and resource discovery and are used to construct the Monitoring and Discovery Service. These protocols are both embedded into the Lightweight Directory Access Protocol (LDAP).

The Grid Resource Registration Protocol is used to notify aggregate directory services of the availability of information about entities, while GRIP is used to access this information.

2.5.5 Service Negotiation and Acquisition Protocol (SNAP)

2.5.5.1 Description

The Service Negotiation and Acquisition Protocol provides lifetime management and an at-most-once creation semantics for remote Service Level Agreements (SLAs). The result is a resource management framework for distributed systems, which is more powerful and generic than current approaches.

2.5.5.2 Implementation Details

As it is still a working item within Globus, we cannot give implementation details or investigate IPv6 considerations as yet.

2.5.6 eXtensible Input Output library for GT (XIO)

2.5.6.1 Description

The eXtensible Input Output library for the Globus Toolkit provides a posix-like API (open/close/read/write) to swappable IO implementations. It provides a single-user API to all Grid IO Protocols and minimise the development time for creating/prototyping new protocols.

2.5.6.2 Implementation Details

Globus XIO and the rewritten Grid-FTP have only been released since April 2004. The Globus implementation group has designed it with IPv6 in mind.

2.5.6.3 IPv6 Considerations

The Globus XIO is protocol independent. It has been written with IPv6-compatibility in mind. That means IP independent data structures and functions are used. However, the new Grid-FTP is only working with IPv4 so far. The Globus implementation group is still working towards IPv6-enabled.

2.6 GT3 Generic Protocol Analyses

2.6.1 HyperText Transfer Protocol (HTTP)

2.6.1.1 Description

The Hypertext Transfer Protocol (HTTP) [RFC 2068] is an application-level protocol for distributed, collaborative, hypermedia information systems. It is a generic, stateless, object-oriented protocol which can be used for many tasks. A feature of HTTP is the negotiation of data representation, allowing systems to be built independently of the development of new advanced representations. HTTP is used to transport a variety of types of data. In the case of GT3, transport of SOAP and XML are of particular interest.

2.6.1.2 GT3 Implementation Details

The following Java package provides most of the HTTP functionality in Java CoG Kit library:

org.globus.util.http

However HTTP underlies a large part in GT3 (e.g. SOAP etc) so it is difficult to compile a comprehensive list of relevant files. HTTP is also implemented as part of the JDK.

2.6.1.3 IPv6 Considerations

When IPv6 addresses are used in Uniform Resource Identifiers (URIs) the literal format for IPv6 addresses described by RFC 2732 (and in the standardisation work 6net is contributing to the GGF [IPGGGF]) should be supported.

2.6.2 Lightweight Directory Access Protocol (LDAP)

2.6.2.1 Description

The Lightweight Directory Access Protocol (LDAP) [RFC2251] is designed to provide access to directories supporting the X.500 models. This protocol is specifically targeted at management applications and browser applications that provide read/write interactive access to directories. When used with a directory supporting the X.500 protocols, it is intended to be a complement to the X.500 DAP. The services may be stand-alone or part of a distributed directory service.

2.6.2.2 *GT3 Implementation Details*

The Monitoring and Discovery Service (MDS) restricted clients to queries using the LDAP protocol, which has a fairly restrictive query language.

The OGSi core in GT3 will provide a generic interface for mapping Service Data queries and subscriptions for Service Data notification to service implementation mechanisms. In essence, this subsumes the role of the GRIS backend server module in MDS, while relying on more basic OGSA binding mechanisms for secure access to the query interface, in place of the GSI-enabled LDAP protocol.

2.6.3 Extensible Markup Language (XML)

2.6.3.1 *Description*

The Extensible Markup Language (XML) is a subset of SGML. Its goal is to enable generic SGML to be served, received, and processed on the Web in the way that is now possible with HTML. XML has been designed for ease of implementation and for interoperability with both SGML and HTML. XML is standardised [XML00] by the W3C.

2.6.3.2 *GT3 Implementation Details*

The following Java package provides some of the XML functionality in Java CoG Kit library:

org.globus.xml

However XML is used throughout GT3, and is also supported within the JDK, so it is difficult to compile a comprehensive list of relevant packages.

2.6.3.3 *IPv6 Considerations*

XML is a high level language, which is IP-protocol independent.

2.6.4 Simple Object Access Protocol (SOAP)

2.6.4.1 *Description*

The Simple Object Access Protocol (SOAP) is a lightweight protocol for exchange of information in a decentralized, distributed environment. It is an XML-based protocol that consists of three parts; an envelope that defines a framework for describing what is in a message and how to process it, a set of encoding rules for expressing instances of application-defined data types, and a convention for representing remote procedure calls and responses. Potentially, SOAP can be used in combination with a variety of other protocols.

2.6.4.2 *GT3 Implementation Details*

GT3 currently supports SOAP over HTTP, so GSRs are in WSDL format. SOAP is largely used in GT3 as a communication protocol.

2.6.4.3 *IPv6 Considerations*

The use of IP addresses in Uniform Resource Identifiers (URIs) [RFC2396] should be avoided whenever possible [RFC1900]. However, when used, the literal format for IPv6 addresses in URIs as described by RFC 2732 should be supported.

2.6.5 Web Services Description Language (WSDL)

2.6.5.1 Description

OGSI is based on Web services, and in particular uses WSDL as the mechanism to describe the public interfaces of Grid services [OGSI10].

The Web Services Description Language (WSDL) is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint. Related concrete endpoints are combined into abstract endpoints (services). WSDL is extensible to allow description of endpoints and their messages regardless of what message formats or network protocols are used to communicate. WSDL is being standardised by the W3C [WSDL11].

WSDL does not introduce a new type definition language. WSDL recognizes the need for rich type systems for describing message formats, and supports the XML Schemas specification (XSD) as its canonical type system. However, since it is unreasonable to expect a single type system grammar to be used to describe all message formats present and future, WSDL allows using other type definition languages via extensibility.

WSDL defines a common binding mechanism. This is used to attach a specific protocol, data format or structure to an abstract message, operation, or endpoint. It allows the reuse of abstract definitions. In addition to the core service definition framework, there are specific binding extensions for the following protocols and message formats: SOAP 1.1, HTTP GET / POST, MIME.

2.6.5.2 GT3 Implementation Details

The following Java package provides some of the WSDL functionality in GT3 core:

org.globus.ogsa.wSDL

However IBM's WSDL4J package also provides WSDL support.

Because WSDL 1.1 is deficient in two critical areas - lack of interface inheritance and lack of open content - OGSi define a separate (and temporary namespace) with the prefix *gwsdl*. *Gwsdl* is "work in progress" within WSDL 1.2 [WSDL12].

2.6.5.3 IPv6 Considerations

In GT3 WSDL is transported using SOAP over HTTP. All the IPv6 considerations for SOAP and HTTP must be extended to WSDL.

When working with WSDL, Uniform Resource Identifiers (URIs) are used. Currently IP addresses appear to be utilised when services are advertised. For protocol independence, hostnames should be used instead. Configurations of preferences or policies are necessary to provide GT3 with mechanisms to control protocol choices.

2.6.6 Transport Layer Security (TLS)

2.6.6.1 Description

A slightly modified version of Transport Layer Security [RFC2246], based on version 3 of Secure Sockets Layer, enables application level encrypted communications between two nodes. The protocol is composed of two layers, the TLS Record Protocol and the TLS Handshake Protocol. The

TLS Record Protocol runs at the lowest level, sitting above some reliable transport protocol (e.g. TCP). This is used in combination with the proxy certificates [PRXCT] to provide transport layer security.

2.6.6.2 GT3 Implementation Details

OGSA provides two different GSI implementations. One is based on message-level security, and the other on transport-level security, which version 2 of the Globus Toolkit (GT2) uses. Transport-level security is supported in order to be backwards compatible with GT2, and also to simplify integration with hosting environments without message-level security support. GSI defines protocols for mutual authentication, credential delegation, proxy signing, message protection, and authorization, and typically sits on top of a Secure Sockets Layer/Transport Layer Security (SSL/TLS) implementation. The “httpg” embodies HTTP over a specialised version of TLS [httpg] which is implemented by:

org.globus.net.protocol.httpg

2.6.6.3 IPv6 Considerations

TLS is largely protocol independent as it uses an externally supplied transport mechanism. However an implementation of the protocol needs to be aware of the underlying transport.

3 The Experimental Environment and Methodology

3.1 Set up of Test-bed

While there are many large Globus systems in operation – most are IPv4 only. We have set up at UCL-CS a system with around 10 nodes, running a mixture of IPv4 only, IPv6 only and dual stack nodes. This would allow all the necessary combinations of inter-working between Globus implementations with different network support. The test-bed can be made available to any group concerned with the porting activity.

While many Globus implementers will not be short of platforms to use, they might still appreciate access to such a test-bed; as a result they would not have to set an IPv6 infrastructure first.

3.2 Choice of Platforms

For the time being, we restrict ourselves to the LINUX platform. We do not intend currently to port the system to a variety of platforms.

The GT3 stand-alone web service container is our default testing web container. We also work on Tomcat, which is recommended by the Globus implementation group. IBM Websphere is also under consideration as an alternative GT3 OGSA service container. (More detail in Section 2.3)

3.3 Methods of Finding IP Dependencies

To find out exactly which lower-layer protocols and APIs are being used, two approaches are taken – firstly the ‘top down’ approach where we execute some upper layer applications. Secondly the ‘bottom up’ approach where we monitor, using *ethereal/tcpdump* [NETMON], all the data traffic between nodes and on the Loopback interface. The functions involving any of the following, have been identified as relevant; they have been modified to be IP-independent:

- General network functions
- Obtaining or generation IP addresses
- Generation of URLs and URIs with IP addresses

- Hard-coded IPv4 addresses
- Linkage to external network based libraries

Additionally external libraries that have been found to be IPv4 dependent have been investigated and recommended modifications sent to the relevant developers.

3.4 Test Scenarios and Stages

The following scenarios are tested in our porting and testing in our UCL GT3 test-bed, see figure 2. According to the details of our testing feedback, the code porting has been made in order to lead us to final IPv6-enabled OGSA.

Up to now, most of the GT3-required external tools have been set up and tested. As mention earlier, they have been built for our GT3 IPv6 tests. The IPv6 support from the internal library, such as Java CoG, Axis, etc has also been investigated.

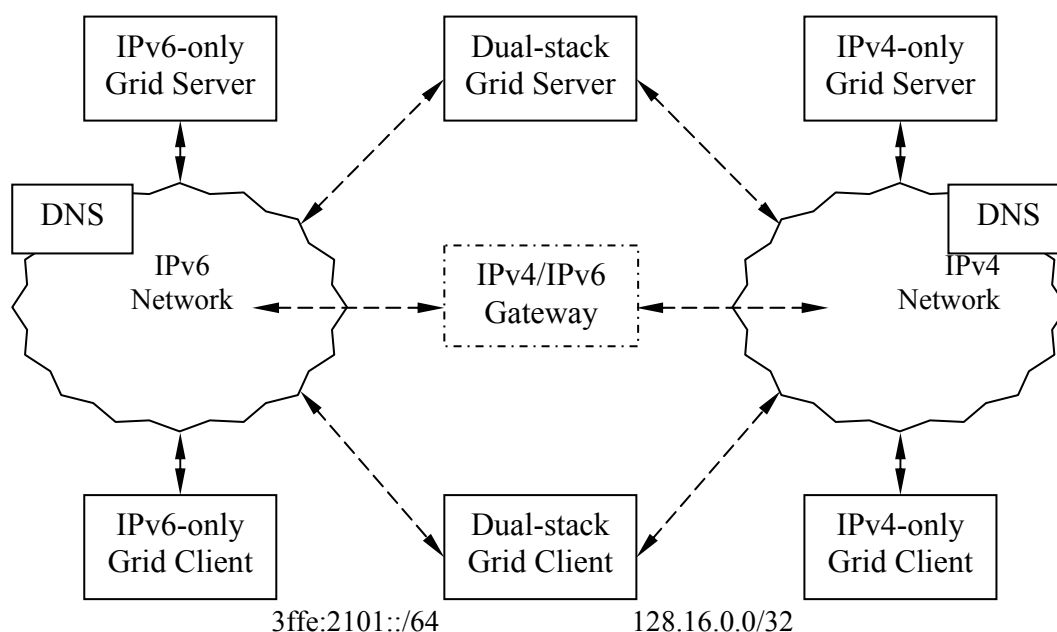


Figure: IP Transition in heterogeneous Grid networks

Initial testing involves the use of the simple tests provided by the Globus group. We also tested systems with externally developed GT3 services as well.

3.4.1 IPv6 and IPv4 Testing

Initially systems were tested on dual-stack machines to ascertain which parts of GT3 require IPv4, allowing progression to IPv6-only environments.

IPv4-only testing is based on our particular experiment environment, which gives the external support for potential IPv6 testing. GT3 has been tested to work in this environment.

3.4.2 IPv6-only Testing

The required modifications were derived from the bottom-up and top-down analyses. This also includes investigation of operation over various transitional approaches such as IPv4-mapped-IPv6 addresses.

We started IPv6-only testing early in May, 2003. We have successfully got the client working on an IPv6-only host, which can communicate with an IPv6-only server.

3.4.3 IPv6 and IPv4 Operation in Heterogeneous IPv4/IPv6 Networks

Since there will be a period of IP transition, consideration must be given to an interim coexistence of IPv4 and IPv6. Our effort to integrate an API within the Grid systems is IP-protocol independent, i.e., it supports both IPv4 and IPv6. The IP-independent server has to be able to respond to client calls according to the IP family that the client uses. Since IPv4-only machines will exist for many more years, while IPv6-only machines are starting to appear, it is necessary to provide support to both IPv4-only and IPv6-only environments. For the communication in heterogeneous IPv4/IPv6 networks, there are many approaches to the provision of transition aids; examples include the tunnelling approach, which links IPv6 islands established over IPv6 networks together, and a protocol translation approach. They need to be considered when building an IPv6 environment within or around current global IPv4 networks.

On dual-stack machines, the Grid server starts with the hostnames, which are IP-independent. It is down to the client to determine which version of IP is used. The Grid servers respond to the client calls according to which IP family the client uses. For instance, an IP-independent Grid server starts with its hostnames. There is no particular IP address binding to the server, and both IPv4 and IPv6 interfaces are accessible. When an IPv4 client calls with IPv4 addresses, the Grid server uses its IPv4 interface to respond; only IPv4 communication takes place. Similarly, when an IPv6 client calls with IPv6 addresses, the Grid server uses its IPv6 interface to respond; only IPv6 communication takes place. For clients on dual-stack machines, the client can choose which IP family is the default or preferred.

The situation becomes complicated when an IPv6-only client requires access to an IPv4-only server. It involves at least one specially configured transition gateway, for which we use a NAT-PT [RFC2766] gateway in our experiments. A specially configured IPv6 DNS server is involved as well. The operation in these transition scenarios is generic as the IPv4/IPv6 transition for any application. When an IPv6-only Grid client launches a job referring to a hostname, it sends a name lookup request to the specially configured IPv6 DNS server. In the normal DNS behaviour, if the DNS server finds the matching IPv6 address, it returns the address to the client. Then the IPv6 connection between the IPv6 client and the IPv6 server will be built up. If there is no matching IPv6 address, the specially configured DNS server returns a special IPv6 address, which is in a particular prefix (for example, we use 2222::1:0:0/96 in our experiments) and includes the IPv4 address of the destination host in the IPv4-compatible IPv6 address format. With specially configured routing on the IPv6 client, all the packets that have the particular prefix in their destination addresses are sent to an IP transition gateway, which has both IPv6 and IPv4 interfaces. The gateway receives the IPv6 packets through its IPv6 interface and gets the IPv4 destination address as well, then re-packages the packets with its own address as the source address, and sends them towards the IPv4-only destination server through its IPv4 interface. A mapping section that records this pair of the source IPv6 address and the allocated gateway IPv4 address is kept on the NAT-PT gateway for a while before it is removed or renewed. The return packages go through the IP transition gateway using the information recorded in the mapping section.

In the opposite direction, an IPv4-only client access to an IPv6-only server experiment is fairly similar. When an IPv4-only Grid client launches a job to an IPv6-only Grid server, it will not receive a valid IPv4 destination address from the DNS server. The DNS server returns one of the gateway addresses to the client. Every gateway IPv4 address could be statically mapped to an IPv6-only server. For the dynamic address mapping, a DNS-ALG (Domain Name Server – Application Level Gateway) is required. According to these maps, the NAT-PT gateway repackages all IPv4 packets and sends them towards the IPv6-only destination server. The dynamic mapping section that records this pair of the allocated gateway IPv4 address and the destination IPv6 address is kept on the NAT-PT gateway for a while before it is removed or renewed. The return packages go through the IP transition gateway using the information recorded in the mapping section. To succeed in the above scenarios, the Grid systems should only have hostnames in their content of the payload rather than any IP addresses. If any IP address was passed in the packet's content, it would lead to later failure if that IP address was used.

The general consensus in the IETF IPv6 Operations WG is that wherever possible a node wishing to communicate with an IPv4 or IPv6 endpoint should support IPv4 or IPv6 as necessary. However, this can only be an interim measure (dual-stack) towards an IPv6-only end game (which could be many years away).

3.4.4 IPv4-only Testing after IPv6-enabled

After we had successfully made Globus Toolkit IPv6-enabled, we ran IPv4-only tests as well, since most Globus users are obviously still IPv4.

3.5 IPv6 Test Services

The services or applications that are distributed with GT3 are used as general test services in our test scenarios. The OGSA graphic user interface service browser is used to access all available Grid Services. The graphic user interface data browser is used to access all Grid data. The GRAM is used to submit jobs remotely. In our preliminary test, we ran the command line GramClient with the *test.xml* file.

The OGSA services are built on web services. The stand-alone web container, which is distributed with GT3, is only really designed for test purposes and so both Tomcat and Websphere web containers were used as well.

We also developed a number of test services that run on GT3, which will exercise the various parts of the system. The eProtein project in UCL had developed a remote executing service based on GT3 GRAM. UoS are working on the Grid-based Medical Devices for Everyday Health (GBMD4EH) architecture as their test OGSA services.

4 Current Status

4.1 General

UCL has set up a GT3 test-bed on which a number of tests, see Section 3.4, have been run. Our test scenarios and code porting are mounted on this test-bed.

UCL has surveyed all GT3 components and source code for IP dependencies. A number of issues have been identified and solved, see Section 4.3. Based on UCL's effort to solve these issues, we have successfully demonstrated full IPv6 functionality within GT3, except for GridFTP. We worked with the Globus implementation group to include the IPv6 modification in the official

release. Since GT3.2, IPv6 support has been provided in the official release and no modification is required.

For full IPv6 functionality, Java SDK 1.5 is required.

UoS were investigating to deploy IPv6-enabled GT3 core services on IBM Websphere and had got a reasonable success [6GWebsphere], see Section 2.3.2.

4.2 GGF Standardisation

Since February 2003, when the Global Grid Forum (GGF) and the IPv6 Forum announced a liaison relationship to drive New Generation Applications deployment worldwide, IPv6 has become relevant to Grid activities.

An IPv6-Grid Working Group has been set up in GGF. The GGF IPv6 WG [GGFv6] has first presented on two work-in-progress drafts in GGF9 (October 2003): “IP version dependencies in GGF specifications” and “Guidelines for IP independence in GGF specifications”. UCL made contributions to both drafts. Both drafts were updated in GG10 (March 2004), where the third draft “Status for Java Developers Kit API for IPv6” was presented and submitted.

The 6net project edited and contributed heavily to the IP independence drafts, and contributed to the Java IPv6 draft. Both chairs of the GGF IPv6 WG are participants in the 6net project (from UCL and IBM).

These drafts have all now been through WG Last Call and are available as GGF standards [IPSGGF, IPGGGF].

4.3 Solved Issues

During our GT3 and IPv6 testing and porting, a number of issues have been identified, such as Java version dependency, hard-coded IP addresses, literal IPv6 address handling implementation, etc. Most of them have been successfully solved. We list all solved issues in the Appendix A with brief descriptions of their solution. With these issues solved, we are able to set up an IPv6-enabled GT3, see Section 4.4. There are still two issues under investigation, see Section 4.5.

4.4 Setting up IPv6-enabled GT3

UCL has successfully demonstrated full IPv6 functionality in GT3. UCL also maintains an online support documentation (at the following URL, also see Appendix D), which describes how to make GT3 work with IPv6 and provides UCL modified binary jar files and source code patches for earlier GT3 releases. It has already become one of the official Globus technology reference documentations.

<http://www.cs.ucl.ac.uk/staff/sjiang/webpage/how-to-IPv6-Globus.htm>

We outline the contents of the documentation as following

- Operating System Support on Hosts for IPv6
- Networking Support for IPv6
- IPv6 Consideration or Request of Associated Applications
- IPv6 Configuration of GT3
- Running Tests with IPv6

- Further modification of IPv6-enabled GT3
- Appendix: the installation from the latest cvs

4.5 Issues Still under Investigation or Implementation

The following issues have been identified during our GT3 and IPv6 development, but we have not yet found any practical solutions.

4.5.1 IPv6 tests on GridFTP

Argonne has rewritten GridFTP using Globus XIO in order to avoid licensing issues. The Globus XIO has been written with IPv6 compatibility in mind. However, the Globus XIO based Grid-FTP is so far only working with IPv4. The Globus implementation group is continuing to work on it.

5 Globus Demonstrators

With the IPv6 capability now available in GT3, UCL and UoS are now building demonstrators of the technology in action.

The two primary demonstrators are:

- IPv6 Weather Station (UoS)

The Weather Station from UoS is a Globus Toolkit-driven weather station monitoring system. It runs over an IPv6-enabled Grid weather sensor network.

- IPv6 eProtein (UCL)

eProtein is a large protein analysis from UCL. It runs the large computational jobs over the IPv6-enabled Grid infrastructure.

There are two separate report documents for these demonstrators available from the 6net website.

6 Conclusion and Further Plan

The work to provide IPv6 support for Globus has involved extensive investigations of the Globus Toolkit source code and necessitated investigations into a wide range of dependent and related technologies. These investigations have yielded information necessary to proceed with the construction of a test-bed which offered the Globus Toolkit running in an IPv6 capable system. The test-beds then aided in testing and debugging of IPv6 issues, which have been fed back into the Globus core and its related technologies.


The work has been referenced by the Globus developers and now provides a means for deployment of IPv6 capable Globus nodes by third parties. The development of Globus is on going and so there continues to be areas of work that require further investigation. These have been identified and will continue to be developed within the timeframe of the 6net project. The approaching developments will, however, aim to primarily assist in preparation of demonstrations of Grid projects running on the capable IPv6 Globus system.

The work has also been beneficial in establishing new standards documents within the GGF, and validating the application transition guidelines (IPGGGF) from the IETF IPv6 Operations WG.

The next step will be deployment of Globus demonstrators at multiple 6net partner sites during the next phase of the project.

7 References

- [RFC 791] ISI, “Internet Protocol DARPA Internet Program Protocol Specification”, 1981.
- [RFC 959] J. Postel, and J. Reynolds, “File Transfer Protocol (FTP)”, October 1985.
- [RFC 1900] B. Carpenter, Y. Rekhter, “Renumbering Needs Work”, February 1996.
- [RFC 2068] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, T. Berners-Lee, “Hypertext Transfer Protocol -- HTTP/1.1”, January, 1997.
- [RFC 2251] M. Wahl, T. Howes, S. Kille, “Lightweight Directory Access Protocol (v3)”, December 1997.
- [RFC 2246] T. Dierks, C. Allen, “The TLS Protocol”, January 1999.
- [RFC 2396] T. Berners-Lee, R. Fielding, L. Masinter, “Uniform Resource Identifiers (URI): Generic Syntax”, August 1998.
- [RFC 2428] M. Allman, S. Ostermann, C. Metz, “FTP Extensions for IPv6 and NATs”, September, 1998.
- [RFC 2460] S. Deering, R. Hinden. “Internet Protocol, Version 6 (IPv6) Specification”, December 1998.
- [RFC 2732] R. Hinden, B. Carpenter, L. Masinter, “Format for Literal IPv6 Addresses in URL’s”, December 1999.
- [RFC 2743] J. Linn, “Generic Security Service Application Program Interface”, January 2000.
- [RFC 2766] G. Tsirtsis, “Network Address Translation - Protocol Translation (NAT-PT)”, 2000.
- [RFC 2768] B. Aiken, “Network Policy and Services: A Report of a Workshop on Middleware”, February 2000.
- [RFC 3542] W. Stevens, M. Thomas, E. Nordmark, T. Jinmei, “Advanced Sockets Application Program Interface (API) for IPv6”, May 2003.
- [TCPIP] D. Comer, “Internetworking with TCP/IP, Volume 1, 4th edition,” Published by Prentice Hall, 2000.
- [phyGrid] I. Foster, C. Kesselman, J. Nick, S. Tuecke, “The Physiology of the Grid – An Open Grid Services Architecture for Distributed Systems Integration”
<http://www.Globus.org/research/papers/OGSA.pdf>
- [OGSI10] S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maquire, T. Sandholm, D. Snelling, P. Vanderbilt, “Open Grid Service Infrastructure (OGSI) – Version 1.0 (draft)”, <http://www.ggf.org/OGSI-wg>
- [GridS] I. Foster, “The Grid: A New Infrastructure of 21st Century Science”, Physics Today Volume 55: 42-52, 2002.
- [Globus00] I. Foster, C. Kesselman, “Globus: A Metacomputing Infrastructure Toolkit”, Intl J. Supercomputer Applications Volume 11(2): 115-128, 1997.
- [WSRF] Web Services Resource Framework, <http://www.globus.org/wsrfl/>
- [GlobusW] Globus project official website, <http://www.globus.org>

IST-2001-32603	Deliverable D 5.12	
----------------	--------------------	---

[CoGM] The Java CoG Kit User Manual – Draft Version 1.1a, <http://www.Globus.org/cog/manual-user.pdf>

[jdkv6] Networking IPv6 User Guide for J2SDK/JRE 1.4, http://java.sun.com/j2se/1.4/docs/guide/net/ipv6_guide/

[gftp] GridFTP: Protocol Extensions to FTP for the Grid, April 2002, http://www.gridforum.org/meetings/ggf6/ggf6_wg_papers/GridFTP.doc

[XML00] Extensible Markup Language (XML) 1.0 (Second Edition) <http://www.w3.org/TR/REC-xml>

[WSDL11] Web Services Description Language (WSDL) 1.1, <http://www.w3.org/TR/wsdl>

[WSDL12] Web Services Description Language (WSDL) 1.2, <http://www.w3.org/TR/wsdl12>

[Jcog] Java COG Kit, <http://www-unix.globus.org/cog/java/>

[GSI] GT3 Grid Security Infrastructure, <http://www.globus.org/Security/GSI3/>

[httpg] Globus Toolkit 3 Core – A Grid Service Container Framework. http://www-unix.globus.org/toolkit/3.0/ogsa/docs/gt3_core.pdf

[Gram] GT3 GRAM Overview http://www-unix.globus.org/ogsa/docs/alpha/gram/gt3_gram_overview.htm

[GGF] Global Grid Forum website, <http://www.ggf.org>

[GGFv6] Global Grid Forum IPv6 Working Group, <http://forge.gridforum.org/projects/ipv6-wg/>

[IPSGGF] Rute Sofia, GGF standard, “Survey of IPv4 Dependencies in Global Grid Forum Specifications”.

[IPGGGF] Tim Chown, GGF draft, “Guidelines for IP version independence in GGF specifications”.

[6GWebsphere] Report on deploying Globus Toolkit v3.0.2 within Websphere's Application Server, over IPv6 <http://www.ecs.soton.ac.uk/~dgm/work/>

[PSQLv6] Postgresql-7.3 IPv6 patch: <http://www.lbsd.net/>

[AXIS] Apaches AXIS: <http://ws.apache.org/axis/index.html>

[WSDL4J] IBM WSDL4J: <http://www-124.ibm.com/developerworks/projects/wsdl4j/>

[UMPKI] UMU PKI: <https://pki.umu.euro6ix.org/pkiv6.html>

[XIND] Apache Xindice: <http://xml.apache.org/xindice>

[TOM] Apache Tomcat: <http://jakarta.apache.org/tomcat/>

[IBMWS] IBM WebSphere <http://www.ibm.com/websphere>

[MSNET] Microsoft .NET <http://microsoft.com/net/>

[PRXCT] IETF Draft: Internet X.509 Public Key Infrastructure Proxy Certificate Profile <http://www.ietf.org/internet-drafts/draft-ietf-pkix-proxy-10.txt>

[NETMON] Ethereal and Tcpdump: ethereal.org tcpdump.org

Appendix A: Details of Solved Issues

Java SDK

As mentioned in Section 2.2.1, Java SDK 1.4 is requested for IPv6 support. GT3 uses the xml-security package, which Java SDK 1.4 has an unstable version of included. The Xalan jar needs to be replaced by a stable version (v 2.2.0 or later). Java SDK 1.4 cannot provide the right IPv6 reverse lookup service due to a bug in the code. This has already been fixed in Java SDK 1.5. We have tested GT3 with Java SDK 1.5 beta. Two incompatible problems [Globus Bugzilla #1410 and #1531] between GT3 and JDK 1.5 have now been solved.

GT3 Version Dependency

The GT3 alpha version became available at the beginning of 2003. Since then, we mainly worked on the GT3 alpha version because it provided the most verbose debug information. We have demonstrated successfully IPv6 functionality by running our tests on the GT3 alpha version. We started to move to release GT 3.0 when it came out at the end of June 2003. We have demonstrated the GT 3.0 (both 3.0.1 and 3.0.2) core working with IPv6 with minimal modification. There are more components involved in GT 3.0 and so they were surveyed for IP dependencies. Some of the GT 3.0 components, such as the Globus Resource Allocation Manager (GRAM), have been identified as having a higher degree of IP-dependent coding. We worked with Globus implementation group to modify the source code to be IP dependent and this is now included in official releases. We are now using the latest cvs installation, which provides IPv6 support and no modification is required. For full IPv6 functionality, Java SDK 1.5 is required. We have successfully demonstrated this functionality on the GT3.2 release for GT3 core, OGSA service browser and GRAM. We also successfully tested the IPv6 functionality in GT3.3 (developer release only).

PostgreSQL

As mentioned in Section 2.2.3, Postgresql is not IPv6 enabled by default. An IPv6 patch from <http://www.lbsd.net> needs to be applied to the source code. By doing this, the database becomes accessible through an IPv6 interface.

The IPv6 functionality has been successfully tested using the *psql* utility. This version of postgresql functions with GT3, which interacts with the MMJFS and RFT, since Globus [WHAT??] is not fully functional over ipv6 it has not been tested with Globus IPv6 job submission. The URL configuration option for invoking JDBC jobManagerDb database has been identified.

Tomcat

As mentioned in Section 2.3.1, using Java SDK 1.4 for IPv6 support with the “full” edition of Tomcat cannot work correctly, as Tomcat duplicates some of libraries and thus conflicts with Java SDK 1.4. The “lightweight edition” of Tomcat has to be used instead. Additionally, extra configuration is necessary in order for Tomcat to respond correctly to a client request with literal IPv6 addressing.

We have tested Jakarta Tomcat on IPv6. In Redhat, Tomcat starts a web service, which listens on an IPv6 port. Tomcat has been successfully accessed using an IPv6 enabled web browser with both IPv4 and IPv6 requests.

There is a coding bug resulting in Tomcat not responding correctly to IPv6 address calls. It has been identified as the well-known ‘IPv6 address handling problem’, which adds square brackets to literal IPv6 addresses when used in a URL.

We later tested Tomcat 5 (version 5.0.12) in November 2003. The mentioned bug has been fixed. It supports IPv6 more fully now.

Survey All GT 3.0 Components for IP Dependencies

More components are involved since GT 3.0. Some of GT 3.0 components, such as GRAM, have been changed a lot since GT3-alpha. And there are a few components we did not involve within our earlier testing and porting. We surveyed them. We went through all their source code for checking IP dependencies. The IP dependencies were reported to Globus implementation group through Globus Bugzilla while we solved them with our modification. Globus implementation group modifies their source code as well and includes them into the latest cvs and their future official release.

Gatekeeper Configuration

We have identified three configuration options involved to access gatekeeper: gatekeeperOption, gatekeeperHost and gatekeeperPort. Gatekeeper can be potentially configured to work over IPv6.

However, Globus implementation group has determined that the gatekeeper gateway prototype is not used in GT3 distribution any longer.

Hard-Coded IP Addresses

In the GT3 source code (alpha version, 3.0.1 and 3.0.2), there are 10 Java source files that include at least one IPv4 loopback address (127.0.0.1), listed as the following. They need to be modified to be IP version independent: replaced by hostname "localhost" or local hostname, then using Java IP-Related System Properties to choose which IP are used. By using "localhost", two entries, one links "localhost" with an IPv4 loopback address, another links "localhost" with an IPv6 loopback address, should be configured in */etc/hosts*. UCL reported the these 6 files to Globus implementation group through the Globus Bugzilla (#1035). Globus implementation group has changed them in their GT3.2.

- HostingEnvStarter.java
- HostStarter.java
- WebSphereTest.java
- EJBServletClient.java
- ServiceURL.java
- TestServer.Java
- OpenJMSAdapter.java
- ManagedJobImpl.java
- UHEActivityTask.java
- UserHostMap.java

Invoking JDBC

Postgresql is listening on an IPv6 port after the IPv6 patch is deployed. UCL had identified the dbConnectionURL configuration option, which is in sever-config.wsdd, as the one that configure database accessing reference.

GT3 Container Configuration for IPv6

The Grid Service Container is an abstract OGSF run-time environment. We identified two configuration options for our IPv6 testing: logicalHost and publish HostName. By setting these two options, we successfully made Grid Service Container working with IPv6 DNS name. IPv6-enabled stand-alone Web container and client container communicate each other in pure IPv6.

IPv4 and IPv6 Co-exist Service

Since there will be a period of IP transition, considerations must be given to an interim coexistence of IPv4 and IPv6. It is straight that IPv4-only machines communicate only in IPv4 and IPv6-only machines communicate only in IPv6. However, the situation becomes much complicated on the dual IP stack machines. The IP version independent services should be provided. The host starts with hostname, which are always IP-independent. The host responds client calls according to which IP family the client uses. When an IPv4 client calls with IPv4 addresses, the Grid server uses IPv4 interface to respond; only IPv4 communication takes place. Similarly, when an IPv6 client calls with IPv6 addresses, the Grid server uses IPv6 interface to respond; only IPv6 communication takes place. For the client from the dual-stack machine, client can choose which IP family is the default or preferred.

Java IP-Related System Properties

On dual stack machines, system properties are provided for setting the preferred protocol stack—IPv4 or IPv6—as well as the preferred address family types—inet4 or inet6.

IPv6 stack is preferred by default, since on a dual-stack machine an IPv6 socket can talk to both IPv4 and IPv6 peers. By default, IPv4 addresses are preferred over IPv6 addresses.

UCL has put in code to show that the Java IP-Related System Properties work. They are particularly useful for IPv4/IPv6 coexisting. The tests using these IP-Related System Properties with Globus work well during the initialisation.

java.net.preferIPv4Stack (default: false)

If IPv6 is available on the operating system, the underlying native socket will be an IPv6 socket. This allows Java applications to connect too, and accept connections from, both IPv4 and IPv6 hosts.

If an application has a preference to use only IPv4 sockets then this property can be set to true. The implication is that the application will not be able to communicate with IPv6 hosts.

java.net.preferIPv6Addresses (default: false)

If IPv6 is available on the operating system, the default preference is to prefer an IPv4-mapped address over an IPv6 address. This is for backward compatibility reasons - for example applications that depend on access to an IPv4 only service or applications that depend on the %d.%d.%d.%d representation of an IP address. This property can be set to try to change the preferences to use IPv6 addresses over IPv4 addresses. This allows applications to be tested and deployed in environments where the application is expected to connect to IPv6 services.

Setup IP Family System Configuration Options

In the command-line based, Java IPv6-Related System Properties can be set up with the following format:

`-D<name>=<value>`

In the programming, Java IPv6-Related System Properties can be setup with the following format:

```
System.setProperty ("propertyname", "value")
```

Java CoG Kit IPv6-enabled Porting

As mentioned in Section 2.2.6, the Globus Java CoG Kit is designed in the Java language. IPv6 underlying support can come by using Java SDK1.4. So far, Globus Java CoG Kit has preformed successfully in IPv6 testing.

UCL and ANL have gone through the source code of Globus Java CoG Kit for IP dependencies checks [See Globus Bugzilla #1034]. All those found pieces of code are now fixed.

Literal IPv6 Address Handling in Java

So far, the majority of the problems we have encountered in trying to run under IPv6 are with the literal IPv6 addresses in URLs [RFC2732]. In the Java implementation, this could be solved by using the “right way” (“new URL”) to generate URLs everywhere. In implementation with JDK 1.4, the following operations are automatically completed:

- There are no differences for IPv4 addresses, hostname;
- For IPv6 addresses, it checks the square brackets;
 - If there are no square brackets, it put brackets in;
 - url.gethost functions output literal IPv6 addresses with the square brackets;
 - As input parameters, the address with square brackets are acceptable for URL and InetAddress functions.

Literal IPv6 Address Handling Implementation in GT3

A Uniform Resource Identifier (URI) [RFC2396] is a compact string of characters for identifying an abstract or physical resource. But literal IPv6 addresses [RFC2732] are still problematic in some particular practical implementation. UCL has met such problem in GT3 and IPv6 testing. That is nature since the original GT3 implementation did not give any IPv6 consideration. While some IPv6 host may have no IPv6 DNS name, it is necessary to include literal IPv6 address dealing in UCL’s porting coding.

In the Java implementation, “new URL” should be used whenever generate URLs. GT3 mostly uses strings to generate URLs (See Globus Bugzilla #1361). We have identified 19 code points within 14 files, which should be modified. We have modified 13 code points within 8 files during our porting for GT 3.0.1 and GT 3.0.2. In order to keep the minimum modification, our solution is that we use standard Java “new URL” to generate URLs, then later use URL.toString() to change URLs back to string style, which matches the way that parameters are passed between the Globus functions. ANL has also fixed most of them and included them in the latest cvs source code and GT3.2 release.

Java Reverse Lookup Bug for GRAM in GT 3.0

UCL has demonstrated full IPv6 functionalities on GT3-alpha (GT3-a) for GRAM. However, GRAM has been changed since GT3-alpha. Problems were encountered running GRAM, even with IPv4, since GT 3.0. A few other people have met the same problem and reported. We followed Globus Bugzilla #1140 and solved that problem by switch JVM from IBM JVM, which distributed with GT 3.0 to sun JVM.

The GRAM in GT3.0 is different from GT3-alpha. More IP dependencies have been identified within it. However, our working was being held by a Java reverse lookup bug in Java SDK 1.4. In Java 1.4.0, IP addresses (both IPv4 and IPv6) are acceptable as legal hostname. Therefore, when an IP address is given, the above Java function are not look up hostname. It works with Java 1.3. This

bug is partly fixed in Java 1.4.2: IPv4 address would be translated to hostname. But it is still problem for looking up hostname with IPv6 address.

As mention in Section 2.2.1, Java SDK 1.5 has become available January 2004. It has fixed the IPv6 reverse lookup bug in Java SDK 1.4. With Java SDK 1.5, we have already demonstrated full IPv6 functionalities on latest GT3.2 for GRAM.

Investigating Other Web Service Container

We have got the reasonable success on the stand-alone Web service container (GT3-alpha so far). We also investigated these issues of deploying GT3 on other web service containers. As recommend of Globus implementation group, Jakarta Tomcat is our first investigate target. As mentioned earlier (Section 2.3.1), there is a coding bug that Tomcat 4 cannot respond correctly with IPv6 address calls. We tested Tomcat 5 (version 5.0.12) individually November 2003. It provides the fully support for IPv6. UoS were investigating to deploy IPv6-enabled GT3 core services on IBM Websphere and had got a reasonable success.

URL Port Check Functions

When a default port is used (no particular port given in URL or command arguments), the port is displayed incorrect ("-1"). The program should get the default port according to the protocols. In order to do so, a check functions should be applied after `URL.getPort()` function. If `URL.getPort()` function return value ("-1"), use `URL.getDefaultPort()` to get the default port. However, this is not an IPv6-associated issue. But it is a network-associated issue.

GRAM Client Configuration

GRAM client uses different configuration file - *client-server-config.wsdd*. To config GRAM client to use IPv6 address or hostname, *logicalHost* option must be included in the *globalConfiguration* section.

Heterogeneous IPv4/IPv6 Environment

For the reasons mentioned in Section 3.4.3, our effort to porting Globus IPv6-enabled takes an IP-protocol independent approach, i.e. it supports both IPv4 and IPv6. We have already got the Globus server and client both working on the dual-stack host. We tested the dual-stack server accessed by both IPv4-only and IPv6-only client successfully. We also tested the dual-stack client to access both IPv4-only and IPv6-only server successfully.

For communication in heterogeneous IPv4/IPv6 networks, there are a number of network transition aids, which essentially translate the packet headers between IPv4 and IPv6, leaving the payload untouched. We used a NAT-PT (Network Address Translator – Protocol Translator) gateway in my experiments. It also involved a ALG-DNS (Application Level Gateway – Domain Name Server) as well.

Appendix B: IPv6 Relevant Bugs

We list the IPv6 relevant bugs as following. They reference the Bugzilla numbers. We also give a brief description and status for each bug.

- Globus Bugzilla #1032, Porting GT3 IPv6-enabled - Solved Issues 1 (JDK 1.4, JDBC IPv6, Tomcat edition and GT3 Container Configuration)

[Resolved] It is about to get the IPv6-enabled environment for GT3.

- Globus Bugzilla #1033, Porting GT3 IPv6-enabled - Identified Issues 1 (Invoking JDBC and Gatekeeper Configuration)

[Resolved] It is the configuration for GT3 to use IPv6 APIs.

- Globus Bugzilla #1034, Porting GT3 IPv6-enabled – Identified Issues 2 (CoG)

[Resolved] Java Commodity Grid Kit provides access to Grid services. GT3 needs the IPv6 support from it.

- Globus Bugzilla #1035, Porting GT3 IPv6-enabled – Identified Issues 3 (Hard-coded IP Addresses and literal IPv6 address handling)

[Resolved] About ten Java files have hard-coded IPv4 addresses. It has been marked for GT 3.2. For literal IPv6 address handling, see Globus Bugzilla #1361.

- Globus Bugzilla #1256, cannot submit job to localhost

[Resolved] Since GT 3.0.1 the localhost is not a valid target to submit job. For submitting job through IPv6 interface, we meet the Java reverse lookup problem, see below.

- Globus Bugzilla #1342, IPv6 non-compatible Axis library in GT3 - and fixed axis

[Resolved] The Globus-shipped axis library (axis.jar) has a few IPv4 dependencies. We has helped Axis fixed it, see axis-Apach Bugzilla #23576. It has been marked for GT 3.2.

- Globus Bugzilla #1361, IPv6 non-compatible URIs and URLs generating

[Resolved] The source code needs to be changed to meet the particular literal IPv6 address format in URIs and URIs.

- Axis-Apach Bugzilla #23576, IPv4 dependencies in AXIS - org.apache.axis.types.URI

[Completed] URI functions were IPv4-only. This has been fixed to be IP independent.

- Report IPv6 reverse look up bug to Sun Java

The reverse lookup function does not work properly in Java 1.4. In Java 1.4.0 and 1.4.1, both IPv4 and IPv6 reverse lookup do not work. Java 1.4.2 fixed the IPv4 reverse lookup, but the IPv6 reverse lookup does not work yet. The bug has been fixed in JDK 1.5

Appendix C: Non-IPv6 Relevant Bugs

We met a number of non-IPv6 relevant bugs when we were porting GT3 to be IPv6-enabled. We list them as following. They reference the Bugzilla numbers. We also give a brief description and status for each bug.

- Globus Bugzilla #1140, UserContainer loops continuously during initialisation (JVM)

[Resolved] GT3 was hard-coded to use shipped IBM JVM. It causes problems on some machines to use MMJFS.

- Globus Bugzilla #1246, cannot resolve symbol - when compiling mmjfs from cvs

[Resolved] Some of these cvs build.xml do not consider different configuration well.

- Globus Bugzilla #1410, local part cannot be "null" when creating a QName

[Resolved] Globus uses JAX-RPC library, which has bug to check the validation of QName. GT3 has problems causing by the invalid QName with the after-fixed JAX-RPC library.

- Globus Bugzilla #1531, no such algorithm: MD5/RSA for provider Cryptix

[Resolved] One of the un-compatible problem between GT3 and Java 1.5.

Appendix D: Online Guide Documentation “How-to IPv6 in Globus Toolkit 3”

This documentation has become part of the official Globus technology reference documentations.

How-to IPv6 in Globus Toolkit 3

Sheng JIANG

s.jiang@cs.ucl.ac.uk

University College London

Status of this documentation

This documentation is for people who have known Globus already and want to develop IPv6 support with Globus. The normal installation and configuration of Globus are not introduced in this

document through we give a brief description of the cvs installation as an appendix. The differences between these three Globus version (3.2, latest cvs, 3.0.2, 3.0.1 and alpha) are described individually. For the full IPv6 support, we recommend GT 3.2 or the installation from the latest cvs, which has included our IPv6 modification. In the documentation, we try to give as general as possible considerations for IPv6 deployment. Another principle in our working is to keep our modification as little as possible while the Globus Toolkit is developing very quickly. We have identified a few further modifications, which will make IPv6 configuration and operation earlier and smoother. They will be implemented on the reasonable stabled Globus Toolkit distribution later. We will keep tracking the major official release of Globus Toolkit and giving the correspondent guide in the future.

1. Operating System Support on Hosts

First of all, the hosts should be IPv6-enabled. The operating system needs to provide IPv6 support. Type the following command to see whether the IPv6 module is loaded on your machine or not.

```
# lsmod | grep ipv6
```

If the IPv6 module is loaded, you could find your global IPv6 address by using "*ifconfig*" command. As long as you have IPv6 modules loaded on your hosts, your IPv6 tests could be run locally.

On Linux Red Hat 8 or later, IPv6 module is provided and auto-load as default.

On Linux Red Hat 7.x, IPv6 is support by kernel, which IPv6 module is not load as default.

On Linux Red Hat 6.2, users have to re-compile the kernel to get the IPv6 support. (Reference to <http://www.bieringer.de/linux/IPv6/>)

IPv6-capable application API libraries need to provide support for upper-layer applications, such as java. JDK 1.4 (see later Section 3.1) or later provides the IPv6 support on Solaris, Unix and Linux while JDK 1.5 (see later Section 3.2) provides IPv6 for WinXP and Win2003 server.

2. Networking Support for IPv6

To start any IPv6 experiment, we need the IPv6 support from the platform. First we need the IPv6-enabled networking. It requires the IPv6-enabled routers, which provide forwarding and dynamic routing, and support from IPv6-enabled network services, such as IPv6 DNS, Web services, etc.

Be sure your hosts can communicate with each other using IPv6 interfaces. On IPv6-enabled machines, use "ping6" command followed by destination IPv6 address or IPv6-enabled hostname to check the IPv6 communication.

```
# ping6 3ffe:2101:7:4:2e0:18ff:fe34:150b          or
# ping6 mocha.ip6.cs.ucl.ac.uk
```

If there is no IPv6 DNS server, the hostname may fail. (You could configure your local hostname lookup in */etc/hosts* configuration file.)

3. Associated Applications

The Globus system also utilises extend applications. These applications need to be IPv6-enabled as well.

3.1. Java SDK 1.4

In GT3, Java run-time environment need to be IPv6-enabled as mentioned earlier. Java SDK 1.4, which has the IPv6 support, has been tested to work but requires some extra security configuration. The security problem appears to arise because Sun packaged a beta of Xalan with Java SDK 1.4, but the xml-security package requires a stable version of Xalan (v 2.2.0 or later). To fix the problem, the stable version of xalan.jar needs to be installed into the \$JAVA_HOME/jre/lib/endorsed directory.

The latest Java distribute can be download from Sun website (<http://java.sun.com>). Follow the instructions for installation, set JAVA_HOME to the installation directory, and put \$JAVA_HOME/bin on your PATH. Make sure java can be found as a command.

You can use JRE if you do not need to build any source.

3.2. Java SDK 1.5

Java SDK 1.5beta was released in January 2004 with fuller IPv6 supports. The IPv6 reverse lookup bug in JDK 1.4 has been fixed. The IPv6 reverse lookup functionality is necessary for GRAM of GT3. Therefore, JDK 1.5 is requested for all GT3 users, who need GRAM. Since the earlier release, GT 3.0 is not compatible with JDK 1.5.

As a conclusion, for IPv6-enabled GRAM, please use the latest cvs installation with JDK 1.5.

We have a small test program for the IPv6 reverse lookup functionality ([download link](#)):

```
# java lookupAndReturn -6 hostname
```

If you have IPv6 address displayed and the reverse lookup hostname is not the address, your system is ready for the IPv6-enabled GRAM.

3.3. PostgreSQL (JDBC)

A JDBC compliant database is required for use of the Reliable Transfer Service (RFT) and Master Managed Job Factory Service (MMJFS). Currently PostgreSQL is recommended. For IPv6 support, an IPv6 patch needs to be applied. It can be downloaded from <http://www.lbsd.net/>. Download the patch, apply to PostgreSQL source code, then build and install PostgreSQL from the source code.

3.4. Web Containers

A Web container is the execution infrastructure for operating OGSA services. The container environments need to provide IPv6 Web services for OGSA.

GT3 provides a stand-alone web container. But it is only for test purpose. Jakarta Tomcat has been recommended by the Globus implementation group. A few other web service containers may be used as well. The configuration of stand-alone web container is introduced in Section 4.1.

Tomcat5 is recommended with fully IPv6 support. For Tomcat4, since we use JDK 1.4 for IPv6 support, the "lightweight edition" of Tomcat4, which just excludes several libraries that are already included in the JDK1.4 distribution, has to be used instead of the "full edition" of Tomcat4. However, there still be problem for Tomcat4 respond a literal IPv6 address call. Tomcat could download from apache website <http://jakarta.apache.org/tomcat/>.

For deploying IPv6-enabled GT3 on Websphere, please reference:

<http://www.ecs.soton.ac.uk/~dgm/work/work.php>

4. IPv6 Configuration of GT3

For the general installation and configuration of GT3, please follow the Globus admin guide (<http://www-unix.globus.org/toolkit/3.0/ogsa/docs/admin/>). We only describe the particular steps for IPv6.

4.1 Stand-alone Web container configuration

By default, the Globus stand-alone Web container binds with the IPv4 address of local host. However, the webcache port is connectable through both IPv4 and IPv6 interface. But in IPv6 calls, Globus server side will translate the IPv6 communication into IPv4 since the port is binding with IPv4 address of local host.

To use hostname instead of IP address, set up the configuration option "publishHostName" be "true" in the globalConfiguration section of server-config.wsdd as following:

```
<parameter name="publishHostName" value="true"/>
```

Then, the Globus stand-alone Web container looks up the hostname of local host and uses it in the initial processing. On the dual stack machine, the hostname could bind to both IPv4 and IPv6 address. Then the Globus server is IP independent and could communicate with client according to the IP family that client uses.

If there are other hostnames could link to the localhost, such as a particular IPv6 hostname, you could set up the configuration option "logicalHost" in the globalConfiguration section of server-config.wsdd as following:

```
<parameter name="logicalHost" value="mocha.ip6.cs.ucl.ac.uk"/>
```

In the above example, *mocha.ip6* is a hostname that binds only IPv6 address. Then Globus stand-alone Web container will bind with IPv6. The IPv4 calls are still acceptable on dual stack machines. But Globus server side will translate the IPv4 communication into IPv6.

In the original Globus coding, literal IPv6 address is not acceptable. To use a literal IPv6 address in a URL [See RFC 2732], the literal address should be enclosed in "[" and "]" characters. To use literal IPv6 address, see Section 4.3.

Notice: The client side shares the same configuration file with the server side. If you are going to run client and server on different machines, the above configuration needs to be done on both client and server sides.

4.2. PostgreSQL IPv6 configuration

As mentioned in Section 3.2, an IPv6 patch needs to be applied with PostgreSQL source code. After that, you need to make sure that postmaster will be started with the "-i" flag to allow TCP/IP based connections. This will be in /etc/init.d/postgresql. If your postgresql service script uses pg_ctl to start postmaster, use "-o 'i'" to pass argument to postmaster. Secondly, if you want to allow remote hosts to connect to your DB, you will need to edit pg_hba.conf. By default, this will only allow connections from 127.0.0.1. Put IPv6 loopback address in as a new entry with IP-Mask `ffff:ffff:ffff:ffff:ffff:ffff` as following:

```
host all all ::1 ffff:ffff:ffff:ffff:ffff:ffff trust
```

The invocation of JDBC database from GT3 is configured in the *server-config.wsdd*. The JDBC databases are different in GT3 alpha and in GT3.0.

In GT3 alpha, JDBC databases are used for both ManageJobFactoryService and ReliableTransferFactoryService. The URL connection in both services should be changed as following (the entries in local-server-config.wsdd and altered-server-config are used in some particular conditions):

```
<service name="base/gram/ManagedJobFactoryService" provider="Handler" style="wrapped"
use="literal">
<parameter name="dbConnectionURL" value="jdbc:postgresql://[::1]/jobManagerDb"/>

<service name="base/reliabletransfer/ReliableTransferFactoryService" provider="Handler"
style="wrapped" use="literal">
<parameter name="connectionURL" value="jdbc:postgresql://[::1]/testDatabase"/>
```

In GT3.0, GRAM starts to use embedded database Xindice, while RFC renamed after MultiFileRFTFactoryService. The URL connection should be changed as following:

```
<service name="base/multirft/MultiFileRFTFactoryService" provider="Handler"
style="wrapped" use="literal">
<parameter name="connectionURL" value="jdbc:postgresql://[::1]/rftDb"/>
```

4.3. Special for literal IPv6 dealing

As mentioned earlier, in the original Globus coding, a literal IPv6 address is not acceptable. This is mainly caused by using String to create URLs. In order to compatible with literal IPv6 address, Java standard method - "new URL" - should be used whenever create a URL. A few Java files have been identified and modified in our porting. Globus 3.0 invokes a shipped axis library (apache project) to create SOAP scheme, and that axis does not support IPv6. Therefore, we modified xis (only the piece of code that Globus invokes) as well. The axis library file is provided besides our modified ogsa.jar. The latest axis has fixed the problem. Globus has already included the fixed axis in their cvs and GT 3.2.

IPv6 modification has already included in GT 3.2 and the latest cvs. To get our modification in order to enable literal IPv6 address usage in earlier version of GT3, download our modified ogsa.jar (and axis.jar) from the following URL. Replace the existing ogsa.jar (and axis.jar) in \$OGSA-DIR/lib.

<http://www.cs.ucl.ac.uk/staff/sjiang/webpage/GT3-IPv6-Download.htm>

After that, you could use the global IPv6 address with the configuration option "logicalHost" in the globalConfiguration section of server-config.wsdd as following:

```
<parameter name="logicalHost" value="2001:630:13:101:2e0:18ff:fe34:150b"/>
```

Then Globus stand-alone Web container will bind with IPv6. The IPv4 calls are still acceptable on dual stack machines. But Globus server side will translate the IPv4 communication into IPv6.

4.4. Tomcat Configuration for IPv6

The server side of GT3 could be deployed on other Web containers. The deployment processing generates the configuration file from the temp, not the configuration file. Therefore, you have to repeat what we mentioned in Sections 4.1 & 4.2.

Tomcat version 5 has been tested with fully IPv6 support. Tomcat4 has problems to correctly respond the call with a literal IPv6 address.

5. Running Tests with IPv6

If the server side of Globus binds to literal IPv6 address or IPv6-only hostname, no matter which IP family client use, the communication will be through IPv6 interface. In the following tests, the client's choice of IP family is based on the fundamental concept that the server side binds to the hostname that has both IPv4 and IPv6 address.

5.1. Start Web Container

To start Globus stand-alone Web container, run "ant startContainer" at the Globus installed directory. To start the Tomcat, run "service tomcat5 start" ("service tomcat4 start" for Tomcat4) as root for the rpm installation or run "source bin/startup.sh" at the Tomcat installed directory.

5.2. Running GUI (OGSA Service Browser)

To start OGSA Service Browser, type the following ant command. It will start with the local host and the default port 8080.

```
# ant gui
```

In the graphic browser, you could change the host and port in order to browse the OGSA services on remote machines or on other web container. Change *service.root* and *service.port* in *ogsa.properties* to configure the default host and port.

5.3. Running GRAM

To run the GRAM job on remote machines or on other web container, use the following command according to your GT version. Replace the "localhost" by you remote hostname and replace port "8080" by your actually port number. Notice: the GT3 alpha and GT3.0 cannot communicate with each other.

GT 3.2:

```
# java org.globus.ogsa.impl.base.gram.client.GlobusRun -factory  
http://localhost:8080/ogsa/services/base/gram/MasterForkManagedJobFactoryService -file  
schema/base/gram/examples/test.xml
```

GT 3.0, GT 3.0.1 and cvs installation:

```
# java org.globus.ogsa.impl.base.gram.client.GlobusRun -factory  
http://localhost:8080/ogsa/services/base/gram/MasterForkManagedJobFactoryService -file  
etc/test.xml
```

GT 3 Alpha:

```
# java org.globus.ogsa.impl.base.gram.client.GramClient  
http://localhost:8080/ogsa/services/base/gram/MasterManagedJobFactoryService etc/test.xml
```

5.4. To choose the IP family on client

To choose the client IP family, run the above Java command with the following arguments:

```
-Djava.net.preferIPv6Addresses=true
```

- Use IPv6 addresses if it is possible

-Djava.net.preferIPv4Stack=true

- Use IPv4-only Stack on client side

6. Further modification

While keeping the minimum modification during the testing and experiments, we have identified a few further modifications, which will make IPv6 configuration and operation earlier and smoother. They will be implemented on the reasonable stable Globus Toolkit distribution later. They include:

1. Literal IPv6 address check for URL and dealing function
2. Change the default value of publishHostName
3. Set up the default IP family configuration option
4. Integrate IP choose configuration option into GUI
5. Integrate IP choose configuration option into GRAM
6. Modify the port check functions for URL display
7. Modify ant commands and binary client commands

7. The installation from the latest cvs

Check out the packaging through cvs:

```
# cvs -d :pserver:anonymous@cvs.globus.org:/home/globdev/CVS/gridservices login
```

```
# cvs -d :pserver:anonymous@cvs.globus.org:/home/globdev/CVS/gridservices co packaging
```

Under packaging directory, run install script with anonymous:

```
# ./make-packages.pl --install=<install-dir> --anonymous --verbose
```

Or run helper install script to install without scheduler bundles packages:

```
# ./helper.sh --anonymous --verbose
```

Appendix E: Proposal for a Reusable Grid Service Demonstration Component

1. Concept

This section proposes the development of a Grid Service using a recent version of the **Globus Toolkit** (GT3). The service is intended to demonstrate the value and feasibility of the GT3 in grid applications that may be developed by IBM and its partners. As stated in the GT3 whitepaper at <http://www-unix.globus.org/toolkit/3.0/ogsa/docs/>:

The core infrastructure of Globus Toolkit 3 (GT3 Core) is based on the Open Grid Services Infrastructure (OGSI) primitives and protocols. The main design goal has

been to make the OGSi technology easy to use, reuse, and extend when developing new Grid applications.

Since GT3 is the lowest implementation layer of **OGSI**, this document proposes to develop a module which would not only demonstrate GT3's infrastructure, but also provide a useful and reusable Grid Service function.

2. Application

According to globus.org, GT3-style Grid Services are being studied and applied largely for the following interrelated areas:

- ⇒ Science Portals
- ⇒ Distributed Computing
- ⇒ Large-Scale Data Analysis
- ⇒ Computer-in-the-Loop Instrumentation
- ⇒ Collaborative Work

All of these application areas share functional requirements that are on a higher level than those offered by components implemented by GT3. In particular, Grid applications of this nature need to be able to do what we might call “**collective machine capacity assessment**”.

Put simply, this means that a Grid Services “client” needs to know how many physical machines or nodes may be connected to a Grid and what their computational abilities and/or storage capacities represent at a given time. To an extent, the capability of a Grid Service is the sum of its components. **It would therefore be advantageous to develop a service which could poll nodes on a Grid and aggregate data concerning their characteristics.**

Such a Grid Service would provide information such as:

- ⇒ CPU characteristics
- ⇒ Number of CPUs
- ⇒ Grid Container Memory (used and free)
- ⇒ Hard Disk volume storage available

All of this information is accessible through the Java *Runtime* object and Java System properties hosted by the Globus container.

3. Architecture

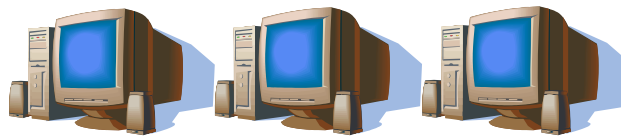
The Grid Service for machine capacity reporting would be invoked in five steps:



1. Globus Client queries Grid Service Reporting Node



2. Reporting Grid Service locates known Grid Work Nodes



3. Reporting Grid Service queries Work Nodes



4. Work Nodes provide capacity information



5. Response forwarded to Client

The Grid Service would be developed using Websphere Studio Application Developer version 5.1 deployed to Websphere Application Server, also version 5.1 – which is compatible with IPV6 addressing. Here are the anticipated design approaches:

- **Implementation approach:** inheriting from a Grid Service skeleton class (as opposed to a delegation model, with operation providers)
- **Lifecycle management:** configurable for persistent or transient.
- **Service Data:** Service data is used to index nodes according to their characteristics and capabilities.
- **Notifications (optional):** if a change occurs in a node, that change is propagated to all subscribers.

This design is estimated to require 200 hours of development, not including integration testing.