ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
Τομέας Επικοινωνιών, Ηλεκτρονικής & Συστημάτων Πληροφορικής
Εργαστήριο Διαχείρισης και Βέλτιστου Σχεδιασμού Δικτύων - NETMODE

Ηρώων Πολυτεχνείου 9, Ζωγράφου, 157 80 Αθήνα, Τηλ: 210.772.1448
e-mail: maglaris@netmode.ntua.gr, URL: http://www.netmode.ntua.gr

ΔΙΑΧΕΙΡΙΣΗ ΔΙΚΤΥΩΝ
ΕΥΦΥΗ ΔΙΚΤΥΑ

Συμπληρωματικό Υλικό

Εναλλακτικά Εργαλεία Διαχείρισης

Καθηγητής Βασίλης Μάγκλαρης

Σύνταξη: Χρήστος Αργυρόπουλος, cargious@netmode.ntua.gr
Βασισμένο σε υλικά του εργαστηρίου του μαθήματος και δραστηριοτήτων του NETMODE

1. INTRODUCTION

Transport network monitoring includes in general two different fundamental aspects using real-time hardware/software solutions for:

- Traffic volume measurement: is used mainly for traffic surveillance and traffic shaping providing a database for statistics. The main goal is to record the average amount of traffic in the downlink and uplink directions as well as the total traffic for all nodes in the network. The data is averaged over a period of approximately five minutes.
- Flow identification and description: A flow is usually defined as a sequence of packets that share the same source and destination IP addresses, port numbers, and transport protocol. The information stored for each flow entry includes traffic volume (at the bytes and packets levels), port numbers, source and destination IP addresses Type of Service (ToS), input and output interfaces indices (as per SNMP MIB), timestamps for the beginning and end of flows [1]. Generally these flow descriptions are gathered in a central repository, or collector, with an average input rate of 2 Mbps.

While traffic volume measurement is restricted to the network level (network internal data), flow measurement includes detailed information about a certain number of x-tuple, such as source and destination IP addresses, IP transport protocol, source and destination ports (when the transport protocol is TCP or UDP), ingress interface, and ToS; another aspect is the deterministic sampling of packets that finalize a TCP session (i.e. either the FIN flag or the RST flag set).

Depending on the monitoring level (node, network, service) different information is typically measured, as shown in Table 1.

Monitoring level	**Typical measured characteristics**
Per network element	Overall load and per- traffic aggregate statistics
IP management system	Network-wide data, averages, trend analysis
Aggregate performance	Delay, jitter, packet loss, and available bandwidth in a network domain
Service level	Service Level Specification components

Table 1: Comparison of network monitoring levels [2]

In view of performance data related to a particular service quality support, aggregate level performance measurement means may also require less processing than performance estimation obtained by combining data from individual network elements. Generally, three different types of methods can be used for aggregate level performance assessment:

1. Active (intrusive) measurements
2. Passive (non-intrusive) measurements
3. Piggybacking measurements

Active measurement tools:

Active measurement includes methods that involve adding traffic to the network for the purposes of measurement. These methods are based on controlled transmission test traffic streams through the measured network. Such measurements can be implemented relatively straightforwardly, and thus have been used relatively early to assess the suitability of the Internet for transporting real-time streams

Typical examples of active measurement include ping, i.e. sending ICMP ECHO_REQUEST and capturing ECHO_REPLY messages, where only the sender needs to be under experiment control. It is used for measuring RTTs (Round Trip Times). One-way delay, on the other hand, is measured using a daemon running on the target, which listens for and records probe packets sent by the sender. Both the sender and the receiver are under control with a synchronized clock or other method to remove clock offset. Another publicly used tool is the so-called traceroute. It is used for determining the path (formed by IP addresses) the packets cross through, from a source to destination equipment. It uses the Time-To-Live (TTL) field within the IP header. A large-scale measurement system may use traceroute methods to discover network topology, but with some drawbacks given its blindness to IP addresses. Traceroute captures addresses without distinguishing between equipment or port addresses. In case of aliases, the topology discovery process keeps whole network parties in the dark, without correct resolution of aliases. Moreover, traceroute adds a considerable load to network traffic when attempting the topology discovery of a large-scale network.

Passive measurement tools:

Passive measurements do not require additional traffic to be injected into the operational network. Passive measurement methods capture the performance characteristics with reference to the traffic passing through the network as generated by users and applications. There are two basic ways of obtaining operational performance data from network elements:

1. Reading performance counters in the network element, i.e. polling. This includes both IP or link level characteristics and service-level characteristics. The data to be monitored can be packet statistics, traffic aggregates, flows, links, and then other service events. The palette of available characteristics may depend on the operating environment of the network element. For instance, in case of DiffServ the performance of complex per-flow operations at the edge of the network is monitored.

2. Reporting as predefined. This can be both periodic reporting and notification.

For instance, IGP objects such as OSPF LSA can be captured and processed to generate router graphs within an Autonomous System (AS), or Border Gateway Protocol (BGP) views, i.e. routing tables, even from a large set of ASs, can be collected by the Routeview repository tool. This is valid for a large set of AS-AS, router-router connections, even though route aggregation and filtering tend to hide some BGP connections.

Piggybacking:

Piggybacking is at the moment mostly a research-oriented approach, where time stamping and sequence numbering information are attached to traffic payload. Overheads from the measurement as well as required processing in the elements inserting and removing measurement fields to payload data packets need to be taken into account.

Method	**Passive measurements**	**Active measurements**	**Piggybacking**
Delay	Two measurement points	X	X
Delay variation	Single measurement point	X	
Packet loss	Single measurement point	X	X
Throughput	X	Indirect measurement	Indirect measurement

Table 2: Comparing measurement methods

Measurement for routing control:

Accurate information about traffic load in IP networks is important mainly because of two factors: (1) the non-connection-oriented nature of the Internet Protocol, and (2) the distributed nature of IP routing protocols.

The most important data for routing control are load levels on individual links and traffic aggregate performance characteristics, i.e. the measurement for routing table monitoring information in IP networks. The measurement includes methods such as:

- Performing two-point link or element monitoring with deriving information for routing
- Performing one-point monitoring supplemented with routing information
- Active measurements on the route

As a result of the two factors, traffic volumes on a particular link in a network domain may vary as a result of the interplay of BGPs and IGPs. Further potential reasons for traffic volume variability include burstiness of traffic sources, the operation of load-sharing schemes, and routing fallback to resiliency routes. Obtaining load information is also important as a basis for performing MPLS-based traffic engineering.

Traffic engineering tools for IP networks have been designed, using the IETF traffic engineering concepts as a framework. The traffic engineering process consists of configuration, monitoring, and optimization of network resources using a management system. Other tasks of IETF traffic engineering include capacity planning, routing control, and resource management.

In the following sections some of the existing monitoring tools are presented in order to compare functionalities and to investigate which are the most appropriate tools for the project's needs.

2. NAGIOS

Nagios is a host and service monitor designed to inform network managers of network problems. It has been designed to run under the Linux operating system. Nagios is licensed under the terms of the GNU General Public License Version 2 as published by the Free Software Foundation [3].

The monitoring daemon runs intermittent checks on hosts and services specified using external "plugins" which return status information to Nagios. When problems are encountered, the daemon can send notifications out to administrative contacts in a variety of different ways (e.g. email, instant message, SMS, etc.). Current status information, historical logs and histograms, reports, direct checks can all be accessed via a web browser.

Nagios has a lot of features, making it a powerful monitoring tool. Some of the major features are listed below:

- Monitoring of network services (SMTP, POP3, HTTP, FTP, NTP, PING, etc.)
- Monitoring of host resources (processor load, disk and memory usage, running processes, log files, etc.)
- Monitoring of environmental factors such as temperature and humidity
- Simple plug-in design that allows users to easily develop their own host and service checks
- Ability to define network host hierarchy, allowing detection of and distinction between hosts that are down and those that are unreachable
- Contact notifications when service or host problems occur and get resolved (via email, pager, or other user-defined method)
- Optional escalation of host and service notifications to different contact groups
- Ability to define event handlers to be run during service or host events for proactive problem resolution
- Support for implementing redundant and distributed monitoring servers
- External command interface that allows on-the-fly modifications to be made to the monitoring and notification behavior through the use of event handlers, the web interface, and third-party applications
- Retention of host and service status across program restarts
- Scheduled downtime for suppressing host and service notifications during periods of planned outages
- Ability to acknowledge problems via the web interface
- Authorization scheme that allows restricting what users can see and do from the web interface.

In the following subsections the most important characteristics that are especially relevant to autonomic functions and IPv6 are highlighted.

2.1 Nagios Architecture Overview

The main advantage of Nagios, in comparison with other network monitoring tools, lies in its modular structure. The Nagios core does not contain monitoring functions. Instead it uses external programs for service and host checks, which are known as *plug-ins* in Nagios service and network monitoring program. Hence Nagios package must be combined with Nagios plug-ins package for having a functional monitoring tool. Moreover, ready to use plug-ins are available, especially in the Nagios exchange platform, http://www.nagiosexchange.org/. The basic equipment already contains a number of standard plug-ins for the most important application cases. Special requests that go beyond these are answered by plug-ins that programmers can write themselves.
A plug-in is a simple program, often just a shell script (Bash, Perl etc.), that gives out one of the four possible conditions: OK, WARNING, CRITICAL, or UNKNOWN. These conditions can be mapped to configurable thresholds that network managers define according to the needs of provided network services.

This means that in principle Nagios can test everything that can be measured or counted electronically (e.g. temperature and humidity in the server room, amount of rainfall, presence of persons in a certain room at a time when nobody should enter it) There are no limits to this, provided that one can find a way of providing measurement data or events as information that can be evaluated by computer (for example, with a temperature and humidity sensor, an infrared sensor, etc.).

Nagios possesses a sophisticated, as well as an escalated notification system. The persons and the notification media are fully configurable per service, or per host, if that is needed. It has also the ability to notify different persons, based on a time scheduler.

The existence of a multiple-user web interface, with different user-access policies, makes Nagios functional. Nagios provides the administrators with a wide range of information, clearly arranged according to the issues involved. Whether the administrator needs a summary of the overall situation, a display of problematic services and hosts and the causes of network outages, or the status of entire groups of hosts or services, Nagios provides an individually structured information page for nearly every purpose. Through the Web front end, an administrator can inform colleagues upon accepting a particular problem so that they can concentrate on other things that have not yet been seen to.

The Nagiostats utility allows the user to graph various Nagios performance statistics over time using MRTG [4]. A snapshot of Nagios monitoring console and MRTG graphs can be seen in Figure 3-1. This can help to ensure that Nagios is operating efficiently. It can also be useful for locating problem areas in the monitoring process and observe the performance impacts of changes in your Nagios configuration. There are also more graphing tools. One of these is NagiosGrapher, an Open-Source tool which extends the monitoring software Nagios by automatically graphing performance data and embedding links of it into the Nagios web interface.

2.2 Influence of network topology on status and reach-ability determination

In medium and large-scale networks the monitoring of hosts and services and especially the immediate administrator's response depends on network topology knowledge. Nagios takes into account network topology, when it executes its tests for host reachability and services availability. When a service or a host check fails, it is not certain that the source of failure is the service or the host itself. As a matter of fact there are dependencies between nodes and services which can disorientate a network manager from the source of the occurred problem. Hence Nagios distinguishes the state of non-operational nodes services in two different states, unreachable and down. These are very different, although related states and can help you determine the root cause of network problems. Introducing parent/child relationships between hosts, Nagios achieves to enhance monitoring procedures. When a host-check fails, Nagios starts to check the parents-hosts. If a parent host is also non-operational, then Nagios characterizes the first host (child) as unreachable and the second host (parent) as down.

2.3 IPv6-enabled monitoring plug-ins

Some options are available in all plug-ins. The standard options of plug-ins are listed below [5]. The version of the IP protocol can be assigned as a parameter. The user can choose the version of the IP protocol with parameters "-4" or "-6" for IPv4 and IPv6 respectively. Plug-ins such as check_smtp, for SMTP monitoring, ckeck_imap, for IMAP server checks, check_dns, check_http, for http server checking, check_ssh, for monitoring Secure Shell servers can be executed using IPv6 packets.

Option flag	Description
-h	**Output of the online help**
-V	**Output of the plug-in version**
-v	**Output of additional information**
-H	**Host name or IP address of the target**
-t	**Timeout in seconds after which the plug-in will interrupt the operation and return the CRITICAL status**
-w	**Specifies the warning limit value**
-c	**Specifies the critical limit value**
-4	**Force IPv4 to be used**
-6	**Force IPv6 to be used**

2.4 Distributed Monitoring

Nagios can be configured to support distributed monitoring of network services and resources [1]. Generally speaking the main issue in a distributed monitoring environment is to offload the overhead (CPU usage, network load, etc.) of performing service checks from a "central" server onto one or more "distributed" servers and network devices. Most small to medium sized networks do not have a real need for setting up such an environment. However, when hundreds or even thousands of hosts (and several times that many services) must be monitored, using one integrated monitoring tool, like Nagios, this becomes quite important.

Central Server against Distributed Servers

When setting up a distributed monitoring environment with Nagios, there are differences in the way the central and distributed servers are configured. The function of a distributed server is to actively perform checks on all the services you define for a "cluster" of hosts. The term "cluster" basically means an arbitrary group of hosts on your network. Depending on your network layout, you may have several clusters at one physical location, or each cluster may be separated by a WAN, its own firewall, etc. The important thing to remember is that for each cluster of hosts (however you define that), there is one distributed server that runs Nagios and monitors the services on the hosts in the cluster. A distributed server is usually a basic installation of Nagios. It doesn't have to have the web interface installed, send out notifications, run event handler scripts, or do anything other than execute service checks. The purpose of the central server is to simply listen for service check results from one or more distributed servers. Even though services are occasionally actively checked from the central server, the active checks are only performed in extreme circumstances, so the central server only accepts passive checks for now. Since the central server is obtaining passive service check results from one or more distributed servers, it serves as the focal point for all monitoring logic (i.e. it sends out notifications, runs event handler scripts, determines host states, has the web interface installed, etc). The diagram below [1] reveals a general idea of how distributed monitoring works with Nagios.

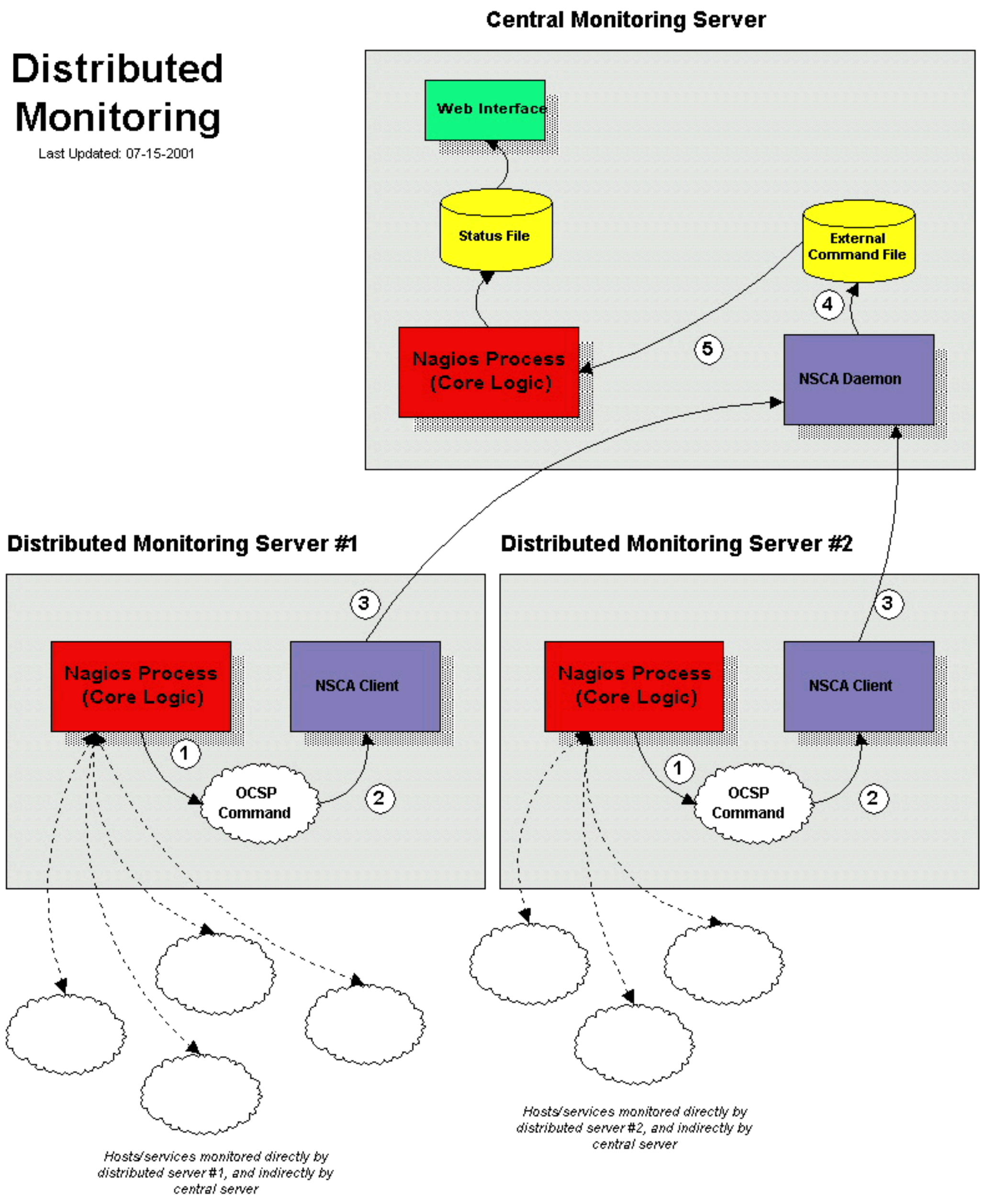
Distributed Monitoring
Last Updated: 07-15-2001
Central Monitoring Server
Web Interface
Status File
External Command File
4
5
Nagios Process (Core Logic)
NSCA Daemon
Distributed Monitoring Server #1
Distributed Monitoring Server #2
3
Nagios Process (Core Logic)
NSCA Client
1
OCSP Command
2
3
Nagios Process (Core Logic)
NSCA Client
1
OCSP Command
2
Hosts/services monitored directly by distributed server #1, and indirectly by central server
Hosts/services monitored directly by distributed server #2, and indirectly by central server

Figure 2-1: Distributed Monitoring

2.5 Integration Overview

Nagios is quite a popular monitoring application because it can be integrated in an existing infrastructure. There are several methods of integrating Nagios with the management software a network manager already uses. Also it can monitor almost any type of new or custom hardware, service, or application. In Figure 2-2 [1] we can see the integration points of Nagios. The functionality of the Nagios monitoring tool is based on its core logic. The manager is able to modify configuration files according to his monitoring, notification and performance needs.

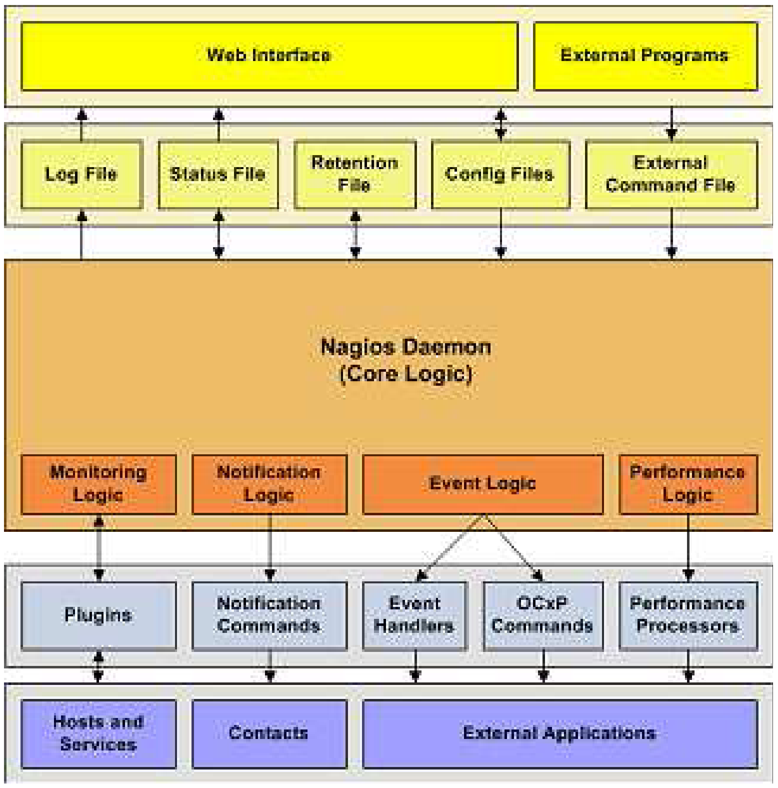

Figure 2-2: Nagios Overview

3. PERFSONAR

PerfSONAR [6], [9] is an infrastructure for network performance monitoring, making it easier to solve end-to-end performance problems on paths crossing several networks. It contains a set of services delivering performance measurements in a federated environment.

PerfSONAR is a services-oriented architecture, which means that the set of elementary functions have been isolated and can be provided by different entities called services. All those services communicate with each other using well-defined protocols.

The PerfSONAR infrastructure provides a multi-domain monitoring (MDM) service that answers the need for cross-domain monitoring capability. For the first time, network administrators are able to access network performance metrics from across multiple domains and can perform network monitoring actions in different network domains. Using out-of-the-box or customised web-interfaces, network problems and performance bottlenecks can be tracked and eliminated quickly, and potential performance issues can be identified and prevented before service disruption occurs.

PerfSONAR contains a set of services delivering performance measurements in a multi-domain environment. Network administrators are able to access performance metrics from different domains and can perform custom network monitoring actions. Using customised web-interfaces, PerfSONAR can track network problems and performance bottlenecks and prevent potential performance issues.

Services offered by PerfSONAR include [7], [8]:

- Measurement Point Service: The Measurement Point (MP) service is in charge of providing measurement data.
- Measurement Archive Service: A Measurement Archive (MA) service is used to publish historical monitoring data which are stored in an archive.
- Lookup Service: enables users to discover other services (MP, MA, Authentication service, etc.) and, in particular, the tools, capabilities, or data offered by those services.
- Authentication Service: Manages domain-level access to services via tokens.
- Topology Service: used to make network topology information available to the framework. Typically, visualization clients can make use of such information to retrieve the topology for the creation of network maps.
- Transformation Service: offers custom data manipulation of existing archived measurements
- Resource Protector Service: manages granular details regarding system resource consumption.

In addition to the above services, PerfSONAR also contains:

- RRD Measurement Archive (RRD MA): The service is a wrapper around Round Robin Databases, i.e. it makes data contained in database files available via the perfSONAR interface. Metrics that have been specified in this context are mainly utilization, but also interface input errors and output drops.
- SQL Measurement Archive (SQL MA): as an alternative to the RRD MA the SQL MA is a wrapper around SQL databases, such as mySQL, PostgreSQL. In addition to the metrics which have been defined for the RRD MA circuit status data can be made available via this MA.
- Command Line Measurement Point (CL MP): This service is used for executing command line tools, such as: ping, traceroute, owamp and bwctl.
- Telnet/Secure Shell Measurement Point (Telnet/SSH MP): this service provides an abstraction of commands that can be executed on Cisco and Juniper routers. It therefore enables queries to these routers like BGP commands or IP route commands. The name of the service related to the access to the routers via telnet or ssh.
- HADES Active Delay Evaluation System Measurement Archive (HADES MA): this service is an archiving service for active measurements of one-way delay, IP delay variation and packet loss

To address the objectives, a framework with three layers has been developed to separate measurements, management and visualization. Measurement Points from the lowest layer in the system (Measurement Point Layer) and are responsible for measuring network metrics. The measurements can be carried out using active or passive monitoring techniques, i.e. by the use of additional test packets or the monitoring of existing packets. The Service Layer is the middle layer of the framework and consists of administrative domains. The main responsibility is the exchange of measurement data and management information between domains. The User Interface Layer consists of visualization tools, which present measured data according to the needs of specific user groups with the ability to perform custom tests using the lower layers of the framework.

The PerfSONAR architecture is constructed as follows:

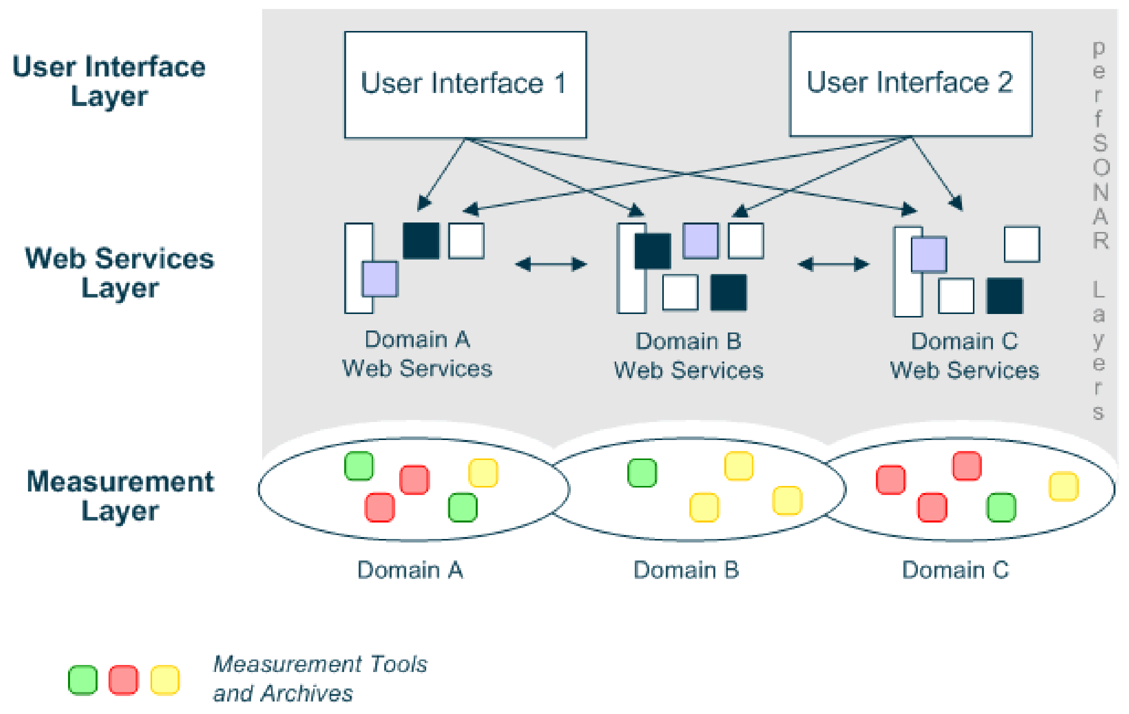

Figure 3-1: perfSONAR architecture

PerfSONAR also provides a range of open-source visualisation tools which allow users to query and view network end-to-end performance data in a graphical, user-friendly way. The most representative tools are:

- PerfsonarUI is designed and implemented as an open source, easy to use and yet powerful stand alone graphical user interface client capable of querying a range of PerfSONAR services deployed around the world (currently 18 interface utilization Measurement Archives and one IP Performance Metrics Measurement Archive). The primary target user groups for PerfsonarUI include NOC and PERT staff, as well as projects with demanding network performance requirements. End-users with some basic technical background can master the tool quite easily.
- Customer Network Management (CNM): The CNM tool shows the network topology using a set of hierarchical topology maps. The maps contain network nodes (routers or switches) and links including status information and metrics. CNM is a graphical, map-based interface which displays hierarchical maps of network topologies with performance metrics, including current and historic link utilisation and capacity for network maps and single interfaces.
- Hades Visualisation tool: The Hades visualisation tool is a web-based interface that presents IP Performance Metrics (IPPM – delay based) data.

The perfSONAR project is carried out by a consortium of organisations (GEANT 2, ESnet, Internet 2 and RNP). PerfSONAR framework has been applied for measurements in the European Research Network (GÉANT) and connected National Research and Education Networks in Europe as well as in the United States.

3. OPENNMS

Open source community aiming at filling the gap of an enterprise-grade network monitoring platform under the open source software model with General Public License (GPL) [13], has started OpenNMS development. In the following lines an architecture overview of Network Management Systems is deployed.
Scientifically, the core technologies that are central to OpenNMS' design act to fulfill the requirements of what is called a "management station" [10]
The term management station (MS) was first defined in paragraph 3 of RFC 1157, "SNMP Architecture" and, again, more recently, in paragraph 2 of RFC 1901, "Components of the SNMPv2 Framework". The industry has since coined the term NMS, or Network Management System, hence OpenNMS. An integrated network management system is shown in Figure 3-2, constituting the heart of a Network Operation Center (NOC). Since the publication of these RFCs in 1990 and 1996, respectively, internet network (internet) and related technologies have improved, adapted, and grown in ways that were never anticipated. Yet the fundamental architectures of SNMP and the underlying transport, TCP/IP, have not significantly changed from their original designs. These designs have stood the test of time by sustaining the tremendous growth of IT and the network management problem as we see it today. No truer testament can be awarded to the vision set forth by the designers in their original specifications. The developers of OpenNMS are continuously crafting a better and more robust MS based on these transport technologies. The goal of the project’s community of scientists is to provide users with quality software that gives them the ability to manage any advanced application of technology today and in the future.
There are two network management models that are fundamental to the implementation of a MS, such as OpenNMS, that were also introduced in 1996. The International Telecommunication Union (ITU) published the Telecommunications Management Network (TMN) model as a design for carriers to manage service delivery. Then, in 1997, the ITU published the FCAPS model for comprehensive management of an organization's technology assets. These models are still used today by network management professionals as a guide for designing comprehensive monitoring solutions. The OpenNMS maintainers also use these references as milestones toward achieving the project’s goals and continued improvement of the software.
TMN is a model for managing "Open Systems" within a telecommunications network and defines four logical layers: Business Management (BM), Service Management (SM), Network Management (NM), and Element Management (EM). FCAPS is an acronym for a model that further defines NM using the terms:

- Fault Management (FM)
- Configuration Management (CM)
- Accounting (A)
- Performance Management (PM)
- Security (S).

Consequently, these two models are also often used as a checklist when evaluating and choosing a MS. A quick evaluation of OpenNMS will find that it provides features that support the SM and NM logical layers of the TMN model and each component of the FCAPS model with an emphasis on FM and PM.
OpenNMS provides extremely comprehensive FM. Faults in OpenNMS are detected via three distinct and separate mechanisms: service polling, receipt of unsolicited messages

(typically SNMP traps), and thresholds evaluated against performance data. OpenNMS also provides extremely comprehensive PM via several mechanisms that are based on a robust data gathering API called the Service Collector Interface. Current implementations of the Service Collector, SNMP, JMX, HTTP and NSClient, gather data that can then be utilized in performance graphs, thresholds, and TopN analysis.
The remaining components of the FCAPS model: CM, A, and S, are each addressed, to some degree, within OpenNMS. However, they are not implemented as comprehensively as defined in the model or as dictated by de-facto standards established by software that focuses on each these components. For example, the A component is partially implemented by providing usage data for the network and system resources such as bandwidth, CPU, and disk usage. While this data isn't directly exposed to customer billing systems via the WebUI or an API, the data can be exported for use in these billing systems. An NMS, supporting component A, should synchronize with the company's provisioning system so that customer records can be easily associated with the resource's management data. OpenNMS provides this capability via one of the enterprise integration features available in OpenNMS, the Model Importer Service.
The C component of FCAPS is also partially implemented through the use of OpenNMS' asset records and with its ability to shut down or turn on network interfaces via the WebUI. OpenNMS partially implements the S component by providing support for SNMPv3, user based access control to the WebUI via local or LDAP security models, and with its integration to the intrusion detection and vulnerability assessment systems, Snort and Nessus [11].

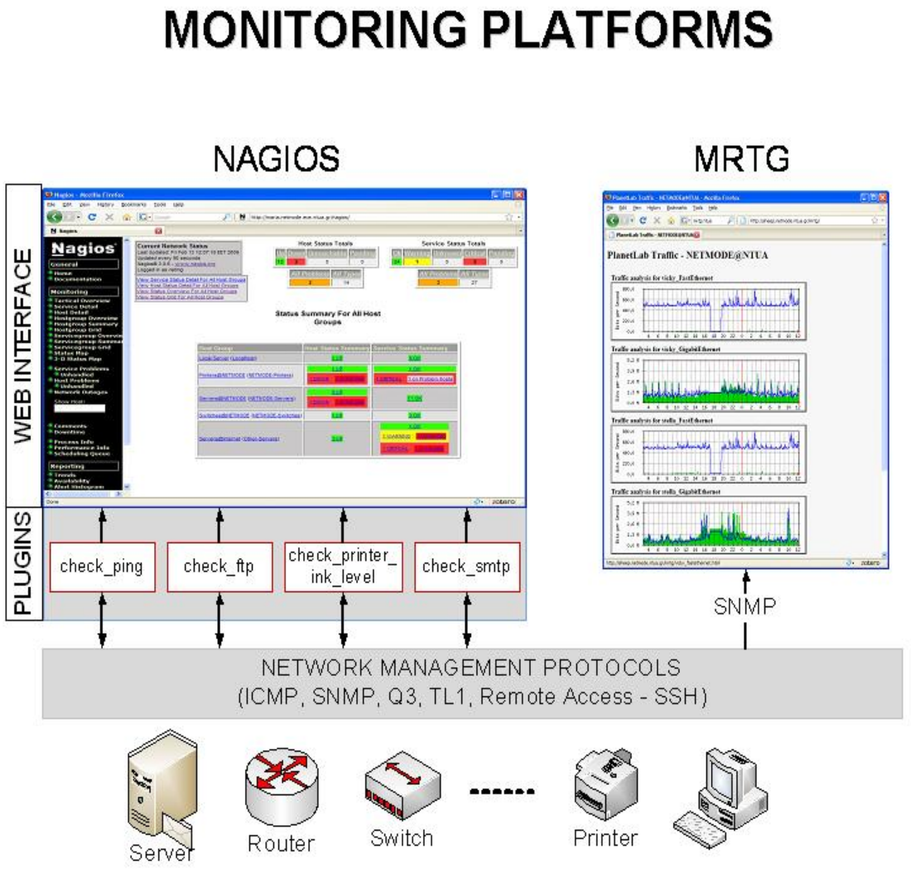

Figure 3-1 Open Source Monitoring Platforms

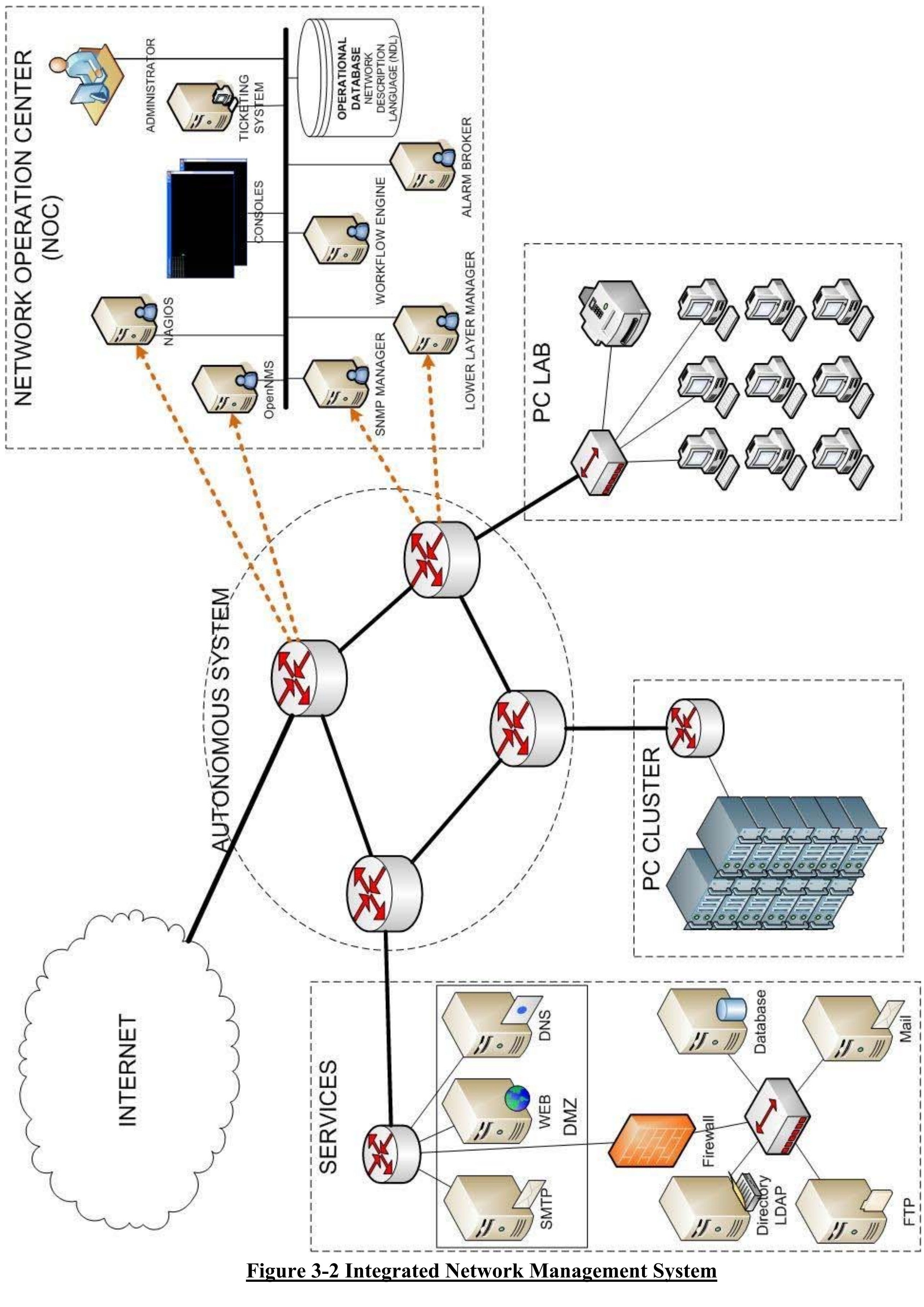

Figure 3-2 Integrated Network Management System

4. REFERENCES

[1] Evaluation of Candidate Protocols for IP Flow Information Export (IPFIX) - http://www.ietf.org/rfc/rfc3955.txt
[2] Implementing Service Quality in IP Networks, Vilho Raisanen, Chichester, West Sussex, England ; Hoboken, NJ : John Wiley & Sons, Ltd. (UK), 2003
[3] Nagios Version 3.x Documentation - http://www.nagios.org/docs/
[4] http://oss.oetiker.ch/mrtg/
[5] Wolfgang Barth, "Nagios – System and Network Monitoring", Published by Open Source Press GmbH, Munich, Germany, 2005
[6] "PerfSONAR: A Service-Oriented Architecture for Multi-Domain Network Monitoring" accepted by ICSOC (International Conference on Service Oriented Computing) for the December 13-15, 2005 Conference
[7] http://www.geant2.net/server/show/nav.1801
[8] http://www.perfsonar.net/description.html
[9] http://wiki.perfsonar.net/jra1-wiki/index.php/Main_Page
[10] White Paper: Project OpenNMS, David Hustace - http://www.opennms.org/images/a/a8/Project_opennms-abstract-v2.pdf
[11] http://www.opennms.org/index.php/Main_Page
[12] http://en.wikipedia.org/wiki/Comparison_of_network_monitoring_systems
[13] http://en.wikipedia.org/wiki/GNU_General_Public_License