DataGrid

GRID NETWORK MONITORING

DEMONSTRATION OF ENHANCED MONITORING TOOLS

Document identifier:	**DataGrid-07-D7.2-0110-4-1**
Date:	**21 January, 2002**
Work package:	**WP7**
Partner(s):	**CNRS – INRIA - PPARC – CERN – INFN - SARA**
Lead Partner:	**PPARC**
Document status	**Final Draft**
Deliverable identifier:	**D7.2**

Abstract: This document reports the progress made by WP7 in defining an initial architecture for network monitoring in the GRID. It outlines the requirements for network monitoring. The classical metrics associated with network monitoring are described and existing monitoring methods and associated tools are catalogued. An initial network monitoring distribution was made in PM9 and is here, at PM12, extended in capability; and this new distribution is described.

Delivery Slip

	Name	Partner	Date	Signature
From				
Verified by				
Approved by				

Document Log

Issue	Date	Comment	Author
1_0	5-12-2001	First draft	Robin Tasker
2_0	10-12-2001	Second draft	Robin Tasker
3_0	17-12-2001	Third Draft	Robin Tasker
4_0	20-12-2001	Fourth Draft	Robin Tasker
5_0	9-1-2002	Fifth Draft	Robin Tasker
6_0	11-1-2002	Final Draft	F Bonnassieux, P Mealor, P Primet and R Tasker
6_1	14-1-2002	Final Draft	F Bonnassieux, P Mealor, P Primet and R Tasker

Document Change Record

Issue	Item	Reason for Change
1_0	First outline document	
2_0	Inclusion of LDAP and publication detail	Adding necessary content
3_0	Addition of MapCenter and UDPmon details; NM architecture diagram	Adding more detail
4_0	Additional material plus editorial comments	RTPL details added, textual editorials + update to Figure 2
5_0	Additional material plus editorial comments	Terminology and Glossary added, Franck's comments included and Tiziana's suggestion for NetSaint.

6_0	Additional material plus editorial comments	Incorporation of comments from Pascale and Jules, final tidying up of document.
6_1	Modified Exec.Summary plus minor editorials	Incorporates comments from Pascale.

Files

Software Products	User files
Word	WP7-D7-2-0110-4-1.doc

CONTENT

1. EXECUTIVE SUMMARY

This document reports the progress made by WP7 in defining an initial architecture for network monitoring in the Grid. Measuring and monitoring of network performance is required for two important reasons. The first is to provide the tools necessary to view the network performance from a Grid applications standpoint and hence identify any strategic issues which may arise (such as bottlenecks, points of unreliability, Quality of Service needs). The second is to provide the metrics required for use by Grid resource broker services.

An initial network monitoring distribution was made available in PM9 and is here, at PM12, extended in capability.The object for the PM12 deliverable was to provide a prototype network monitoring toolkit within the framework of a simple and extensible architecture that permitted basic network metrics to be published to the Grid middleware, and also for them to be available, via visualisation, to the human observer.

The aim was to make use of well-understood basic network monitoring tools in a coherent and simply extensible manner to demonstrate the publication of network metrics to the Grid middleware via LDAP, and also to make this information available via a Web interface.

This document outlines the requirements for network monitoring. The classical metrics associated with network monitoring are then described and existing monitoring methods and associated tools are catalogued. The architectural design of the monitoring system is presented. It comprises four functional units, namely, monitoring tools or sensors; a repository for collected data; the means of analysis of that data to generate network metrics; and the means to access and to use the derived metrics.

The WP7 Grid network monitoring system uses well-known tools to gather statistics on well-known metrics but also new specific software , like UDPmon and MapCenter, that have been developed within the WP7 is described.

Basic services are provided and enhancements will be made after this.

The system implements basic monitoring tools realising the standard measurement: Round Trip Delay, Packet Loss, Total traffic volume,TCP and UDP throughput, site connectivity, service availability.

In respect of network information services, WP7 has developed an LDAP back end to make measurements visible through the GIIS/GRIS system. Grid applications are able to access network monitoring metrics via LDAP services according to a defined LDAP schema. The LDAP service itself gathers and maintains the network monitoring metric data via backend scripts that fetch the current metric information from the local network monitoring data store. Independently, a set of network monitoring tools are run from the site to collect the data which describes the local view of network access to other sites in the Grid. The separate components that are required by the network monitoring architecture have been developed and demonstrated.

The WP7 network monitoring testbed sites have been used to demonstrate the operation of a variety of network monitoring tools and their ability to collect data and to make the network monitoring metrics available via a Web interface. Separately scripts have been demonstrated that extract the network metrics from the network monitor data store and make them available via an LDAP service. The combined capability was demonstrated in the WP7 PM9 release that has been installed in the DataGrid testbed1. In that release the metrics of round trip time and packet loss have been made available.

WP7 wishes to receive user feedback on its deliverable so that monitoring tools better reflect the needs of the Grid communities. Over the next period WP7 will concentrate effort on the subject of forecasting with respect to the identified network monitoring metrics; will review Grid user requirements in the area; identify available techniques to deliver such requirements; and understand how to incorporate them into the network monitoring architecture it has defined.

2. INTRODUCTION

This document is provided as an accompaniment to the WP7 PM12 deliverable (D7.2), a prototype demonstrating enhanced network monitoring tools. It provides a framework for network monitoring against which WP7 has operated in preparing this prototype. The prototype is designed to demonstrate how basic network monitoring metrics can be made available to the Grid middleware in addition to the visualisation of the metrics via the Web. The architecture upon which this is based is simple and easily extensible. It is in these respects that WP7 views this deliverable to be "enhanced" with respect to the operation of the Grid.

WP7 expects this prototype monitoring tool to be developed further. Specifically WP7 is focusing its work in the area of prediction, of forecasting, network metrics based upon current measurements. WP7 believes that if Grid middleware is to make best use of metric information what is required is a current measurement and a prediction over the "next" time interval. Based upon this work WP7 expects to review its monitoring architecture with a view to accommodating forecasting techniques within the existing scheme, or if necessary to replace what currently exists with a more suitable environment.

WP7 will also work with the other Work Packages to ensure the network metrics from monitoring are usable by the EDG middleware (the resource brokers). In addition, WP7 will review the output from its network monitoring deployed on the Testbed1 and will also accommodate the monitoring data collected by the Grid applications, that is to say, the Grid traffic.

2.1. OBJECTIVES OF THIS DOCUMENT

This document provides a review of the requirements for monitoring with respect to Grid operations, a review of the classic network monitoring metrics and a review of the well known network monitoring tools. It describes the tools that have been selected for this prototype release and how they provide metric information to both the Grid middleware and, via visualisation, to a human observer. Finally this document outlines future areas of work that WP7 believes are required to complete its task.

2.2. APPLICATION AREA

Network monitoring will be used to calculate network metrics that characterise a network. These will be used by the Grid middleware to optimise the performance of Grid applications; they will also be used by network research and developers, and network support personnel to maintain and manage the network upon which the operation of the Grid depends.

2.3. REFERENCE DOCUMENTS

Reference documents

R1	http://www.slac.stanford.edu/comp/net/wan-mon.html
R2	http://www.caida.org/home/
R3	http://www.slac.stanford.edu/comp/net/wan-mon/iepm-cf.html
R4	RFC 2330, Framework for IP Performance Metrics http://www.ietf.cnri.reston.va.us/rfc/rfc2330.txt?number=2330
R5	http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html#variable
R6	RFC 2679, A One-way Delay Metric for IPPM http://www.ietf.cnri.reston.va.us/rfc/rfc2330.txt?number=2679
R7	http://www.slac.stanford.edu/comp/net/wan-mon/resp-jitter.html
R8	http://icfamon.dl.ac.uk/cgi-bin/frequency.pl?sites=ns2.slac.stanford.edu&style=linespoints&days=1

R9 PingER - http://www-iepm.slac.stanford.edu/pinger/
R10 http://www.slac.stanford.edu/comp/net/wan-mon/surveyor-vs-pinger.html
R11 RTPL - http://www.phys.uu.nl/~wwwfi/rtpl/
R12 RIPE NCC Test Traffic Box - http://www.ripe.net/ripencc/mem-services/ttm/index.html
R13 Surveyor - http://www.advanced.org/csg-ippm/
R14 MRTG - http://ee-staff.ethz.ch/~oetiker/webtools/mrtg/
R15 traceping - http://av9.physics.ox.ac.uk:8097/www/traceping_description.html
R16 iperf - http://dast.nlanr.net/Projects/Iperf/
R17 http://www-iepm.slac.stanford.edu/monitoring/bulk/
R18 cflowd - http://www.caida.org/tools/measurement/cflowd/
R19 Network Weather Service - http://nws.cs.utk.edu/
R20 MapCenter - http://ccwp7.in2p3.fr/mapcenter/
R21 A. Martin. Ftree. http://www.gridftp.ac.uk/linux/openldap-ftree.shtml
R22 OpenLDAP. http://www.openldap.org/
R23 The European Datagrid Project. http://www.eu-datagrid.org
R24 P. Mealor, Y. Lee, and P. Clarke. Datagrid network monitoring scheme proposal.
R25 PingER (EDG release). http://ccwp7.in2p3.fr/
R26 NetSaint, http://www.netsaint.org/
R27 Replica Selection in the Globus Data Grid
by Sudharshan Vazhkudai, Steven Tuecke and Ian Foster
R28 NIMI, http://ncne.nlanr.net/nimi/
R29 Creating a Scalable Architecture for Internet Measurement
by Andrew Adams, Jamshid Mahdavi, Matthew Mathis and Vern Paxson
http://www.psc.edu/~mahdavi/nimi_paper/NIMI.html
R30 A network performance tool for Grid environments
by GA Lee, J Stepanek, R Wolski, C Kesselman and I Foster
see http://www.globus.org

2.4. TERMINOLOGY - DEFINITIONS AND GLOSSARY

Definitions

AS	A unit of router policy within the Internet
C	A computer programming language
Globus	See http://www.globus.org
Linux	An open source variant of Unix
Perl	A computer scripting language
RFC	A formal document of the IETF
Unix	A generic computer operating system
VMS	A proprietary computer operating system from Compaq

Glossary

AS	Autonomous System
CE	Computing Element
CGI	Computer Gateway Interface
DA	Destination Address
DIT	Directory Information Tree
DN	Distinguished Name
EDG	European Data Grid
FTP	File Transfer Protocol
GIIS	Grid Index Information Service
GPS	Global Positioning System
Grid FTP	Grid File Transfer Protocol
GRIS	Grid Resource Information Service
HEP	High Energy Physics
HTML	Hypertext Markup Language
ICMP	Internet Control Message Protocol
IETF	Internet Engineering Task Force
LDAP	Lightweight Directory Access Protocol
LDIF	LDAP Data Interchange Format
MDS	Metacomputing Directory Service of Globus
NIMI	National Internet Measurement Infrastructure
NTP	Network Time Protocol
RFC	Request For Comment
SA	Source Address
SE	Storage Element
SNMP	Simple Network Management Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
URL	Universal Resource Locator

3. NETWORK MONITORING IN THE DATAGRID

3.1. INTRODUCTION

Network monitoring will be used in two distinct ways with respect to the Grid and together these describe the aims for network monitoring

Firstly and foremost network monitoring will be used by the Grid applications to optimise their usage of the networks that comprise the Grid. Of primary importance will be the publication of the metrics that describe the current and future behaviour of the network to the Grid middleware such that the Grid applications are able to adjust their behaviour to make best use of this resource.

Secondly, it will be used to provide background measurements of network performance which will be of value to network managers and those tasked with the provision of network services for Grid applications.

A network performance tool for the Grid environment, Gloperf [R30], has been designed as a part of the Globus computing toolkit but is not currently available in the Globus version running in the EDG Testbed1. For that reason, WP7 has developed a network monitoring architecture for the Grid.

3.2. A NETWORK MONITORING ARCHITECTURE FOR DATAGRID

Objectives

The WP7 objective for the PM12 deliverable was to provide a prototype network monitoring set of tools within the framework of a simple and extensible architecture that permitted the basic network metrics to be published to the Grid middleware, and also for them to be available, via visualisation, to the human observer. This accorded with the broad requirements from the other EDG Work Packages.

Overview

Figure 1shows in outline the architecture for network monitoring. It comprises four functional units, namely, monitoring tools or sensors; a repository for collected data; the means of analysis of that data to generate network metrics; and the means to access and to use the derived metrics.

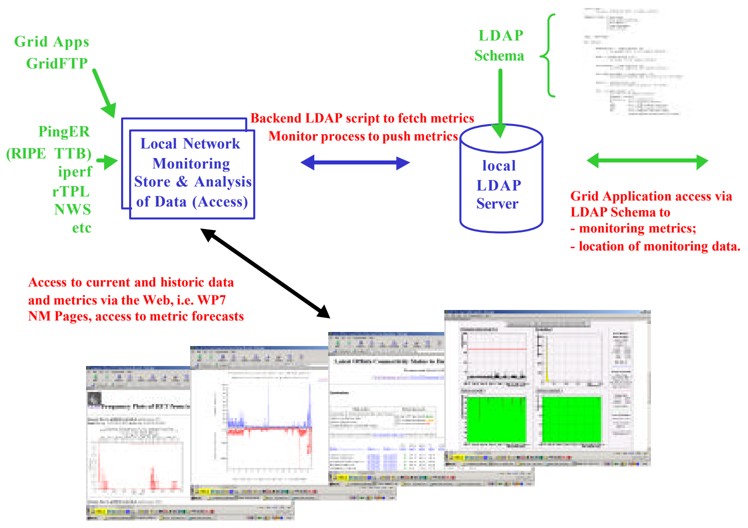

Grid Apps
GridFTP
LDAP Schema
PingER
(RIPE TTB)
iperf
rTPL
NWS
etc
Local Network Monitoring Store & Analysis of Data (Access)
Backend LDAP script to fetch metrics
Monitor process to push metrics
local LDAP Server
Grid Application access via LDAP Schema to
- monitoring metrics;
- location of monitoring data.
Access to current and historic data and metrics via the Web, i.e. WP7 NM Pages, access to metric forecasts

Figure 1 : Architecture for network monitoring in the GRID

Grid applications are able to access network monitoring metrics via LDAP services according to a defined LDAP schema. The LDAP service itself gathers and maintains the network monitoring metric data via backend scripts that fetch, or have pushed, the current metric information from the local network monitoring data store. Independently, a set of network monitoring tools are run from the site to collect the data which describes the local view of network access to other sites in the Grid. This data is stored in the network monitoring data store which is here modelled as a single entity. A set of scripts associated with each monitoring tool is available to provide web based access for viewing and analysing the related network metrics. This architecture allows additional monitoring tools to be easily added with the only requirement being that the means for analysis and visualisation of the data and, either a push mechanism to update local LDAP server, or, a backend script to allow LDAP server access to specific metrics, are provided.

Realisation of the network monitoring architecture

The separate components that are required by the network monitoring architecture have been developed and demonstrated. The WP7 network monitoring testbed sites have been used to demonstrate the operation of a variety of network monitoring tools and their ability to collect data and to make the network monitoring metrics available via a Web interface. Separately scripts have been demonstrated that extract the network metrics from the network monitor data store and make them available via an LDAP service. The combined capability was demonstrated in the WP7 PM9 release that has been installed in the DataGrid testbed1. In that release the metrics of round trip time and packet loss have been made available.

This prototype deliverable extends the PM9 work to include the publication of RTT, packet loss and both TCP and UDP throughput via both LDAP services and the Web. In addition multiple tools have been provided that both measure and report on these same metrics. Within the LDAP schema a "default metric" measurement will be available, and also one specific to a particular monitoring tool. The purpose here is to demonstrate the extensibility of the architecture; to provide an easy means of validating output from a specific tool; and to provide a different "look and feel" to the visualisation process. This architecture is shown in Figure 2 where the relationship between the components of the network monitoring deliverable can be seen.

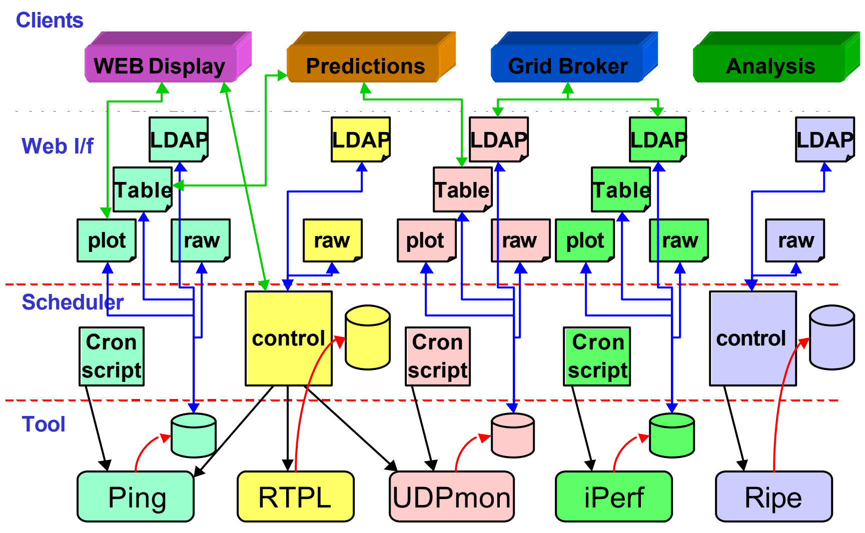


Figure 2 : Diagrammatic representation of network monitoring architecture showing the components and their inter-relationship.

3.3. NETWORK MONITORING

In essence the results of monitoring activities may be separated on the basis of time. The immediate output from monitoring provides a snapshot of existing conditions within the network, whilst the historic data can be used to look back over days, weeks and months at the behaviour of the network. These two uses of network monitoring are for very different purposes. It can be envisaged that the former will be used either directly by an end user wishing to optimise the run of a particular application or, more likely, by the application itself, via the middleware, to adjust, in real-time, its usage of network resources; while the latter will be used to manage the network, to review trends in usage and to ensure sufficient provision exists to meet demand.

Active and Passive Network Monitoring

In order to obtain monitoring information about the network, tests need to be performed and these can be broadly classified into two categories.

Active network monitoring occurs where test data are run directly through the network in order to discover the properties of the end-to-end connection. The traffic generated by such testing is in addition to the usual traffic load on the network. Such an approach makes use of a variety of network monitoring tools and can be appropriately scheduled to minimise the impact to the users of networks whilst still providing an accurate measurement of a particular network metric.

Passive network monitoring makes use of real applications and their traffic to record the application experience of using the network. So for example Grid ftp can be used to record throughput of real Grid traffic across the network and similarly with other applications. This is beneficial in that no additional traffic is introduced across the network but in scheduling will reflect the users' experience in performing some task and as such may not accurately record the capability of the network. In addition any maintenance issues relating to the monitoring aspects of the application are dependent for correction on maintainers of the Grid application itself.

The PM12 deliverable supports only active forms of monitoring where network metric information from the monitoring processes is made available in a form which allows it to be published to the middleware via an LDAP service. WP7 has worked closely with the Gr id ftp developers to ensure that in the future such applications will record the appropriate information and, when available, will fit directly into the architectural model described here.

This initial implementation uses a query/response scheme to request network-monitoring information collated from logs and stored in an LDAP server. This is broadly a "pull mechanism" whereby the monitoring information is stored locally and periodically the LDAP service updates its information by fetching the network monit oring metrics. In the future consideration will also be given to some form of automatic streamed updates through a "push mechanism" where the monitoring tool periodically makes the network metric information available to the LDAP service.

There is concern over the feasibility of using the "push mechanism" in real GRID environments, as servers may be flooded with so much information they have little resources to do anything else. This has been identified as a topic for future work.

3.4. WIDER EXPERIENCES OF NETWORK MONITORING

The subject of network monitoring is well documented and many communities have active programmes in place. Of particular relevance is the work undertaken by the HEP community and within Internet2. The following references [R1] and [R2] are of value and present a number of techniques for monitoring and the visualisation of the resulting data while [R3] provides a detailed comparison between a variety of network monitoring techniques.

4. THE METRICS OF NETWORK MONITORING

4.1. CLASSICAL NETWORK METRICS

Introduction

The IETF IP Performance Monitoring (IPPM) WG have produced RFC 2330 which describes a framework for IP performance metrics [R4] and provides a valuable discussion of the issues relating to network monitoring. A more general discussion of the use of the different metrics described in [R4] can be found in [R5].

Connectivity

Connectivity provides a simple metric on the reachability of a particular end system. The metric may denote the simple presence or absence from the network of an end system, or the presence or absence of a particular service hosted by an end system.

Packet Loss

Packet loss provides a good measure of the quality of the route between end points. However the manner in which applications can deal with such loss will vary greatly depending upon their use of TCP or UDP and the particular requirements of the application.

Round Trip Time (RTT)

The RTT is the time taken to traverse the path from the source to the destination and back. Formally, given a packet p, the time at which the last packet byte departs from the source t(D), and the time at which the last packet byte arrives at the packet destination t(A), RTT = t(A) - t(D)

The RTT is the sum of the propagation time between the end points plus the queuing delays introduced at each hop along the path between the sites. It is therefore characterised on the distance between the end points of the route, the number of router hops on the route and the delay encountered at each hop.

It is a "there and back" measurement which has the benefit that timing measurements are confined to the source, i.e. there is no need for clock synchrony between source and destination. On this basis it is a simple metric.

One-Way Delay

The one-way delay metric has been developed within the IPPM WG of the IETF and is dealt with comprehensively in [R6].

In essence, one-way delay measures the path between source and destination and is the sum of the propagation delays of the data links and the delay introduced at each router hop on the path. One-way delay measurement requires external clock sources (like GPS or NTP - depending on the precision required) for synchronisation and the co-ordination of source and destination processing to make the measurements.

One-way delay measurement has the benefit of providing the measurement of a specific path through the Internet and recognises that asymmetric routing commonly occurs within the Internet. However for an application, the communication between it and the remote client is what matters and regardless of the particular routes taken for the traffic, it is the "there and back" characteristics that matter. Whilst an instance of communication will in all probability be dominated by a flow in one direction or the other, the successful communication relies upon not only the data traffic but also the control traffic. In this respect the "there and back" characteristic is of significant importance.

RTT / One-way Delay relative to a Given Route

The RTT / One-way Delay is dependent upon the route taken across the network and such route variations are likely to introduce differing transit delays. For example, a provider may choose to route traffic on a particular link because of capacity and take no account of distance and associated delay; or because of fault conditions route flapping will have an indeterminate effect of transit delay.

Variation in RTT (frequency distribution of RTT)

A plot of the frequency distribution of RTT measurements can be used to provide a reasonable estimation (the inter-quartile range) of jitter. [R7] and [R8] provide both examples and rationale for this work. The benefit of this approach is that variation in RTT can be readily computed using the data gathered by the simple measurement of RTT.

Jitter

Jitter is the variation in arrival times of successive packets from a source to a destination. It is formally defined by the IETF as the "instantaneous packet delay variation" (IPDV) and is the difference experienced by subsequent packets, I and I+1, on a one-way transit from source to destination. The measurement of one-way delay and derived IPDV provides a means whereby a more rigorous characterisation of the Internet can be developed

Bit Rate and Throughput

Bit rate is a measure of the rate at which binary elements are carried by the network, and throughput is a measure of the effective rate between specific end points across the network.

4.2. GRID NETWORK USAGE METRICS

Volume (number of (Grid) bytes exchanged)

This is the measure of the total number of bytes exchanged per specified Grid activity. Traffic volume estimated for each transaction on a daily/weekly/monthly basis would provide value.

Per Flow Application Throughput

This metric defines the throughput measured for a specific Grid application between specified end-points, i.e. DA/SA/Port. Typically it will be based on the measurement of the actual data transferred during the exchange, i.e. a "passive" measurement, and not on additional (test) data inserted into the network.

Aggregate Network Throughput

This metric defines the aggregated throughput within the network between source and destination end points. This may measure the current utilisation as a rate (volume/time) or as a proportion of the total capability within the path across the network.

5. NETWORK MONITORING TOOLS

5.1. ACTIVE NETWORK MONITORING

Basic Tools

The well known tools - traceroute, pathchar, netperf, etc. - will be used to provide the measurement of the basic metrics.

Many network monitoring tools make use of ICMP, a layer 3 protocol, which may be subject to different traffic management strategies from TCP- or UDP- based traffic (layer 4 protocols). For example, under congestion conditions ICMP traffic will often be preferentially dropped or dealt with at a reduced priority. However work has been published comparing the use of Ping and TCP Syn/Ack packet exchanges to characterise a specified network route. The results were broadly equivalent which suggests that to a first approximation the use of ICMP based tools provide reliable measurements.

PingER

The use within the HEP community of PingER is well established. It is used to measure the response time (round trip time in milli-seconds (ms)), the packet loss percentages, the variability of the response time both short term (time scale of seconds) and longer, and the lack of reachability, i.e. no response for a succession of pings. This is described in detail at [R9].

PingER data is stored locally to allow Web-based access to provide an analysis of packet loss, RTT and the frequency distribution of RTT measurements in both graphical and tabular format. The data is also collected centrally to allow site-by-month or site-by-day history tables for all (or selected) sites as seen from the local monitor point.

The PingER project has a well established infrastructure involving hundreds of sites in many countries all over the world and is particularly focused on the HEP/ESnet communities.

There is clearly a danger that through ICMP rate limiting or ping discard policy that this approach will either give invalid results or no results at all. This is well recognised but to date comparison between PingER and Surveyor and the RIPE NCC TTM box [R10] suggest that the concerns are unfounded. However this offers no guarantees in the future.

RTPL

RTPL (Remote Throughput Ping Load) package is used for periodic network performance measurement tests between a set of locations to view the network performance from a user's perspective. The performance measurements consist of round-trip and throughput measurements between the locations. By default all pairs in the location set are used, but it is also possible to select the location pairs for the tests. In addition, the load on the participating monitoring systems is measured so that any performance change can be related to the particular machine load. Because of the aim to provide the network performance measurement from the user viewpoint and not as the maximum possible capacity of a network, the measurement parameters are configured to default values and the test duration's are limited. Various long-term statistics would otherwise blur the short-time fluctuations.

Measurements are performed by the control host. The control host starts the network performance measurements at each of the participating locations via a secure remote shell command with the results of the measurements being returned using a similar mechanism. The following network performance measurements are made,

> Throughput. As defined in R[4], "The maximum rate at which none of the offered frames are dropped by the device". It is a way to quantify the traffic flow which can be handled by a network connection. The throughput is measured with the public domain command netperf.

Round-trip. The round-trip time quantifies the response offered by a network connection. It will be measured, before the throughput, across the same connections as the throughput. The round-trip time is measured with the system command ping.

Load. This is expressed here as the number of fully active processes at a host. It is not a network parameter, but it may help to explain unexpected performance decreases. The load is measured at the current host using the system command uptime.

The presentation of the results is Web based using a Java Applet to load the data from the files into the memory of the Web browser from a user analysing the results.

RTPL is described in detail at [R11]

Iperf and Throughput Measurements - IperfER

Iperf [R16] is a tool to measure maximum TCP bandwidth, allowing the tuning of various parameters and UDP characteristics. It reports bandwidth, delay jitter and datagram loss. Iperf is widely used however [R17] describes the results of using iperf between several HEP institutes and therefore provides a good example of its usage.

IperfER has been developed based upon the PingER software but replacing the RTT and packet loss measurement based upon ping with TCP throughput measurements making use of the iperf tool. The graphical output from iperfer is very similar to pinger, and the throughput metrics are made available to the middleware via LDAP in a manner consistent with pinger.

UDPMon

UDPmon gives an estimate of the maximum usable bandwidth between two end nodes, a measurement of the packet loss, and the packet jitter or variation in the arrival times between consecutive packets. This packet jitter is an estimator of the variations in the one-way latencies of the packets traversing the network, as defined in [R4]. UPDMon has been developed by Richard Hughes-Jones (PPARC) within WP7 and, as there exists no specific reference describing its operation, a short description is provided here.

UDPmon uses two programs, a listener called udp_bw_resp that receives the incoming test data and a monitor program called udp_bw_mon that performs the test to the remote host.

The test uses UDP/IP frames with the application protocol shown in Figure 3. The test starts with the Requesting node sending a “Clear Statistics” message to the Responder. On reception of the OK acknowledgement, the Requesting node sends a series of “data” packets separated with a given fixed time interval between the packets. At the end of the test, the Requesting node asks for the statistics collected by the Responding node. Packet loss for the control messages are handled by suitable time-outs and re-tries in the Requesting node. If a test is already in progress when a “Clear Statistics” message arrives at a Responder, the Requestor is sent a “Please Defer” message. The defer procedure prevents multiple concurrent tests from distorting the throughput measurements.

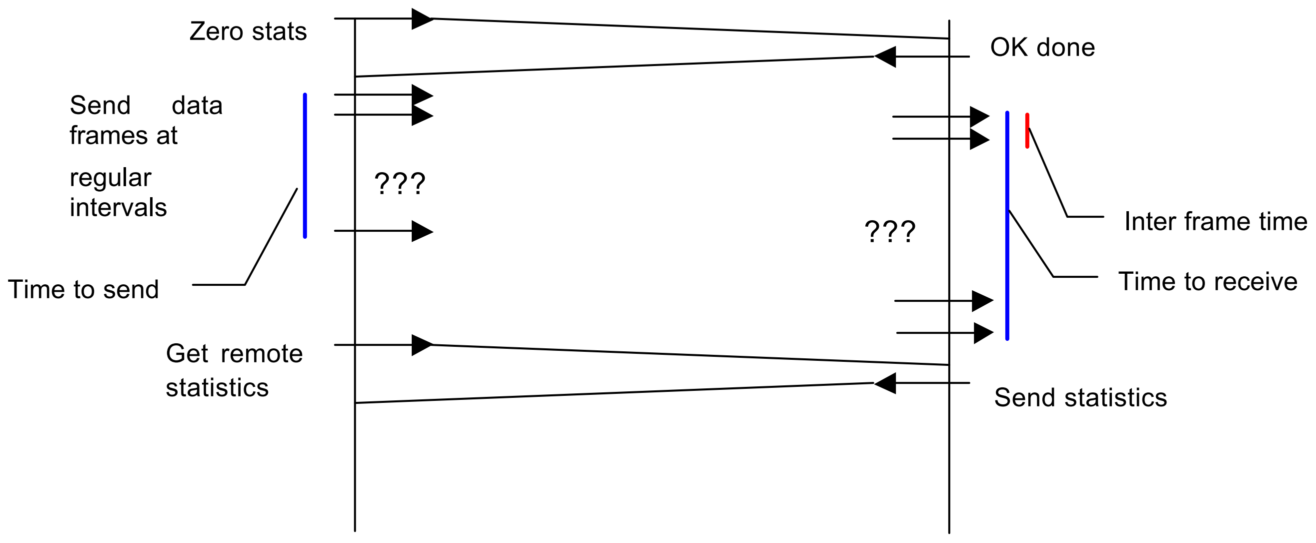


Figure 3: Protocol for the Bandwidth and Packet loss measurements

The transmit throughput is found using the amount of data sent and the time taken; the receive throughput is calculated from the amount of data received and the time from the first data packet to the last packet received.

Packet loss is measured by the remote node by checking that sequence numbers in the packets increase correctly, this also detects out-of-order packets. The number of packets seen, the number missed as indicated the sequence number check, and the number out-of-order are reported at the end of each test.

The Responding node also measures the time between the arrival of successive packets and produces a histogram of these times. This histogram may be requested by the Requesting node at the end of the test. Queue lengths in the network may be investigated by comparing the extra time taken to receive the burst of packets. [Remember that the available bandwidth and packet loss may be different from node a->b and node b->a and he socket buffer sizes, IP precedence bits, and IP tos bits may be set using suitable command line switches.]

The UDPmon tool has been fully integrated into the WP7 monitoring architecture and is provided with Perl scripts for scheduling the tests, generating historical time sequence plots and tables. An LDAP backend script has been provided so that the current snapshot may be obtained for publication to the middleware.

MapCenter

MapCenter [R20] has been created to display a flexible presentation layer of services and applications on a Grid. Current monitoring technologies have great functionalities and store different and accurate results in "Information System" of Grids, but generally, there is no efficient ways to graphically represent all communities, organization, applications running over Grids. MapCenter has been designed to fill this gap.

MapCenter polls computing elements (objects) with different methods, for example, to send ICMP requests (ping) to check the connectivity of computing elements; to make TCP connections to designated ports to check services running on computing elements; and will be able to make request to Information Systems of Grids to check specific grid services availability. MapCenter is able to display various views of grids, for example, via graphical maps; with logical views of services; and with full tree of computing elements.

MapCenter is really focused on the presentation layer in a huge and heterogeneous environment, as in a Grid context. It offers a very flexible and simple model that enables representation of any level of abstraction (national and international organizations, virtual organizations, application etc) needed by such environments. In principle MapCenter can be extended to monitor and present other metrics.

MapCenter has been developed by Franck Bonnassieux (CNRS) within WP7.

RIPE NCC Test Traffic Measurements

The goal of the Test Traffic project is to provide independent measurements of connectivity parameters, such as delays and routing-vectors, in the Internet. The project implements the metrics discussed in the IETF IPPM working group. Work on this project started in April 1997 and over the last years, it has been shown that the set-up is capable of routinely measuring delays, losses and routing vectors on a large scale. The Test Traffic project is being moved to a service offered by RIPE NCC to the entire community. This work is described in [R12]

Surveyor

Surveyor [R13] is a measurement infrastructure that is being currently deployed at participating sites around the world. It is based on standards work being done in the IETF IPPM WG. Surveyor measures the performance of the Internet paths among participating organisations. The project is also developing methodologies and tools to analyse the performance data.

One-way delay and packet loss are measured for each of the paths by sending time stamped test packets from one end of the specified path to the other. The one-way delay is computed by subtracting the timestamp in the packet from the time of arrival at the destination machine.

NIMI

The National Internet Measurement Infrastructure (NIMI) is a project, begun by the US National Science Foundation and currently funded by DARPA, to measure the global internet. It is based on a Network Probe Daemon and was designed to be scalable and dynamic. NIMI is scalable in that NIMI probes can be delegated to administration managers for configuration information and measurement co-ordination. It is dynamic in that the measurement tools are external as third party packages that can be added as needed. Full details can be found at [R28] together with a discussion of associated issues at [R29].

MRTG

The Multi Router Traffic Grapher (MRTG) [R14] is a passive tool to monitor the traffic load on network-links. MRTG generates HTML pages containing images which provide a live visual representation of this traffic. MRTG consists of a Perl script which uses SNMP to read traffic counters of routers, logs the traffic data and creates graphs representing the traffic on the monitored network connection. These graphs are embedded into web pages which can be viewed from any modern Web-browser.

TracePing

Traceping [R15] uses packet loss as its metric of network quality. It has been found, in every case investigated, that the variations in packet loss recorded by traceroute and ping, and hence by Traceping, reflect changes in the real performance experienced at the user level. No measurements or estimates are made of either the systematic or random errors on the packet loss data. The numbers should simply be interpreted as qualitative indicators of the state of a network connection: the higher the numbers the lower the quality. Currently traceping is a VMS specific tools so its value is strictly limited although there is a proposal to make it more generally applicable.

Netflow and cflowd

Cflowd [R18] has been developed to collect and analyse the information available from NetFlow flow-export. It allows the user to store the information and enables several views of the data. It produces port matrices, AS matrices, network matrices and pure flow structures. The amount of data stored depends on the configuration of cflowd and varies from a few hundred Kbytes to hundreds of Mbytes in one day per router.

Network Weather Service

The NWS [R19] is a distributed system that periodically monitors and dynamically forecasts the performance that various network and computational resources can deliver over a given time interval. The service operates a distributed set of performance sensors (network monitors, CPU monitors, etc.) from which it gathers readings of the instantaneous conditions. It then uses numerical models to generate forecasts of what the conditions will be for a given time frame. The functionality is intended to be analogous to weather forecasting, and as such, the system inherits its name.

The NWS provides an inclusive environment to which additional sensors may be added. While it may be seen and used merely as a means of providing forecast data based upon current metric measurement, in reality it can provides a complete, self-contained, monitoring environment.

NetSaint

NetSaint [R26] is a program that will monitor hosts and services on a network. It has the ability to email or to page when a problem arises and when it gets resolved. NetSaint is written in C and is designed to run under Linux, although it should work under most other Unix variants. It can run either as a normal process or as a daemon, intermittently running checks on various services that are specified. The actual service checks are performed by external "plugins" which return service information to NetSaint. Several CGI programs are included with NetSaint in order to allow the current service status, history, etc. to be viewed via a web browser.

5.2. PASSIVE NETWORK MONITORING

The Grid Applications

Grid applications are expected to record the throughput data experienced in their normal operation and to make it available along with the data collected by other network monitoring tools. Already Grid ftp has the necessary capability to record such information on each transfer. This information will form a major component of the network monitoring effort recorded by the Grid but the information it provides will need to be carefully interpreted and compared with the results from active monitoring of the Grid.

Per flow application throughput with GridFTP

GridFTP is an extension of the standard FTP mechanism for use in a Grid environment. It is envisaged that it will be used as a standard protocol for data transfer. Per flow application throughput defines the throughput measured for a specific GridFTP transfer between specified end-points. It will be based on a 'passive' measurement of the data transferred - i.e. only the information transferred - not additional (test) data. The data identified to be stored regarding a GridFTP transfer is as follows,

- Source
- Destination
- Total Bytes / Filesize
- Number of Streams Used
- TCP buffer size
- Aggregate Bandwidth

- Transfer Rate (Mbytes/sec)
- Time Transfer Started
- Time Transfer Ended

A schema has been proposed [R27] that utilises a patched version of GridFTP in which transfers are recorded and summary data stored. This is not incorporated here as debate is continuing as to exact format. However, it is hoped that once a standard form of network monitoring schema is formed, both schemes will be unified.

Future releases of GridFTP propose to incorporate features such as automatic negotiation of TCP buffer/window sizes and parallel data transfer, and reliable data transfer

Grid volume

To supply a measurement of the volume of data transferred on the grid as a function of time (daily/weekly/monthly/yearly), it has been proposed that information from data transfer across the grid be aggregated as transfers proceed. Should GridFTP be used for data transfer, then the various variables shall be obtained from GridFTP measurements and simply added, otherwise the software used for transfers should take account of this requirement. This may or may not include data transferred from active (test) data.

An alternate approach is to infer Grid volume from the counters available in network devices via for example SNMP. This requires the identification of Grid traffic from the set of traffic carried by the network devices and will require further investigation.

6. THE PUBLICATION OF NETWORK METRICS TO THE GRID MIDDLEWARE

6.1. LDAP

LDAP is a form of database that utilises a directory structure and allows the storage of data in attribute fields. These attribute fields are defined in object classes from which a hierarchy and properties of the attributes are defined. A schema is used to represent this information such that the data stored within it are useful and define the following:

- An unique naming schema for object classes
- Which attributes the object class must and may contain.
- Hierarchical relationships between object classes.
- The data type of attributes.
- Matching rules for attributes.

Globus is implementing its own schema for describing computing resources, however there has been little work to describe the volume of data transferred across a network, and network-monitoring metrics. As with LDAP servers and associated schema, the Globus schema can be easily modified, however, in order to maintain consistency amongst the various GIIS infrastructures, the schema structure requires a well-known framework. An example of such a framework - more formally a directory information tree (DIT) - is shown in Figure 4.

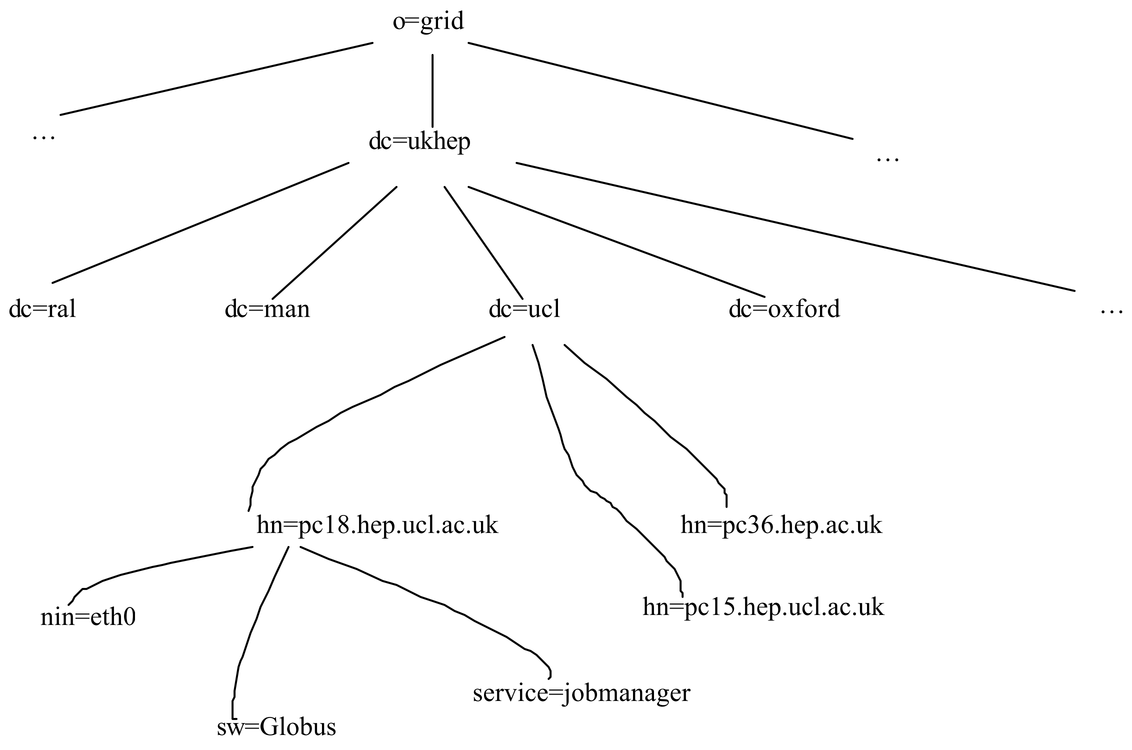


Figure 4 : An example of a DIT representing some UK HEP hosts

6.2. GRIS AND GIIS

The proposed Globus Grid architecture contains two layers in which higher-level services can be built on top of core services. This would enable the re-use of services across applications and tools.

The Globus project has defined two server types which are used to publish Grid resource information - the GRIS and GIIS. A GRIS contains information about a compute element or storage element. A GIIS is an index of all the other GIISs and GRISs on the Grid. Consumers wishing to find out the situation on a particular host on the Grid query the GIIS and are redirected, possibly via other GIISs, to a GRIS machine where that information resides. It is also possible that the GIIS machines may cache data from the GRISs to save redirecting consumers, with updates made using either a pull or push mechanism.

The GIIS is assumed only to use referrals to redirect consumers to the appropriate GRIS machine. If caching occurs at the GIIS, some precautions would have to be made to ensure that it always has up-to-date information, as passive monitoring would occur all the time, and so changes may happen very quickly.

There is a hierarchical structure associated with GRISs and GIISs, which is shown in Figure 5 where a set of GRIS machines in a local network all report to a local GIIS. A clique of local GIIS machines in small area all report to a single "town" GIIS machine. All these GIISs report to a country GIIS. The

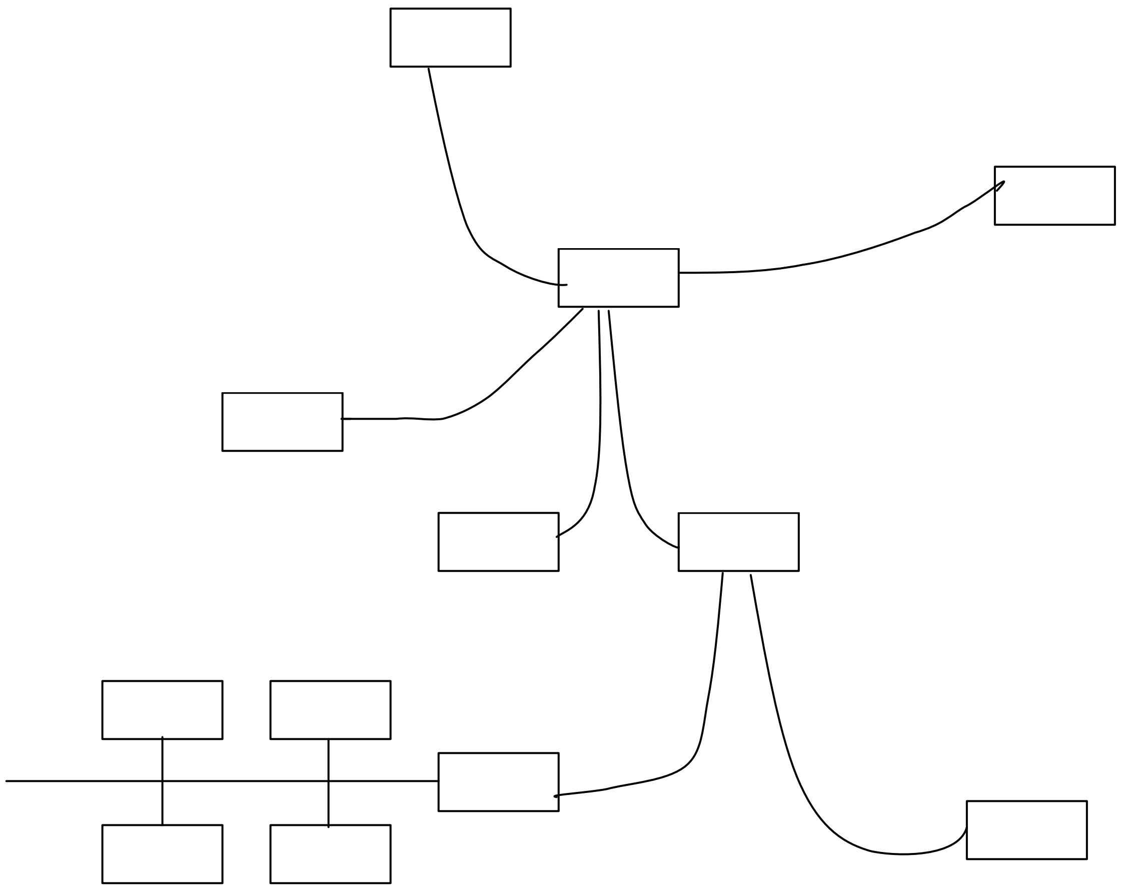


Figure 5 : Proposed structure of GRIS/GIIS relationship.

country GIIS machines are all equal in the hierarchy, and communicate accordingly.

6.3. COMPUTE ELEMENTS AND STORAGE ELEMENTS

Passive monitoring is likely to occur on CEs and SEs as they will be using GridFTP to transfer data. Publication of this data will occur on GRIS machines. Active measurements are unlikely to be made from CEs or SEs as there will likely be dedicated machines to do that.

6.4. PUBLICISING TO THE MIDDLEWARE

LDAP application

It is suggested that the programs that gather or produce the data shall be separate from the GIIS and interfaced via a LDAP backend. The network monitoring application(s) produce a log file from which the network metric values are calculated and/or extracted and converted to a format (such as LDIF) that can then be published on the LDAP server of that machine. This is shown in Figure 6.

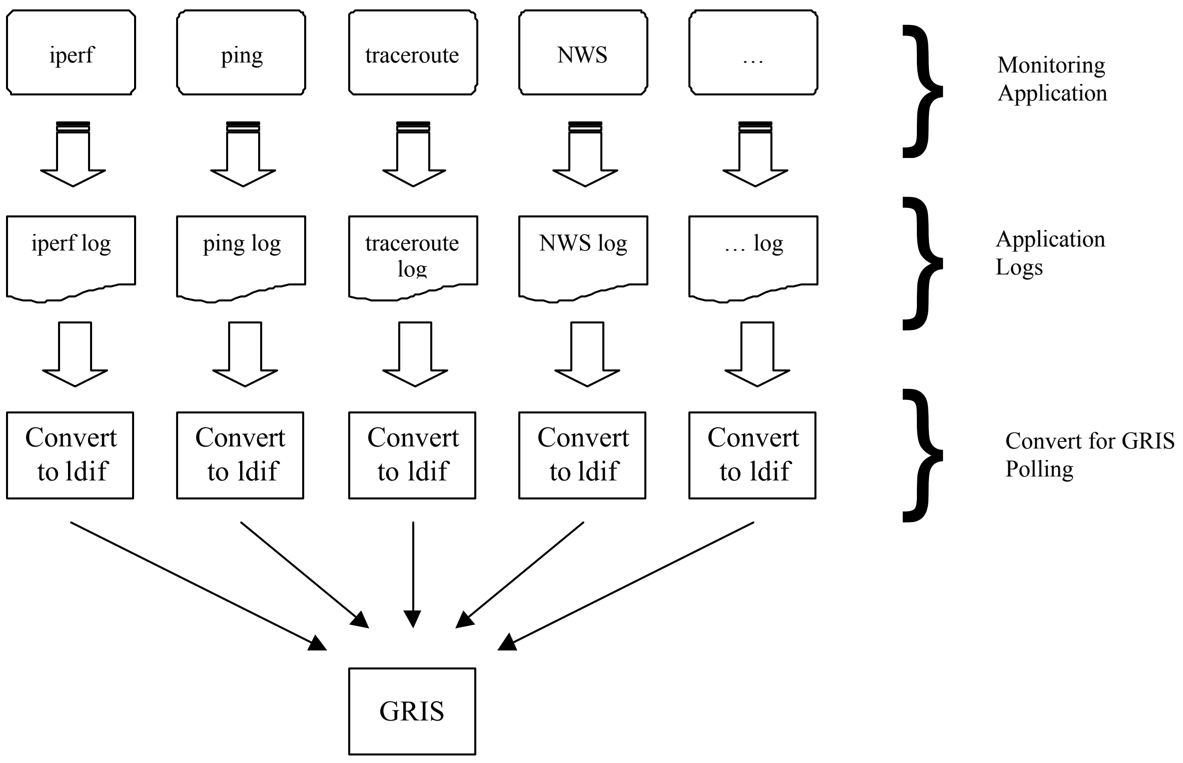


Figure 6 : The relation between data produced by network monitoring programs and the LDAP representation of that data.

By adopting the scheme shown in Figure 6 the use of any user specified network-monitoring package can be used and the information stored on the LDAP directory – as long as an appropriate backend is implemented on the LDAP. This provides extensibility and has the benefit of flexibility as administrators can choose to implement preferred network monitoring tools. However, for this deliverable, the type and number of applications have been limited to certain widely available tools to maintain simplicity and to demonstrate capability at this early stage of the DataGrid development.

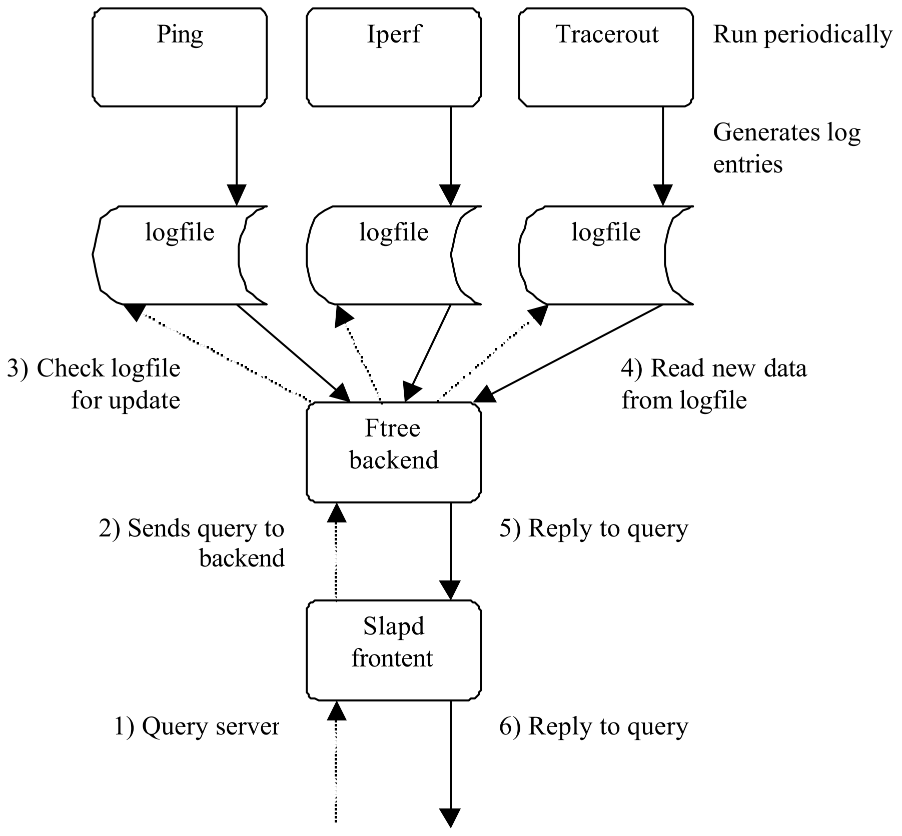


Figure 7 : Example data flow from LDAP request to response including LDAP backend access to network metrics

A initial implementation of the proposed tree structure has been produced using an ftree [R21] backend to slapd [R22]. The data flow of request from the middleware for a specific network metric to the response is shown in Figure 7. This formed a part of the WP7 PM9 network monitoring release and is extended here to support the other network monitoring tools of this deliverable.

Backends

The production of log files will differ in terms of content. It is the purpose of a LDAP backend to convert the log information containing networking metrics into something that can be published on the LDAP. This, in principle implies that every network-monitoring program will require its own LDAP backend to convert the data for publication on to the GRIS.

It has been proposed by groups in the US to adopt a standard NetLogger log file format to store the networking monitoring information. This, if implemented, should allow a single backend to enter network-monitoring information into the LDAP.

Other groups also propose to use "application wrappers" that can be used to encapsulate each network monitoring application. This has the benefit of a centralised interface to setup update frequency and maintenance. Applications such as NIMI support this, whilst NWS required specifically written modules to gather such information.

For the purposes of this PM12 deliverable, specific backend scripts have been provided to support the particular network monitoring tools of the delivery. This is done to demonstrate capability, to provide experience of metric publication , and to move the discussion of future developments forward.

6.5. FUNCTIONALITY OF STORAGE SYSTEM

It is expected that active network performance measurements will only ever be made between networks and not between individual machines on those networks because of the great number of measurements which would have to be made in the latter case. For this purposes, it is assumed that a network corresponds to a site or organisational unit, and therefore that measurements will be made from one site to another.

Each measure of network performance is affected by a great number of parameters, some of which are under the control of the tools used. The information each tool produces can be divided according to these parameters - which gives more detailed information to a consumer about how to optimise a data transfer. It can also be aggregated across all the different values of the parameters – which gives an overview of the performance which can be expected for a data transfer. These parameters variously include packet size, buffer sizes and the number of streams used, and not all measures of network performance are affected by all of them.

A performance-measuring tool might make many measurements of the same variable, and all this information can be collated into something more manageable. How the information is collated depends on its type, and is explained in more detail below.

For the purposes of the discussions here, measurements will be stored per site, e.g. a defined set of CE's and SE's, as opposed to per host. The reason for this is that the sensor machine will not necessarily be a Grid machine, but a dedicated sensor machine, which measures the route from it to other dedicated machine at remote sites.

6.6. DIT AND INHERITANCE

The meaning of many object classes is, in part, defined by their position in the directory information tree (DIT). The proposed DIT is shown in Figure 8.

The `NetworkMonitorOU` object is used to store the organisational units to which the measurements were made. Measurements made with default parameters are stored at the same level; the entity at this level will be of objectclass `NetworkMonitorOU` as well as `NetworkMonitorRTT`, `-Loss`, `-HopCount`, `-Throughput` and so on.

Aggregated information for each tool is stored in the tree at the level below the remote hostname to which that information pertains. For example, the average RTT measured with ping (for any packet size) to qmw might be found in the attribute named `rttavg` under the DN "tool=ping, rou=qmw, cn=netmon, ou=ucl, dc=ukhep, o=grid", where the name of the tool is given by the `tool` field of `NetworkMonitorTool`. Below that level, information is divided according to the parameters used to define it. For example, to find the average RTT for a packet size of 100 bytes, the DN would be "rttpacketsize=100, tool=ping, rou=qmw, cn=netmon, ou=ucl, dc=ukhep, o=grid". To search for the RTT measured by a default tool[1], the entity at "rou=qmw, cn=netmon, ou=ucl, dc=ukhep, o=grid" would contain RTT data if it has attributes from the `NetworkMonitorRTT` object class.

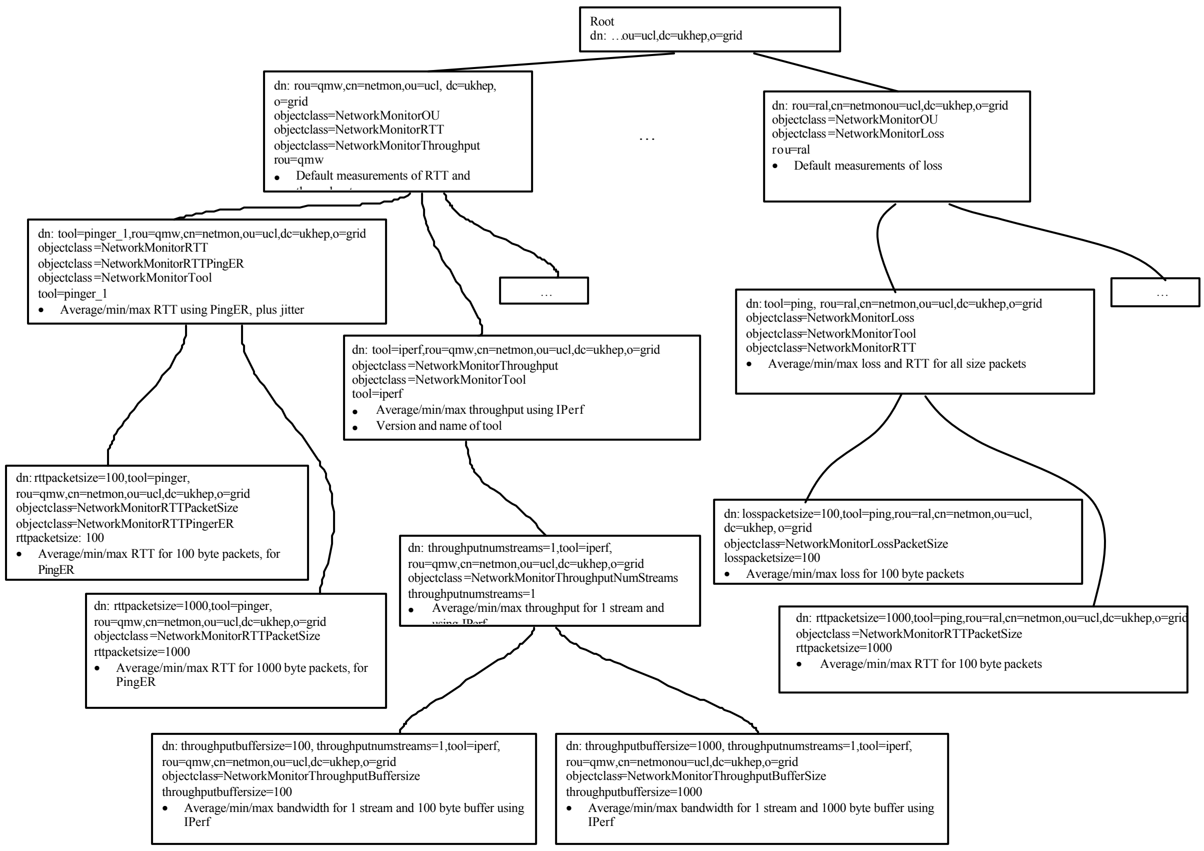

Root
dn: …ou=ucl,dc=ukhep,o=grid
dn: rou=qmw,cn=netmon,ou=ucl, dc=ukhep, o=grid
objectclass=NetworkMonitorOU
objectclass=NetworkMonitorRTT
objectclass=NetworkMonitorThroughput
rou=qmw
• Default measurements of RTT and
…
dn: rou=ral,cn=netmonou=ucl,dc=ukhep,o=grid
objectclass=NetworkMonitorOU
objectclass=NetworkMonitorLoss
rou=ral
• Default measurements of loss
dn: tool=pinger_1,rou=qmw,cn=netmon,ou=ucl,dc=ukhep,o=grid
objectclass=NetworkMonitorRTT
objectclass=NetworkMonitorRTTPingER
objectclass=NetworkMonitorTool
tool=pinger_1
• Average/min/max RTT using PingER, plus jitter
…
dn: tool=ping, rou=ral,cn=netmon,ou=ucl,dc=ukhep,o=grid
objectclass=NetworkMonitorLoss
objectclass=NetworkMonitorTool
objectclass=NetworkMonitorRTT
• Average/min/max loss and RTT for all size packets
…
dn: tool=iperf,rou=qmw,cn=netmon,ou=ucl,dc=ukhep,o=grid
objectclass=NetworkMonitorThroughput
objectclass=NetworkMonitorTool
tool=iperf
• Average/min/max throughput using IPerf
• Version and name of tool
dn: rttpacketsize=100,tool=pinger,
rou=qmw,cn=netmon,ou=ucl,dc=ukhep,o=grid
objectclass=NetworkMonitorRTTPacketSize
objectclass=NetworkMonitorRTTPingerER
rttpacketsize: 100
• Average/min/max RTT for 100 byte packets, for PingER
dn: throughputnumstreams=1,tool=iperf,
rou=qmw,cn=netmon,ou=ucl,dc=ukhep,o=grid
objectclass=NetworkMonitorThroughputNumStreams
throughputnumstreams=1
• Average/min/max throughput for 1 stream and
dn: losspacketsize=100,tool=ping,rou=ral,cn=netmon,ou=ucl,
dc=ukhep, o=grid
objectclass=NetworkMonitorLossPacketSize
losspacketsize=100
• Average/min/max loss for 100 byte packets
dn: rttpacketsize=1000,tool=pinger,
rou=qmw,cn=netmon,ou=ucl,dc=ukhep,o=grid
objectclass=NetworkMonitorRTTPacketSize
rttpacketsize=1000
• Average/min/max RTT for 1000 byte packets, for PingER
dn: rttpacketsize=1000,tool=ping,rou=ral,cn=netmon,ou=ucl,dc=ukhep,o=grid
objectclass=NetworkMonitorRTTPacketSize
rttpacketsize=1000
• Average/min/max RTT for 100 byte packets
dn: throughputbuffersize=100, throughputnumstreams=1,tool=iperf,
rou=qmw,cn=netmon,ou=ucl,dc=ukhep,o=grid
objectclass=NetworkMonitorThroughputBuffersize
throughputbuffersize=100
• Average/min/max bandwidth for 1 stream and 100 byte buffer using IPerf
dn: throughputbuffersize=1000, throughputnumstreams=1,tool=iperf,
rou=qmw,cn=netmonou=ucl,dc=ukhep,o=grid
objectclass=NetworkMonitorThroughputBufferSize
throughputbuffersize=1000
• Average/min/max bandwidth for 1 stream and 1000 byte buffer using IPerf

Figure 8 : Proposed DIT for organising network monitoring measurements

Entities in the DIT can be of more than one object class - an entity that has two object classes will have all the attributes from both classes (although some or all of them may be optional attributes). Some tools may produce more than one type of data as classified by their object. In this case, the entries for that tool may have many object classes, corresponding to the different types of data that it produces.

Object classes which inherit attributes from other object classes do not necessarily attach exactly the same meaning to those attributes. In the following section, most of the inheritance is from an object

[1] The 'default' tool and parameters used could be arbitrarily chosen at each site

class representing an overview of performance data to an object class representing the same performance data for a particular value of a parameter.

6.7. INTEGRATION INTO THE GIIS

The scheme described so far does not mention any way of recording *from* which site measurements are made. It is proposed that the tree would be integrated into the one that might be found on a local GIIS server, as for example shown here, in Figure 9.

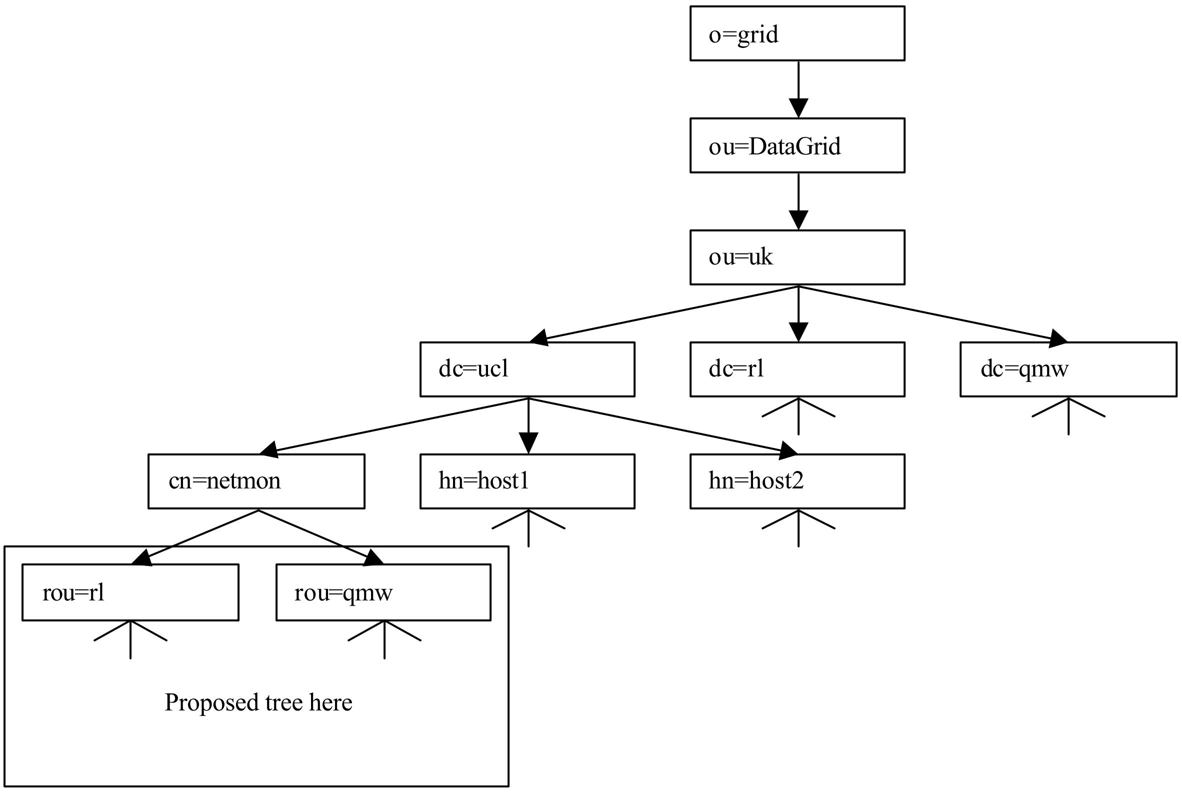


Figure 9 : Proposed position of the tree within the standard tree structure

This does not imply that all these entries would actually be on one server, as the LDAP referencing mechanism can invisibly construct this structure. However, it is expected that the results would be physically published on the site GIIS to which they refer, whilst measurements might be made by another host, and the results transferred in some way.

6.8. SCHEMA

Below are described the attributes of each object class associated with each type of network monitoring metric.

RTT

Round trip time is affected mainly by packet size, so as well as the mean, maximum, minimum etc RTT, entities are stored below that to give per-packet size information.

The `NetworkMonitorRTT` object is used to store information about the RTT for all packet sizes. The minimum and maximum RTTs encountered are stored in `rttmin` and `rttmax`, and the average of all measurements made of RTT is stored in `rttavg`. Some versions of 'ping' produce a value known as `mdev` (mean deviation), and the mean `mdev` for all packet sizes is also stored. The last time information was entered is also stored in `rttlastmodified`; this simply shows how up to date the information is.

`NetworkMonitorRTTPacketSize` is used to store information about the RTT for a specific packet size. It inherits all of the fields from `NetworkMonitorRTT` and adds a new field: `rttpacketsize`, which is used to record the packet size.

Loss

The number of packets lost en-route to a host is also affected by packet size. Thus there are two object classes similar to those for RTT, one of which contains aggregate packet loss information, and the other which contains packet loss information for a particular packet size.

The `NetworkMonitorLoss` object is used to store information about the RTT for all packet sizes. The minimum, maximum and average losses are stored in `lossmin`, `lossmax` and `lossavg`, respectively.

`NetworkMonitorLossPacketSize` inherits attributes from `NetworkMonitorLoss` and adds `losspacketsize`. It is used to store the packet loss for a particular packet size.

Hop count

The number of hops between two hosts is unlikely to change often, and is not affected by anything other than the topology of the network connecting them. For this latter reason, only one object class exists for publishing hop count data. The object class records the minimum and maximum number of hops counted in `hopsmin` and `hopsmax`, plus the current value in. The attribute `hopslastmodified` indicates the time at which the number of hops was last counted.

Throughput

Throughput can be measured using a variety of different tools, such as Iperf [IPERF] or Netperf [NETPERF]. Although these tools are different, they produce similar classes of data and are affected by the same protocol parameter changes. As such, the throughput data is classified by the number of streams used in the transfer, and the buffer size of the local machine

`NetworkMonitorThroughput` is used to store information from all numbers of streams and buffer sizes. It stores the maximum and minimum throughput achieved with any number of streams and any buffer size in `throughputtransmax` and `throughputtransmin`, and the average throughput in `throughputtransavg`. `throughputlastmodified` contains the last time this record was modified, in other words the last time a throughput measurement was made on this host.

`NetworkMonitorThroughputStreams` is used to store the average, maximum and minimum throughput for a particular number of streams. It inherits attributes from `NetworkMonitorThroughput` and adds the attribute `throughputnumstreams` which is used to record the number of streams.

`NetworkMonitorThroughputBufferSize` is used to store the average, maximum and minimum throughput for a particular number of streams and a particular buffer size. It inherits attributes from `NetworkMonitorThroughput` and adds an attribute to store the buffer size, `throughputbuffersize`. The number of streams is indicated by the object's position in the directory tree, below a `NetworkMonitorStreams` object.

6.9. HISTORICAL INFORMATION

This PM12 release proposes only to publish the most recent data for each metric. This may also contain basic statistical information such as minimum, maximum and average, calculated from data stored elsewhere. It is recognised that historical information is important for network monitoring and forecasting; consideration should therefore be given to include such capability in a later release.

For historical information it may be more efficient to publish data using a separate archiving system, which might extract and store data from the local GIIS or have the data installed directly by the monitoring tools themselves.

6.10. IMPLEMENTATION FOR THE DATAGRID

In addition to the Globus 2 Metacomputing Directory Service (MDS), WP3 have released a second MDS based on Alex Martin's Ftree [21] backend to OpenLDAP's [22] slapd server. Both of these MDSs are included in the Testbed 1 release of the European DataGrid [23]. The Ftree backend provides the ability to quickly search a tree structure of rapidly-changing data, which can be entered into the tree in LDIF format plain text by a simple script or executable. The specific network performance data available on the Testbed 1 release of the Ftree MDS, including its organisation and the queries required to retrieve it; together with a short description of the way data is collected and published is described here.

The tree structure in the PM9 implementation and followed in this release differs from the proposal document [24] in one major respect — the hostname of the remote monitoring host is used instead of the DN of the site. This difference resulted from the difficulty of implementing and maintaining a program to match DNs to hosts and vice-versa. Additionaly, it has been found that the absolute DN of a site may not be fixed between virtual organisations. These problems must be addressed in later releases. Network performance metric data is only available at sites which have installed both the MDS scripts and EDG-PingER.

The entities available in the testbed 1 release have attributes as shown in Table 1 (in addition to the standard LDAP attributes like objectClass).

Attribute	Meaning	Units
tool	the tool used to take measurements of a metric	
rou	the remote site to which measurements were made	
rttPacketSize	the packet size used when making RTT measurements	bytes
lossPacketSize	the packet size used when making loss measurements	bytes
rttMin	the minimum RTT at the last set of measurements	ms
rttAvg	the mean RTT at the last set of measurements	ms
rttMax	the maximum RTT at the last set of measurements	ms
lossAvg	the average loss at the last set of measurements	%

Table 1 : The attributes available on the testbed 1 release

Example queries

In order to extract performance data from a server, it must be queried using the LDAP protocol. Some examples of the parameters required for a query are shown here.

RTT between two sites

To extract the RTT in milliseconds from siteA.ac.uk to siteB.ac.uk, the following LDAP query would be formed:

server:	<local GIIS>:2171
base dn:	in=netmon, dc=siteA, ou=ukhep, o=Grid
match:	rou~=siteB.ac.uk
filter:	rttavg, rttmax, rttmin
scope:	one level

Loss between two sites

To extract the percent packet loss from siteA.ac.uk to siteB.ac.uk, the following LDAP query would be formed:

server:	<local GIIS>:2171
base dn:	in=netmon, dc=siteA, ou=ukhep, o=Grid
match:	rou~=siteB.ac.uk
filter:	lossavg
scope:	one level

The mappings between a URL-style site name (e.g. siteB.ac.uk) and a site DN are not entirely intuitive. This will have to be addressed in later testbed releases.

The backend script

This example uses PingER but is equally applicable to the other monitoring tools supported in this deliverable. A script executed by ftree-exec (fwp7-pinger.pl) uses HTTP to retrieve data from a PingER [25] server. PingER can be made to return a table of the most recently measured RTT and loss to a list of remote hosts, in tab-serparated value (TSV) format. 'wp7-pinger.pl' generates a series of entitites in LDIF format, forming as closely as possible the tree structure presented in [24]

'wp7-pinger.pl' accepts two command-line options: the first is the address of the PingER server; the second is the distinguished name that should be appended to all the distinguished names of all the entities generated by 'wp7-pinger.pl'.

Data and control flow

The data and control flow between the various elements of this monitor data publishing system is as follows:

Every 120 seconds Periodically, the PingER server must be checked for updates. This also happens at startup so that the tree can be initialised. The first time the script is run, it must return at least one entry with the distinguished name specified with the script name in the LDIF configuration file (i.e. the root dn of the entries generated by the script). This really means that that entry must be returned every time the script is run.

- Ftree runs the script (U1).
  - The script contacts the specified PingER server using HTTP GET (U2).
  - The PingER server returns a list of RTT and loss values measured to various remote hosts (U3).

- The script builds a tree in the format shown in [24]

  and sends it in LDIF format to standard output (U4), which is read by Ftree.
- The LDIF data returned is parsed and stored in memory by Ftree.

The reaper The reaper is run periodically as specified in the .conf file.

- For each entry, compare the time it was modified plus its time to live with the current time and remove the entry if it has expired.

On a query A client connects to the server to the LDAP server with a query.

- The client submits the query to the slapd server (Q1).
- slapd forwards the query to the Ftree backend (Q2).
- Ftree searches the database in memory and returns the query result to the slapd frontend (Q3).
- slapd returns the query result to the client (Q4).

The LDAP server can also handle persistent searches where updates to the database are forwarded to a client.

7. WP7 NETWORK MONITORING TESTBED

WP7 identified six sites that would together provide a testbed for the development of a network monitoring architecture. The sites provided an environment where monitoring tools could be deployed to understand their capabilities and to gain experience in their use. Contributing sites allowed appropriate access to the monitoring machines for the installation and management of the various monitoring tools. At all times the purpose was to install the monitoring tools at as many of the sites as possible so that some coherent view of the output from a specific monitoring tool could be collected. Each site also provided a Web interface to the monitoring tools they hosted as one of the means of publication of the particular network metric being monitored.

It should be noted that these sites have been used by WP7 to develop and test network monitoring tools and the network monitoring environment described here. As such these sites do not provide operational services but a collection of resources "under test".

The network monitoring testbed sites and their associated Web sites were established at

CERN:	http://pcgiga.cern.ch/datagrid.html
CNRS-Lyon:	http://ccwp7.in2p3.fr/monitor.html
INFN-Bologna:	http://grid001f.cnaf.infn.it/pinger/datagrid.html
SARA:	http://gridmon.sara.nl:8080/
PPARC-Daresbury:	http://icfamon.dl.ac.uk/ppncg/datagrid.html
PPARC-Rutherford:	http://icfamon.rl.ac.uk/ppncg/datagrid.html

8. CONTENT OF NETWORK MONITORING PM12 DELIVERABLE

8.1. INTRODUCTION

Table 2 shows the network monitoring packages and their metrics that will be included in the WP7 PM12 deliverable. Example output from these monitoring tools is shown below.

Network Monitoring Tool	Metric measured
pingER	connectivity, RTT, packet loss
RTPL	connectivity, RTT, packet loss, TCP throughput
iperfER	connectivity, TCP throughput
udpmon	connectivity, UDP throughput

Table 2 : Network monitoring tools included in PM12 deliverable

In addition the deliverable will contain a "model" Web page with access to the visualisation scripts associated with the above tools and the URLs shown in Table 3 Other monitoring resources included in PM12 deliverable that provide access to other closely related resources developed by WP7 are shown in Table 3

URL	Resource
http://www.ripe.net/cgi-bin/gttm/pod	On demand DataGrid plots from RIPE NCC TTBs located at WP7 NM testbed sites
http://ccwp7.in2p3.fr/mapcenter/	Instantaneous status of the DataGrid

Table 3 : Other monitoring resources included in PM12 deliverable

8.2. ISSUES

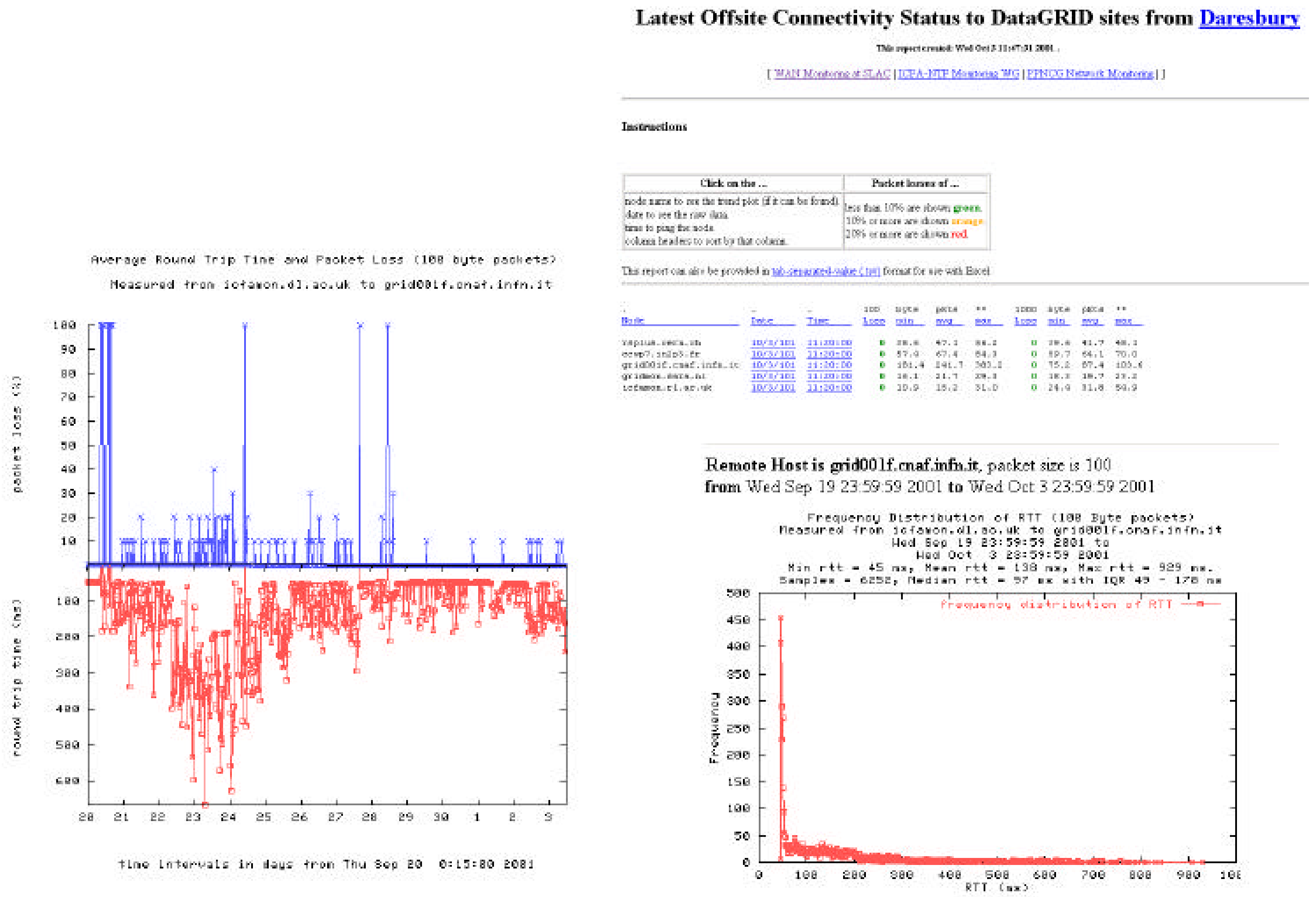
WP7 has determined that a fully meshed topology for network monitoring is unobtainable simply from the perspectives of scale and recognises as a consequence that not every query about network performance or its capability can be answered directly. This is not a limitation of the network monitoring architecture developed here, but a recognition of the scale of the Grid, and of the virtual organisations that exist within it.

Passive monit oring via the Grid applications themselves will provide a part of the solution, and future work of WP7 on monitoring forecasting and prediction may also provide assistance in this area. WP7 realises that there are no simple (or indeed moderately complex) solutions to this problem. It will however discuss these issues in the future with the intent of providing guidance and, as such, would welcome input to its deliberations.

8.3. PM12 DELIVERABLE -OUTPUT OF NETWORK MONITORING PACKAGE

PingER

PingER provides packet loss, RTT and RTT frequency metrics between specified sites. This information is available graphically or as a table via a Web server. The same information is made available to LDAP via an LDAP backend script. The Figure 10 shows representative output from the PingER tools.



Latest Offsite Connectivity Status to DataGRID sites from Daresbury

This report created: Wed Oct 3 11:47:31 2001 .

[WAN Monitoring at SLAC | ICFA-NTF Monitoring WG | PPNCG Network Monitoring |]

Instructions

Click on the ...	Packet losses of ...
node name to see the trend plot (if it can be found) date to see the raw data time to ping the node column headers to sort by that column	less than 10% are shown green 10% or more are shown orange 20% or more are shown red

This report can also be provided in tab-separated-value (.tsv) format for use with Excel

			100	byte	pkts	**	1000	byte	pkts	**
Node	Date	Time	Loss	min	avg	max	Loss	min	avg	max
vapius.cern.ch	10/3/101	11:20:00	0	38.6	47.1	88.2	0	38.6	41.7	48.1
ccwp7.in2p3.fr	10/3/101	11:20:00	0	57.9	67.4	84.3	0	59.7	64.1	70.0
grid001f.cnaf.infn.it	10/3/101	11:20:00	0	181.4	241.7	303.2	0	75.2	87.4	103.6
gridmon.sara.nl	10/3/101	11:20:00	0	16.1	21.7	29.3	0	18.3	19.7	23.2
icfamon.dl.ac.uk	10/3/101	11:20:00	0	10.9	15.2	31.0	0	24.9	31.8	59.9

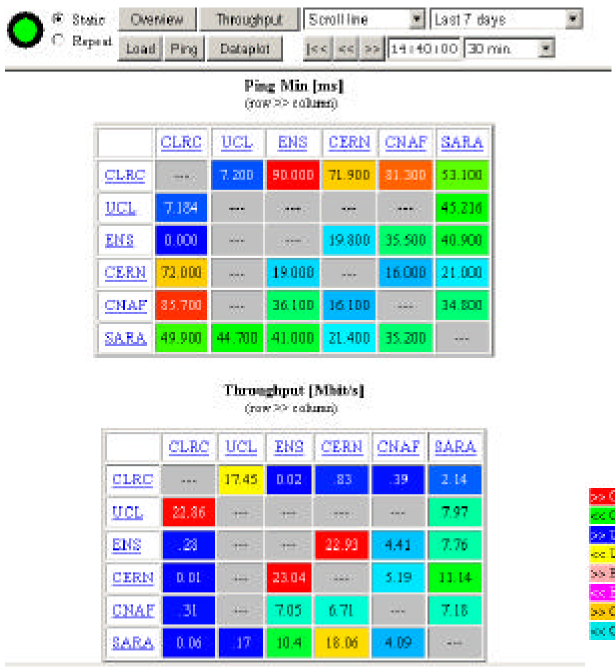
Remote Host is grid001f.cnaf.infn.it, packet size is 100
from Wed Sep 19 23:59:59 2001 **to** Wed Oct 3 23:59:59 2001

Average Round Trip Time and Packet Loss (100 byte packets)
Measured from icfamon.dl.ac.uk to grid001f.cnaf.infn.it
packet loss (%)
round trip time (ms)
time intervals in days from Thu Sep 20 0:15:00 2001
Frequency Distribution of RTT (100 Byte packets)
Measured from icfamon.dl.ac.uk to grid001f.cnaf.infn.it
Wed Sep 19 23:59:59 2001 to
Wed Oct 3 23:59:59 2001
Min rtt = 45 ms, Mean rtt = 138 ms, Max rtt = 929 ms.
Samples = 6252, Median rtt = 97 ms with IQR 49 - 170 ms
Frequency distribution of RTT
Frequency
RTT (ms)

Figure 10 : Representative output from the PingER tool showing RTT and Packet loss metrics

RTPL

RTPL provides packet loss, RTT and throughput metrics between specified site. This information is available graphically or as a table via a Web server. The same information is made available to LDAP via an LDAP backend script. The Figure 11 shows representative output from the RTPL tools.



Ping Min [ms]
(row >> column)

	CLRC	UCL	ENS	CERN	CNAF	SARA
CLRC	---	7.200	90.000	71.900	81.300	53.100
UCL	7.184	---	---	---	---	45.216
ENS	0.000	---	---	19.800	35.500	40.900
CERN	72.000	---	19.000	---	16.000	21.000
CNAF	85.700	---	36.100	16.100	---	34.800
SARA	49.900	44.700	41.000	21.400	35.200	---

Throughput [Mbit/s]
(row >> column)

	CLRC	UCL	ENS	CERN	CNAF	SARA
CLRC	---	17.45	0.02	.83	.39	2.14
UCL	22.86	---	---	---	---	7.97
ENS	.28	---	---	22.93	4.41	7.76
CERN	0.01	---	23.04	---	5.19	11.14
CNAF	.31	---	7.05	6.71	---	7.18
SARA	0.06	.17	10.4	18.06	4.09	---

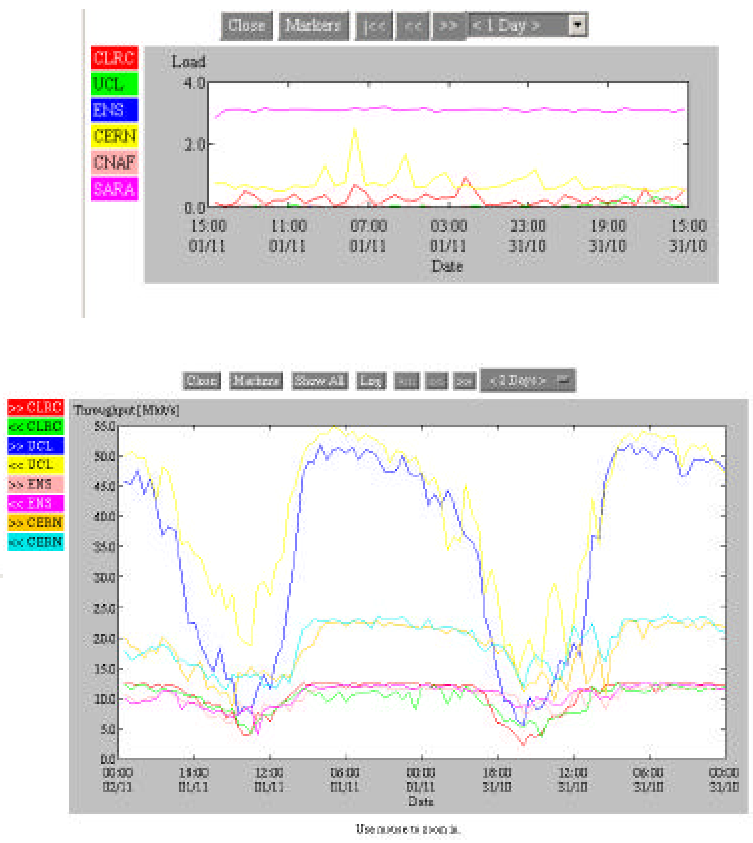


Figure 11 : Representative output from the RTPL tool showing RTT, packet loss and throughput metrics

IperfER

Iperfer measures TCP throughput between specified site and makes this information available graphically via a Web server. The same information is made available to LDAP via an LDAP backend script. The Figure 12shows representative output from the IperfER tool.

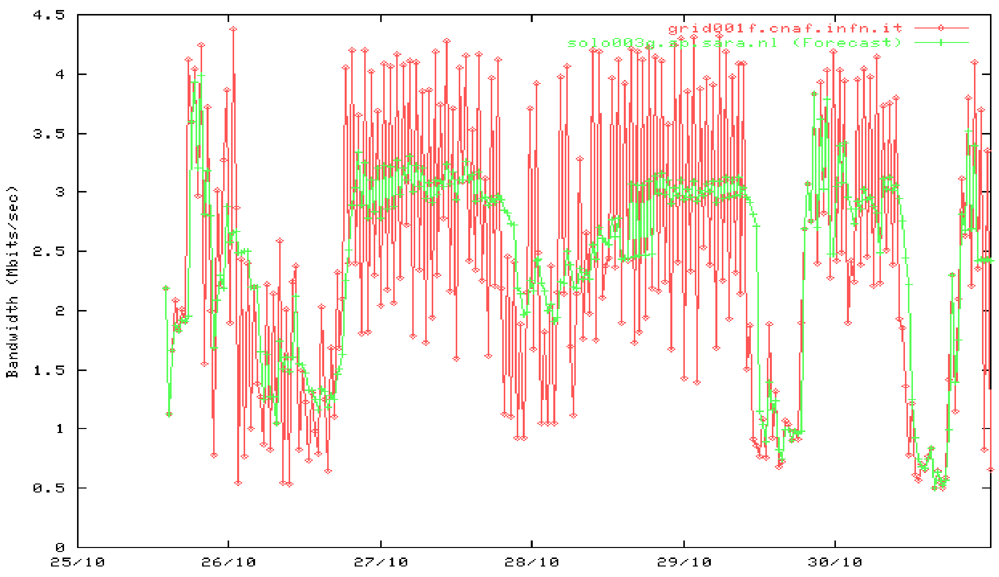


Figure 12 : Representative output from the Iperfer tool showing TCP throughput

UDPmon

UDPmon measures UDP throughput and packet loss between specified site. This information is available graphically or as a table via a Web server. The same information is made available to LDAP via an LDAP backend script. The Figure 13 shows representative output from the UDPmon tool.

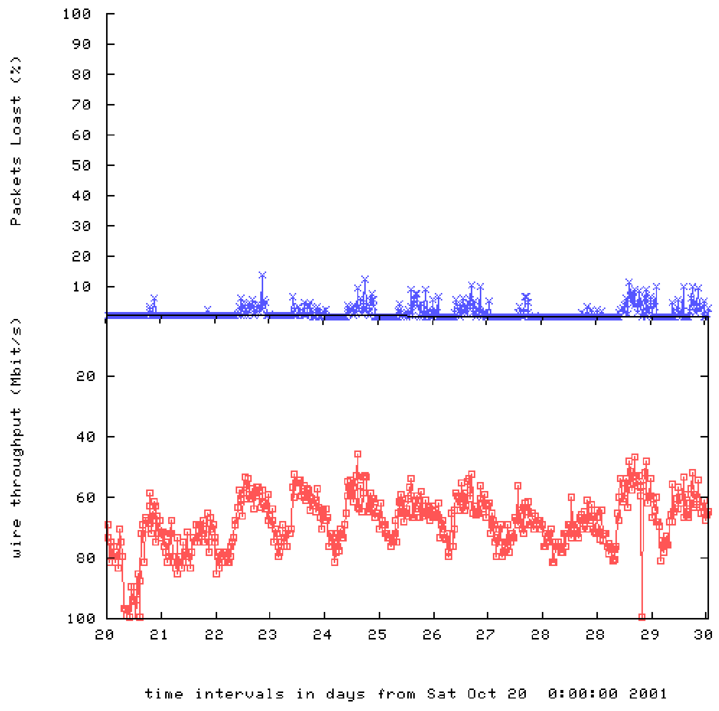
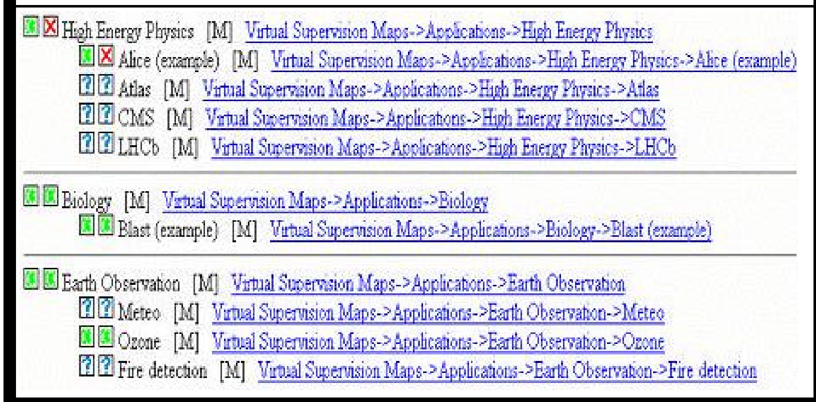


Figure 13 : Representative output from the UDPmon tool showing UDP throughput and packet loss

MapCenter

Logical view : Applications

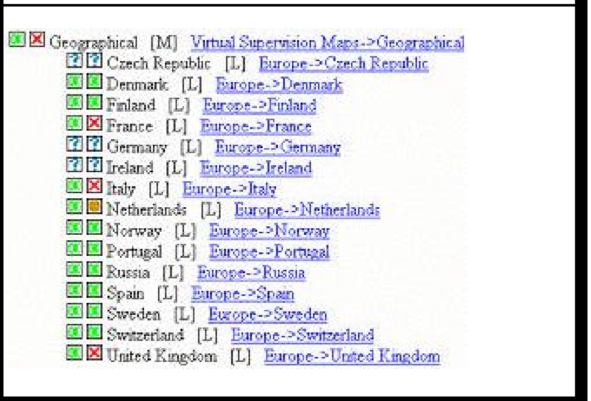


🟩🟥 High Energy Physics [M] Virtual Supervision Maps->Applications->High Energy Physics
- 🟩🟥 Alice (example) [M] Virtual Supervision Maps->Applications->High Energy Physics->Alice (example)
- ❓❓ Atlas [M] Virtual Supervision Maps->Applications->High Energy Physics->Atlas
- ❓❓ CMS [M] Virtual Supervision Maps->Applications->High Energy Physics->CMS
- ❓❓ LHCb [M] Virtual Supervision Maps->Applications->High Energy Physics->LHCb

🟩🟩 Biology [M] Virtual Supervision Maps->Applications->Biology
- 🟩🟩 Blast (example) [M] Virtual Supervision Maps->Applications->Biology->Blast (example)

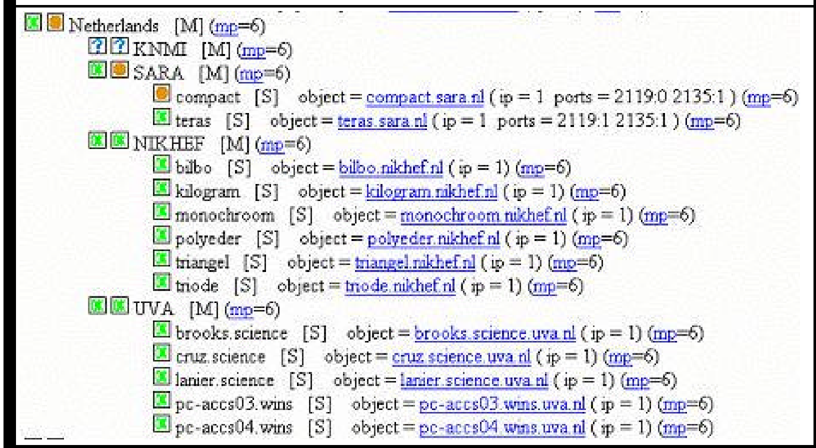
🟩🟩 Earth Observation [M] Virtual Supervision Maps->Applications->Earth Observation
- ❓❓ Meteo [M] Virtual Supervision Maps->Applications->Earth Observation->Meteo
- 🟩🟩 Ozone [M] Virtual Supervision Maps->Applications->Earth Observation->Ozone
- ❓❓ Fire detection [M] Virtual Supervision Maps->Applications->Earth Observation->Fire detection

Logical view : Geographical



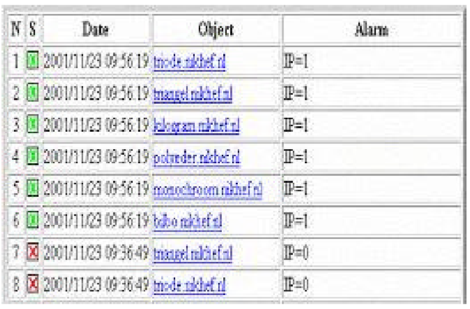
🟩🟥 Geographical [M] Virtual Supervision Maps->Geographical
- ❓❓ Czech Republic [L] Europe->Czech Republic
- 🟩🟩 Denmark [L] Europe->Denmark
- 🟩🟩 Finland [L] Europe->Finland
- 🟩🟥 France [L] Europe->France
- ❓❓ Germany [L] Europe->Germany
- ❓❓ Ireland [L] Europe->Ireland
- 🟩🟥 Italy [L] Europe->Italy
- 🟩🟧 Netherlands [L] Europe->Netherlands
- 🟩🟩 Norway [L] Europe->Norway
- 🟩🟩 Portugal [L] Europe->Portugal
- 🟩🟩 Russia [L] Europe->Russia
- 🟩🟩 Spain [L] Europe->Spain
- 🟩🟩 Sweden [L] Europe->Sweden
- 🟩🟩 Switzerland [L] Europe->Switzerland
- 🟩🟥 United Kingdom [L] Europe->United Kingdom

Full view : Netherlands example part



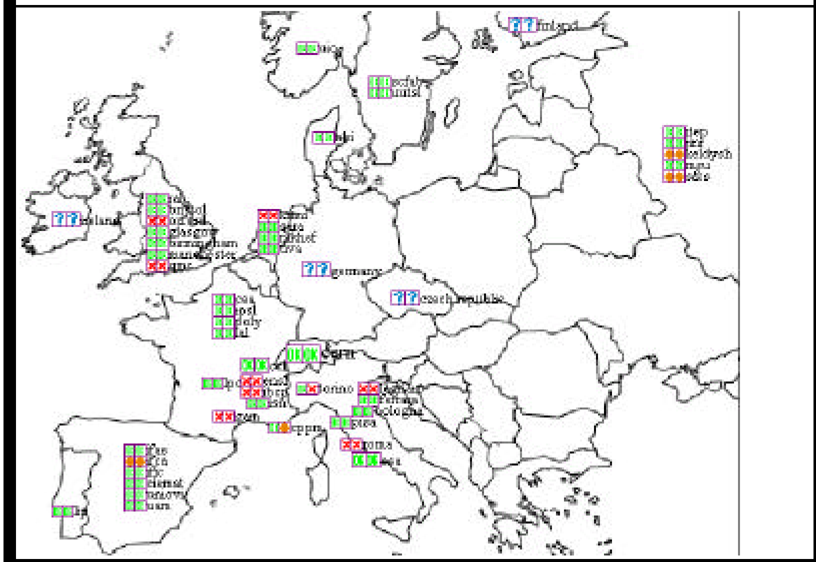
🟩🟧 Netherlands [M] (mp=6)
- ❓❓ KNMI [M] (mp=6)
- 🟩🟧 SARA [M] (mp=6)
  - 🟧 compact [S] object = compact.sara.nl (ip = 1 ports = 2119:0 2135:1) (mp=6)
  - 🟩 teras [S] object = teras.sara.nl (ip = 1 ports = 2119:1 2135:1) (mp=6)
- 🟩🟩 NIKHEF [M] (mp=6)
  - 🟩 bilbo [S] object = bilbo.nikhef.nl (ip = 1) (mp=6)
  - 🟩 kilogram [S] object = kilogram.nikhef.nl (ip = 1) (mp=6)
  - 🟩 monochroom [S] object = monochroom.nikhef.nl (ip = 1) (mp=6)
  - 🟩 polyeder [S] object = polyeder.nikhef.nl (ip = 1) (mp=6)
  - 🟩 triangel [S] object = triangel.nikhef.nl (ip = 1) (mp=6)
  - 🟩 triode [S] object = triode.nikhef.nl (ip = 1) (mp=6)
- 🟩🟩 UVA [M] (mp=6)
  - 🟩 brooks.science [S] object = brooks.science.uva.nl (ip = 1) (mp=6)
  - 🟩 cruz.science [S] object = cruz.science.uva.nl (ip = 1) (mp=6)
  - 🟩 lanier.science [S] object = lanier.science.uva.nl (ip = 1) (mp=6)
  - 🟩 pc-accs03.wins [S] object = pc-accs03.wins.uva.nl (ip = 1) (mp=6)
  - 🟩 pc-accs04.wins [S] object = pc-accs04.wins.uva.nl (ip = 1) (mp=6)

History of events

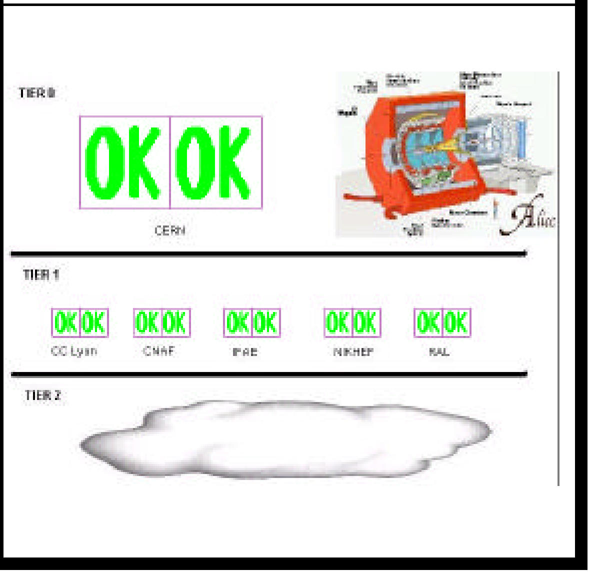


N	S	Date	Object	Alarm
1	🟩	2001/11/23 09:56:19	triode.nikhef.nl	IP=1
2	🟩	2001/11/23 09:56:19	triangel.nikhef.nl	IP=1
3	🟩	2001/11/23 09:56:19	kilogram.nikhef.nl	IP=1
4	🟩	2001/11/23 09:56:19	polyeder.nikhef.nl	IP=1
5	🟩	2001/11/23 09:56:19	monochroom.nikhef.nl	IP=1
6	🟩	2001/11/23 09:56:19	bilbo.nikhef.nl	IP=1
7	🟥	2001/11/23 09:36:49	triangel.nikhef.nl	IP=0
8	🟥	2001/11/23 09:36:49	triode.nikhef.nl	IP=0

Graphical view : Europe

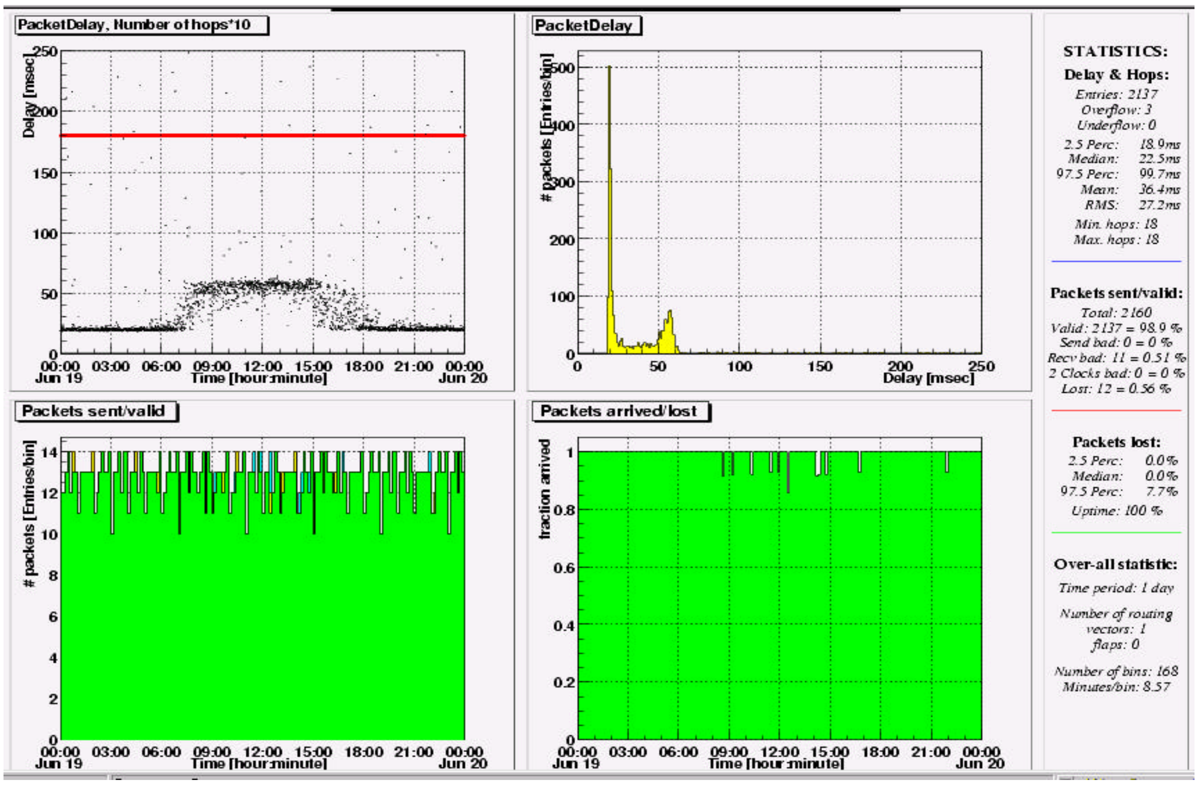


Graphical view : Alice Application



RIPE NCC Test Traffic Measurements

The PM12 deliverable provides access to the RIPE NCC TTM Web pages where the data collected between the WP7 network monitoring testbed sites can be reviewed. Figure 14 shows a typical output from this Web page showing packet loss, one-way delay and a measure of the route (the hop count) between the two sites involved in monitoring.



STATISTICS:

Delay & Hops:
Entries: 2137
Overflow: 3
Underflow: 0
2.5 Perc: 18.9ms
Median: 22.5ms
97.5 Perc: 99.7ms
Mean: 36.4ms
RMS: 27.2ms
Min. hops: 18
Max. hops : 18

Packets sent/valid:
Total: 2160
Valid: 2137 = 98.9 %
Send bad: 0 = 0 %
Recv bad: 11 = 0.51 %
2 Clocks bad: 0 = 0 %
Lost: 12 = 0.56 %

Packets lost:
2.5 Perc: 0.0%
Median: 0.0%
97.5 Perc: 7.7%
Uptime: 100 %

Over-all statistic:
Time period: 1 day
Number of routing vectors: 1
flaps: 0
Number of bins: 168
Minutes/bin: 8.57

Figure 14 : : Representative output from the RIPE NCC TTM box viewed from the RIPE NCC Web pages

9. FUTURE NETWORK MONITORING ACTIVITIES OF WP7

9.1. INTRODUCTION

WP7 has designed and developed a prototype network monitoring architecture and a coherent set of network monitoring tools to populate it. However further work is required in its development. Scalability and intrusiveness are major issues for such tools and during the coming year WP7 will focus its network monitoring activity as specified,

- review the usability of the metrics provided here;
- review the issue of scheduling network monitoring tasks;
- review the performance of the network metrics and their use and value to the Grid community and reflect this feedback into the network monitoring architecture as appropriate;
- consider the applicability and capability of prediction and forecasting methods for network metrics;
- review the use of Grid status tools i.e. MapCenter and NetSaint, and their role within the Grid testbed; and
- consider the opportunity for a common database to hold or to access all monitoring data.

9.2. PREDICTION AND FORECAST OF NETWORK METRICS

WP7 is about to commence on a review of prediction and forecasting of network monitoring metrics. This will include a review of the requirements for such capability by Grid applications and the middleware, a review of the tools and methodologies available; and an assessment of the validity of such approaches. Side by side with this work will be a review of the network monitoring architecture adopted for the PM12 deliverable described above. Whilst it is believed that this existing architecture can easily and simply accommodate forecasting tools and techniques consideration must be given to alternative approaches that provide a complete monitoring and forecasting environment. Whatever the merits of the alternatives available to WP7, prime consideration must be given to users of the Grid and their applications and the need to retain the goals of simplicity and extensibility.

9.3. NETSAINT

NetSaint is a tool to monitor network and computing resources both at LAN and WAN level. With respect to many monitoring tools NetSaint provides complete decoupling between the core logic and the monitoring engine. Monitoring in NetSaint is performed by a series of dedicated "plug-ins" which communicate to the server where the core logic and the web interface are running. In this way the collection of information can be made asynchronous and/or parallel and the servers can be ordered in a multi-layered hierarchy in a way similar to that of the Globus MDS 2 information system (with the obvious scaling benefits). The "core plus plug-in" way of work of NetSaint makes the user interface and the core logic (front-end) of the tool independent of the information system (back-end) underneath. An interface has been already written from NetSaint to RRDTool to plot the historical evolution of the parameters to be monitored and another is in progress to be coded to monitor/store with NetSaint network/resource parameters contained into a LDAP server. Many plug-ins already exist not only for network parameters but also for computing resources like CPU load, disk/tape space occupancy, etc. Of course, NetSaint can also monitor all the MIB parameters through the SNMP protocol.

In this regard, NetSaint provides a complete environment, indeed a monitoring architecture in its own right and how that fits into the network monitoring architecture described here has not yet been discussed and will form a part of ongoing WP7 work. For example NetSaint could be used as a replacement for all described here, or may be used in conjunction with the MapCenter application to provide a "view" into the DataGrid used to visualize the extent of the Grid and its current capabilities. The exact role for NetSaint has yet to be determined but as with the NWS prime consideration must be given to users of the Grid and their applications and the need to retain the goals of simplicity and extensibility.