Design Study for a Detailed Network Monitoring and Debugging Tool

Andreas Hirstius

August 18, 2005

Abstract

This document presents a possible design of a detailed network monitoring tool for the LHC Tier0 → Tier1 network infrastructure.
The data collection is meant to be easy and simple to deploy as it requires only support for standard protocols, such as SNMP, and freely available tools, although the basic ideas are of general nature.
By collecting all available information in one or two central network operation centres and together with the grid services monitoring information, the effort for debugging network problems or network related grid services problems is significantly reduced.

Contents

Chapter 1

Introduction

The proposed tool is not intended to replace the network monitoring tools of the carriers and National Research and Education Networks (NRENs).
It should rather collect the necessary information from all carriers and NRENs in order to provide a global overview over the functioning of the entire network infrastructure.
This is the only reasonable way to enable an efficient debugging in an environment that spans across a large number of technical/political/business/legal/administrative entities.
Furthermore, the monitoring systems of the carriers and NRENs cannot be combined with the information from the monitoring of the grid services.

The number of network devices to be monitored would be relatively small and basically "static".

Should a (fixed) backup route be known in advance, it would be helpful in a better understanding of the network if the device on this backup route could also have some basic monitoring running when there is no LHC traffic crossing those devices.

The lessons learned during the first two Service Challenges (see. [1]) led to the idea of a centralized monitoring of all network operations which are important for the LHC WAN infrastructure.
In order to keep the data collection easy and simple to deploy, only standard protocols should be used in a stand-alone type of application. In particular, this tool should be independent of any of the grid software, i.e. it should not require any of the still very complicated-to-deploy software infrastructure.
By using Java and Scripting languages it is also possible to create a data collection tool which can be deployed on any platform which provides a Java Virtual Machine and support for the Scripting languages, i.e. any available platform. If the tool is written in C or C++ it requires, of course, a recompilation for the platform on which it is supposed to run. Usually this is not a problem if a

portable coding style is used.

For in-depth analysis the collected information should be stored in a database. The data collection tool should present the information in a format which is supported by a variety of databases. This format would be a subset of SQL which is common to most databases (Oracle, MySQL, PostgreSQL, etc.).
Such a database will significantly help to find problems which develop over time and without clear symptoms, especially if the information can be combined with the grid services monitoring information. This is also true for problems with the grid services: If the grid services have a problem to debug, it is important to know what the underlying network looked like at the time the problem occurred. In fact it is quite likely that most of the more complex problems can only be debugged with both types of monitoring information available.

There are commercial network monitoring tools available. These tools are usually very expensive and some even require a specific operating platform. This means that anyone wanting to access the information has to have a license for the commercial tool in use. It is also not known if the commercial tools provide, or are able to provide, all of the proposed functionalities, because some of the proposed features are beyond the scope of "pure" network monitoring.

1.1 The Tools

1.1.1 Data Collection

The network equipment supports the collection of monitoring data via the Simple Network Monitoring Protocol (SNMP). So naturally the data collection software will use SNMP to retrieve the information from the network devices.
There are libraries or classes for all programming or scripting languages publicly available. For simplicity and maintainability, only one programming language and maybe one scripting language should be used.

There are a few reasons for running a decentralized data collection.
One reason is security. Even though SNMP version 3 has authentication and secure data transport mechanisms, it is much more secure to run the data collection software in the much more contained environment of a single carrier, NREN or computing centre. Another reason is the collection of information which would be very difficult, or impossible, to obtain if the data collection just runs from a single site. A useful application for such information would be the search for any type of asymmetry, such as asymmetric routing.

The European Tier 1 centres could, for example, monitor the network equipment between GEANT2 and their centre and GEANT2 could collect the information inside its own network infrastructure before sending the preprocessed information to the central monitoring site.

Since SNMP might not provide all the necessary information in all cases, it might be necessary to collect complementary information from different sources. This can be, for example, simple textfiles or (configuration) databases. An example of such additional information would be the layout of a Layer2 tunnel through a Layer3 network infrastructure. The main problem with such additional information is that it has to be kept up-to-date at all times by the respective persons in charge. This adds work for the responsible persons, but it will significantly improve debugging capabilities. Unfortunately in the past there have been some problems with the correctness of such information.

1.2 Data Storage

All information that has been collected should be stored permanently in a database or any other form which allows for searching and correlating information. Since there are different types of information which are collected (see next chapter), they could be, in principle, stored in different, i.e. specialized, databases.
In order to be able to used all available information, i.e. also the information from the grid services monitoring, there should be a simple way to access all database that are needed.

1.3 Data Analysis

The analysis of the information has two separate parts.
The first part is the graphical presentation of the collected information. This representation should have three different "stages"

- A general overview
- A detailed view of a "network cloud" - includes all active network devices
- A detailed view of a particular link - includes all interfaces in the devices

With such a detailed graphical presentation some network problems can be spotted without any additional effort. The effort that has to be put into the actual debugging is a different matter of course.

The second, and hopefully rarely used, part of the analysis is focused on debugging of network or network related problems. How the available information

and tools will be used in such a case depends entirely on the details of the problem that has to be debugged.
It is also possible to build an expert system which will provide hints or even complete solutions for certain types of problems. Of course, such a tool needs time to develop.

Chapter 2

Information to Collect

The number of data items (Object IDentifiers: OIDs) which are provided by a single device is extremely large (several million items). Even for the relatively detailed monitoring proposed, only a very small number of items (a few hundred to a few thousand) are required.

During normal operations it is sufficient to collect the information every 1 - 5 minutes. When a problem is being debugged, it should be possible to collect the information from a subset of devices every 1 - 10 seconds.

There are two basic classes of information which have to be collected via SNMP: static or descriptive information and dynamic or status information.

In order to be able to detect congestions in the devices, additional information which is not directly related to LHC traffic is needed. It should be possible to collect the necessary information from the interfaces which are on the same blade as the interface for LHC traffic.

2.1 Static Information

The static information is all the information that is needed to describe the device itself and physical layout of the interfaces of that device.

- Device type and operating system ("Cisco Internetwork Operating System Software IOS (tm) ...")
- Name, location and contact for the device ("Cernh7.cern.ch";"CERN, Geneva, Switzerland";""uuu@cern.ch, +41 22 xx xxxxx")

- Number of interfaces ("64")
- Information about the interfaces
  - Description ("TenGigabitEthernet";"Vlan")
  - Location inside the Chassis ("1/1")
  - Type ("ethernetCsmacd(6)";"propVirtual(53)")
  - MAC address
  - IP address if available
  - Destination IP address if available
  - Interface speed (via SNMP or description)

It is, of course, possible that this "static" information is being changed. Those changes are considered to be quite rare, such as once a month, or even once a year.

2.2 Dynamic Information

All dynamic information has to be obtainable via SNMP.
The information that should be collected is basically all interface counters for incoming and outgoing traffic.

- Bytecounts
- Unicast / Non-unicast Packets; Multicast / Broadcast Packets
- Errors
- Discards

Wherever possible the 64-bit counters should be used. This avoids ambiguities which automatically arise when using 32-bit counters with their inevitable wrap-arounds at 2^{32}byte (= 4GB) which is equivalent to less than 4 seconds on a fully utilized 10Gb link.

2.3 Additional Information

The additional information required to obtain a full and complete picture is the following:

- Layer2 tables
- Mapping vlan to real interface(s)

- Mapping port-channels/bundles to real interface(s)
- Possibly more

Some, if not all, of this additional information could be made available via SNMP, but then there has to be an agreement on the actual format of this information. The OID (defined in RFC2863) to be used to store additional information for each interface would be "IF-MIB::ifAlias.<interface>" which can contain a free format string of 64 byte length. The information has to be kept up to date at all times by hand, or scripts for that matter, because the information in this OID is not directly connected to the actual usage. This adds some managerial overhead for the person responsible for the device, but it could make complementary retrieval of configuration information unnecessary.

As an example a possible format for the representation of a port-channel of physical interfaces 11 and 22 into the virtual interface 30 is given. The standard OIDs would be:

- IF-MIB::ifDescr.11 = STRING: TenGigabitEthernet4/3
- IF-MIB::ifDescr.22 = STRING: TenGigabitEthernet7/2
- IF-MIB::ifDescr.30 = STRING: Port-channel1

The Alias information could be:

- IF-MIB::ifAlias.11 = STRING: IF of Po1,IF.30
- IF-MIB::ifAlias.22 = STRING: IF of Po1,IF.30
- IF-MIB::ifAlias.30 = STRING: Po with IF.11,IF.22 ; SPEED 20000 ; IP 123.45.67.89 ; TO IP 132.54.76.98

Chapter 3

Analysis

The collection of the information and its storage in databases are just the first stages. The most important part is, of course, the analysis of this information.

Most of the time the main form of analysis is the graphical presentation of the data flow. An in-depth look into the database will only be necessary in case of problems.

If all necessary information is available, the topology of the network can and should be generated dynamically. Together with the graphical display this will help to spot relatively simple problems, such as routing problems, without even having to dig into the database.

3.1 Graphical Representation

The graphical representation of the collected data is the most visible part of the analysis. In order to be useful it requires a complete understanding of the network topology.
There are already tools available which provide some of the functionality needed.

- MonaLISA (see [2]) - an entire monitoring suite in itself
- Animated Traffic Map software (see [3]) - a "simple" software for creating traffic maps

Ideally, the required additional functionality would be added to such a tool in order to minimize the effort.

The first level of representation should give a general overview over the layout of the network. As an example this would be something that MonaLISA

and the Animated Traffic Map software are capable of doing. All carriers (e.g. GEANT2) and NRENs should already be part of this view in order to clearly define the connectivity of the Tier 1 sites. The lines between should use a color-code for displaying the utilization of the respective link. In this simple view only single lines should be used, even if there are several links present. This is especially true for links to GEANT2. Figure 3.1 shows, in a very simple way, what the basic layout of such a general overview could look like.

The display for the next level of complexity should be reached by clicking on a particular "cloud" or link. In this view each link should be represented by a separate line. If a CERN router is displayed, all connections from this router should be shown even if it adds to the complexity of the plot. Figure 3.2 shows what such a view could look like.

The most detailed view should show all network devices on a particular link. If more LHC traffic is using a particular device it should be possible to display this traffic for completeness. Since such a view can become very complex, it should be an optional view. This detailed view should also show the actual layout of the interfaces on a network device and all information about the links, i.e. IP-addresses. All information about the device and the respective interfaces should be available. A possible layout of such a detailed view is shown in Figure 3.3.

All three example displays are very simple and somewhat crude and are just intended to show the principle layout of such a display. The real displays will have to have, for example, two color-coded lines per link, one line for incoming and one line for outgoing traffic.

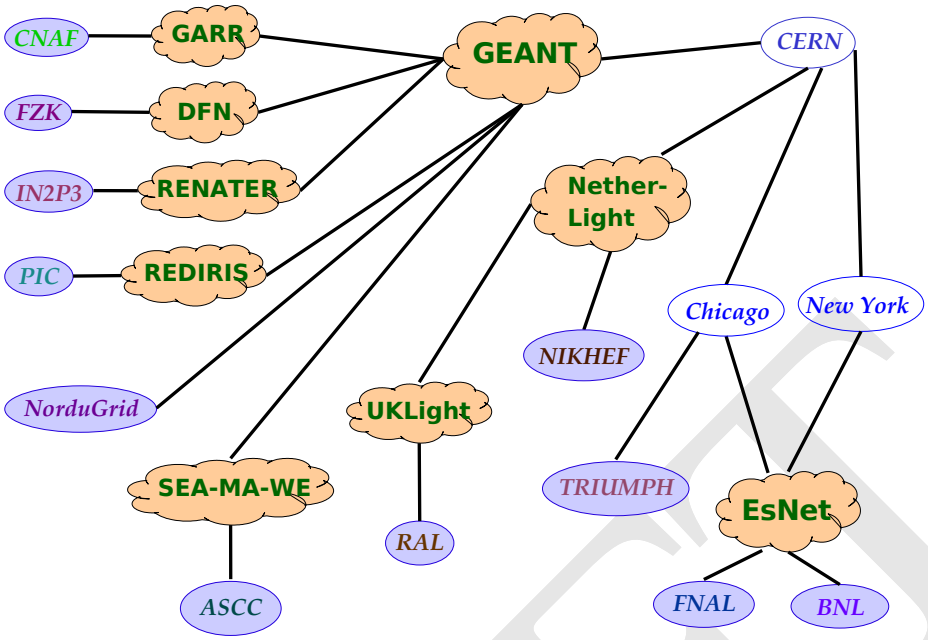

Figure 3.1: A simple example for a possible first level presentation of the network layout. It resembles fairly the view which is provided by MonaLISA. (This is NOT the actual network layout!)

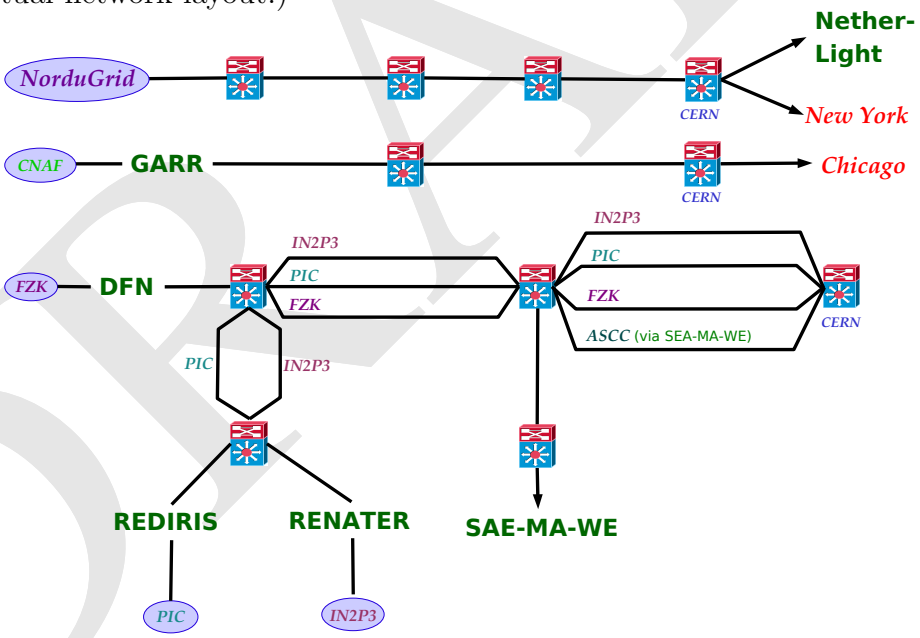

Figure 3.2: A possible second level presentation. This could be the layout of the GEANT infrastructure. Such a graphic is only reasonable if it does not become too complicated. Each separate link should be displayed.

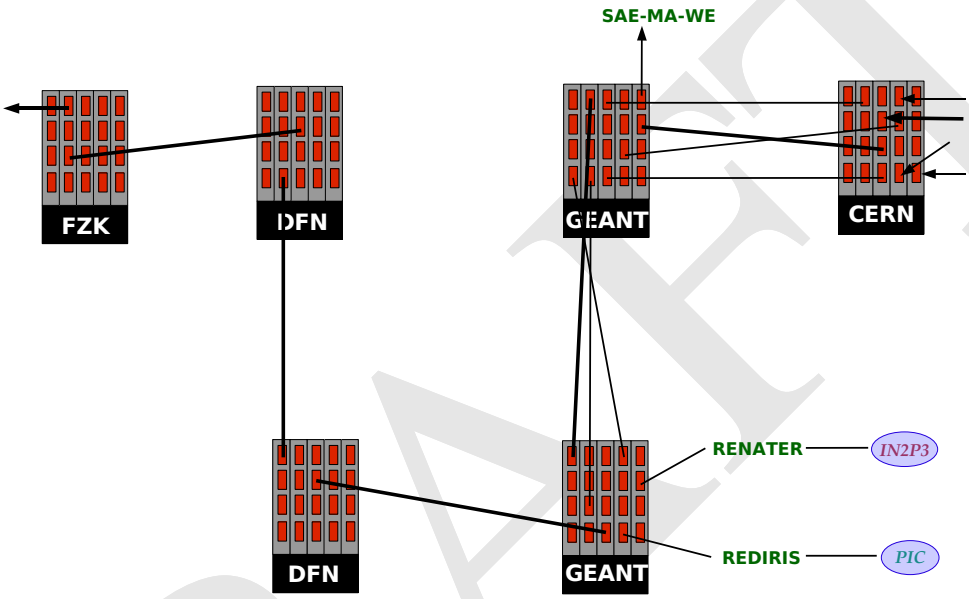

Figure 3.3: This could be the view of a particular link. Additional links for Tier 0 to Tier 1 traffic should also be displayed.

Chapter 4

Testing the Monitoring System

As soon as major parts of the network infrastructure and of the monitoring system are in place a testing phase should start. This testing phase will basically coincide with Service Challenge 4. Such a test phase has to be agreed with the persons responsible for grid services, because the tests will interfere with the Service Challenge. A thorough testing of the underlying network infrastructure is an integral part of the effort of creating a reliable service. The tests will help to gather experience with the monitoring system and its capabilities and to debug possible errors.

The carriers (i.e. GEANT2) and the NRENs should artificially "inject" problems into the network and it has to be checked how those problems appear in the monitoring and how long it takes to debug and fix them.
Almost any (non-trivial) problem the carriers have experienced in the past should be tested if time allows for this. The grid services should be running at all times if possible in order to check their response and to check how their monitoring can help to debug the problem.
If time allows for it, a "real life" test should be conducted, e.g. a simulated network problem without prior notification of the grid services.

There are simple tests where the "debugging capabilities" of the graphical display can be tested. This can be, for example, an asymmetric routing between two centres. If the graphical display of the topology is correct, such a problem could be spotted by simply looking at the more detailed display of the topology.

Annex A

Host-to-Host Monitoring

Since it is possible to monitor all network connections of a computer, it would be possible to map the entire data flow from mass-storage system to mass-storage system with relatively small additional effort. The data flow through the LANs would have to be mapped like the flow through the WAN, which should be trivial because the respective computing centres have already sophisticated monitoring tools in place. There are also monitoring facilities running on each machine in the computing centres. The only thing to be done there would be to add another metric which monitors the network connections of a machine.

Active Monitoring/Debugging

There might be non-trivial problems in the network (see [1]) where it is very helpful for the debugging to be able to inject test traffic at as many points in the network as possible. In the ideal case there would be at least one computer (source/sink of test traffic) connected to each router which can be remotely controlled from the network operation centre.

Bibliography

[1] A. Hirstius, "Service Challenge 1 and 2
The Wide-Area-Network point of view",
http://openlab-mu-internal.web.cern.ch/openlab-mu-internal/Documents/Reports/Technical/sc_network_monitoring.pdf

[2] http://monalisa.cacr.caltech.edu/

[3] http://loadrunner.uits.iu.edu/weathermaps/abilene/