Int. J. Communications, Network and System Sciences, 2009, 6, 528-539
doi:10.4236/ijcns.2009.26058 Published Online September 2009 (http://www.SciRP.org/journal/ijcns/).

A Scalable Architecture for Network Traffic Monitoring and Analysis Using Free Open Source Software

Olatunde ABIONA[1], Temitope ALADESANMI[2], Clement ONIME[3], Adeniran OLUWARANTI[4], Ayodeji OLUWATOPE[5], Olakanmi ADEWARA[6], Tricha ANJALI[7], Lawrence KEHINDE[8]

[1]*Department of Computer Information Systems, Indiana University Northwest, Garry, USA*
[2,6] *Information Technology and Communications Unit, Obafemi Awolowo University, Nigeria*
[3]*Information and Communication Technology Section, Abdus Salam International Centre for Theoretical Physics, Trieste, Italy*
[4,5]*Department of Computer Science and Engineering, Obafemi Awolowo University, Ile-Ife, Nigeria*
[7]*Electrical and Computer Engineering Department, Illinois Institute of Technology, Chicago, USA*
[8]*Department of Engineering Technologies, Texas Southern University, Houston, Texas, USA*
Email: [1]*oabiona@iun.edu*, {[2]*taladesanmi*,[4]*aranti*,[5]*aoluwato*,[6]*kanmiade*}*@oauife.edu.ng*, [3]*onime@ictp.it*, [7]*tricha@ece.iit.edu*, [8]*kehindelo@tsu.edu*

Received February 5, 2009; *revised April* 8, 2009; *accepted June* 10, 2009

ABSTRACT

The lack of current network dynamics studies that evaluate the effects of new application and protocol deployment or long-term studies that observe the effect of incremental changes on the Internet, and the change in the overall stability of the Internet under various conditions and threats has made network monitoring challenging. A good understanding of the nature and type of network traffic is the key to solving congestion problems. In this paper we describe the architecture and implementation of a scalable network traffic monitoring and analysis system. The gigabit interface on the monitoring system was configured to capture network traffic and the Multi Router Traffic Grapher (MRTG) and Webalizer produces graphical and detailed traffic analysis. This system is in use at the Obafemi Awolowo University, Ile-Ife, Nigeria; we describe how this system can be replicated in another environment.

Keywords: Scalable Network Monitoring, Traffic Analysis, Web Log Analysis, Open Source

1. Introduction

The rapid growth of the Internet in size, complexity and traffic types has made network management a challenging task. The ability of a monitoring system to provide accurate information about the nature and type of the network traffic can not be over emphasized. Information about who is generating the most traffic, what protocols are in use, where is the traffic originating from or where is the destination of the traffic can be very important to solving congestion problems. Many network administrators spend a lot of time, trying to know what is degrading the performance of their network.

A typical solution to congestion problem is to upgrade network infrastructure, i.e. replace servers with high end servers and increase the bandwidth. This solution is expensive, short term and does not scale. As soon as the upgrade is done the congestion problem will improve for a while and later gradually deteriorate as the users change their behavior in response to the upgrade. The alternative solution to this problem is to deploy a scalable network traffic monitoring and analysis system, in order to understand the dynamics of the traffic and changes in the internet and overall stability of the network. In addition to knowing the health status of the network, monitoring of network activity also has the benefits of detecting denial of service (DoS) and bandwidth theft attacks. In order to conduct analysis of wide range of network behaviors, it is necessary to collect network traffic on a continuous basis rather than as a one time event which only captures transient behaviors that provides insight into network problems. Collecting long term network traffic data will provide valuable information for improving and understanding the actual network dynamics.

The rest of the paper is organized as follows: in Section 2, we review related work, followed by the system design in Section 3. In Section 4 we describe the implementation and configuration of the system. In Section 5, we present the example application of the system and in Section 6 we conclude the paper.

2. Related Work

The libpcap [1] tool has greatly simplified the task of acquiring network packets for measurement. The limitation of the tool is its inability to analyze the captured data, it will only capture the data and the programmer or network administrator is left to carry out analysis manually. This task can be time consuming and cumbersome and in most cases accurate information about the network is not obtained. Some researchers have developed modular software architectures for extensible system [2,3], however only a few of these systems are optimized to handle large amount of data and continuous monitoring. Other researchers have developed systems for streaming data through protocol layers [4] and routing functions [5], but not much attention has been given to the analysis of large/huge or broad data collected over time.

Simple Network Management Protocol (SNMP) covers a class of tools such as Multi Router Traffic Grapher (MRTG) and Cricket [6], which collect counter statistics from network infrastructures, and visualizes these statistics by means of graphs. The most common use for these tools is graphing the InOctet and OutOctet counters on router interfaces, which respectively provide counts of the number of bytes passing in and out of the interface. Many other tools also support SNMP monitoring [7]. This includes monitoring performed by Remote Monitoring (RMON) agents, which can be useful in determining the top hosts with regards to traffic, as well as the distribution of packet sizes on a network. SNMP can only be used to monitor devices that are SNMP managed, the reading and writing of the In Octet and OutOctet counters in a router could generate substantial traffic. Tcpdump [8] prints out the headers of packets on a network interface that match the boolean expression. It is a network sniffer with in-built filtering capabilities; it can only collect the data from the network, but does not analyze collected data. The collected data can be analyzed offline with another utility namely, tcpshow and tcptrace. As useful and powerful as tcpdump is, it is only suitable for troubleshooting i.e., for tracking network and protocol related connectivity problems.

MRTG is a versatile tool for graphing network data [9], this tool can run on a Web server. Every five minutes, it reads the inbound and outbound octet counter of the gateway router, and then logs the data to generate graphs for web pages. These graphs can be viewed using a web browser. Although MRTG gives a graphical overview, it however does not give details about the host and protocol responsible for the traffic monitored. Windmill [10] is a modular system for monitoring network protocol events; it is useful for acquiring the data from the network, but it is however limited in its capability by not providing any facility to aid in the analysis of those events or non protocol events acquired. WebTrafMon [11] uses a probe to extract data from network packets and composes log files. Analysis results are based on the collected log files. Furthermore the user is able to view the analysis result via a generic web browser. WebTrafMon can show traffic information according to the source and destination host through any web interface; it can also show the traffic status according to each protocol in use. Although WebTrafMon has good capabilities, it can not monitor and analyze traffic in a switched network such as Fast Ethernet and Gigabit Ethernet.

The continuous Query Systems such as [12] and [13] share many of the concerns of other systems in acquiring and filtering continuous streams of data, this system however lack the ability to easily add new functions over the data, hence they are not extensible. The agile and scalable analysis of network events is a system based on modular analysis and continuous queries [14] allowing users to assemble modules into an efficient system for analyzing multiple types of streaming data. The system is optimized for analyzing and filtering large streams of data, and makes extensive use of polymorphic components that can perform common functions on new and unforeseen types of data without requiring any additional programming. This data analysis system provides a scalable, flexible system for composing ad-hoc analyses of high speed streaming data but does not provide infrastructures for data gathering and network monitoring.

Ntop [15] provides a display similar to the UNIX top command, but for network traffic, it can be used for traffic measurement and monitoring. Features such as the embedded HTTP server, support for various network media types, light CPU utilization, portability across various platforms, and storage of traffic information into an SQL database makes ntop versatile. However, ntop is limited by its high memory requirements when operating in a continuous monitoring environment. The extensive cache usage has the drawback that memory usage is increased [16]. This makes ntop a memory and computational intensive application. Like several of the other tools, ntop uses the same packet capture library to obtain the network data. Since ntop operates in continuous mode, it is designed to operate on networks with speeds of less than 100 Mbps [17]. A continuous network tracing infrastructure was described in [18], it is multi-user based and capable of collecting archives and analyzing network data captured. The system is limited by not providing a universal web interface to display the results of its monitoring. The system requires huge storage, 11TB of shared disk space for data repository since it is not web based and also there is high overhead since the storage system is accessed through an NFS server over an Ethernet link.

To understand network dynamics, and to analyze a wide range of network behaviors, network data needs to be collected over a long period of time, rather than as a one time event to capture transient behaviors that provide insight into immediate network problems. In this paper we describe the development and implementation of a scalable, passive and continuous network monitoring system capable of collecting, archives, and analyze network traffic.

3. System Design

3.1. The Obafemi Awolowo University Network

The Obafemi Awolowo University Network (OAUnet), was established in the Obafemi Awolowo University, Ile-Ife, Nigeria and began operation in June, 1996. OAUnet was an initiative of the Academic Computer Networking Project of the International Centre for Theoretical Physics (ICTP) (now Abdul Salam International Centre for Theoretical Physics in Trieste, Italy) for developing countries. It is a campus wide intra academic network with a gateway to the INTERNET. In the last one decade of its existence, OAUnet has grown to be one of the largest intra academic networks among the Nigerian universities. The development of the OAUnet from a network of three workstations with an e-mail gateway to the Internet now to a network of about one thousand nodes serving close to 30,000 staff and students population can be distinctly fused into four phases over a period of ten years.

3.1.1. Present Status (2005 Till Date)

OAUnet currently connects to the Internet on 6Mbps downlink and 1.5Mbps uplink bandwidth. The Intranet is made up of four radio connections running at 3Mbps to the University collaborating research centres and 9 student information centres/cafes located in/around the student halls of residences. The main academic subnet comprising of two colleges and thirteen faculties linked together by over 2Gbps Fibre connections. The network serves a population of close to 26,500 students and 3,000 academic, administrative and technical staff. The network uses TCP/IP protocol and all the server namely web, mail, proxy, firewall, authentication, DNS and core routers runs on Linux operating system (a mixture of Redhat and Mandrivia). The network topology is hybrid, a sort of hirachical star, a logical layout of the network is shown in Figure 1. Recently, wifi/hotspot service have been deployed in the university to extend network access to places which hitherto has no netwotk presence such as the staff quarters.

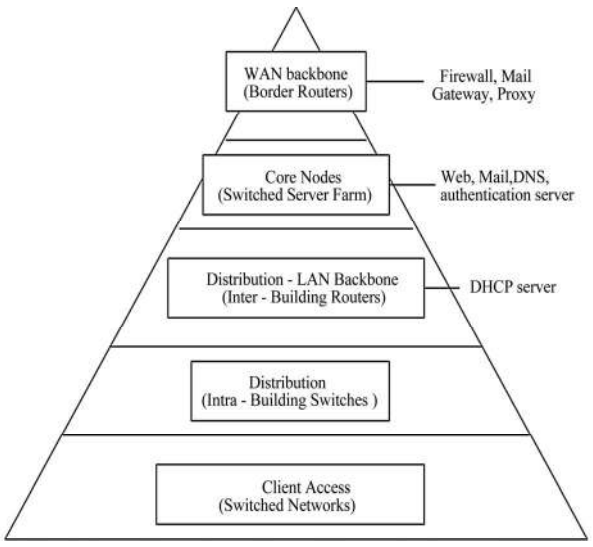

WAN backbone (Border Routers)
Firewall, Mail Gateway, Proxy
Core Nodes (Switched Server Farm)
Web, Mail,DNS, authentication server
Distribution - LAN Backbone (Inter - Building Routers)
DHCP server
Distribution (Intra - Building Switches)
Client Access (Switched Networks)

Figure 1. A logical layout of the OAUnet.

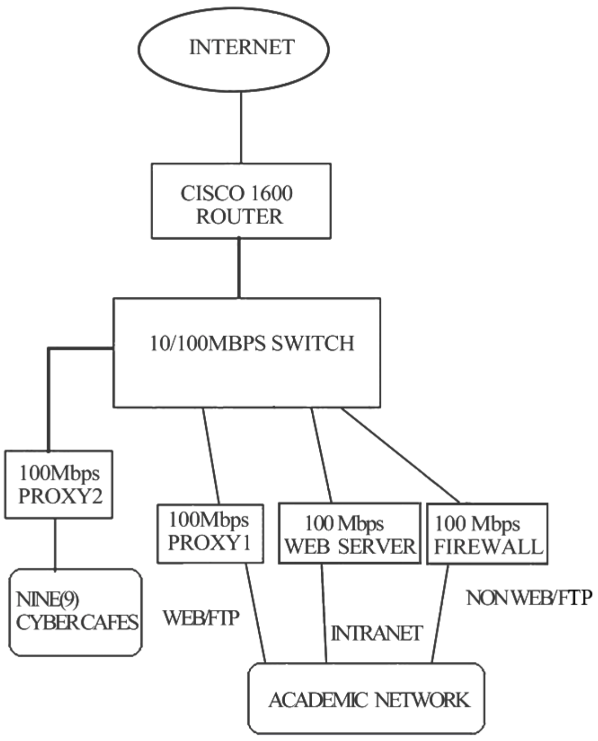


Figure 2. OAUnet internet link.

3.2. The Network Monitoring System

The scalable network monitoring and analysis tool com-

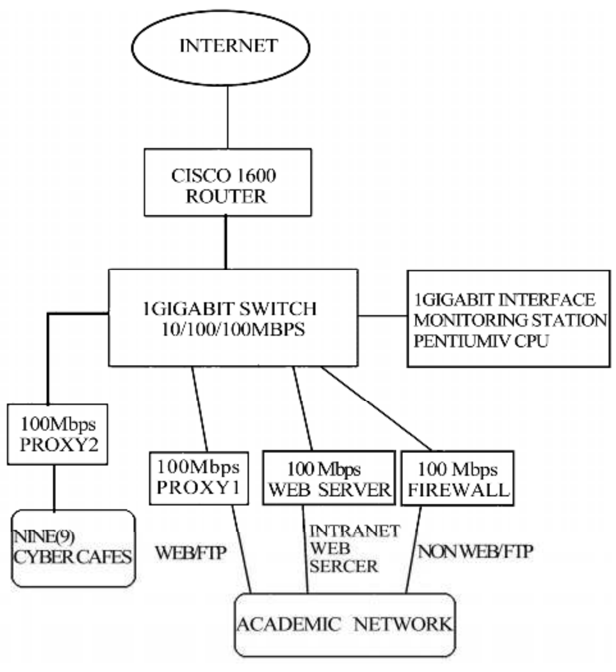


Figure 3. Scalable traffic monitoring setup.

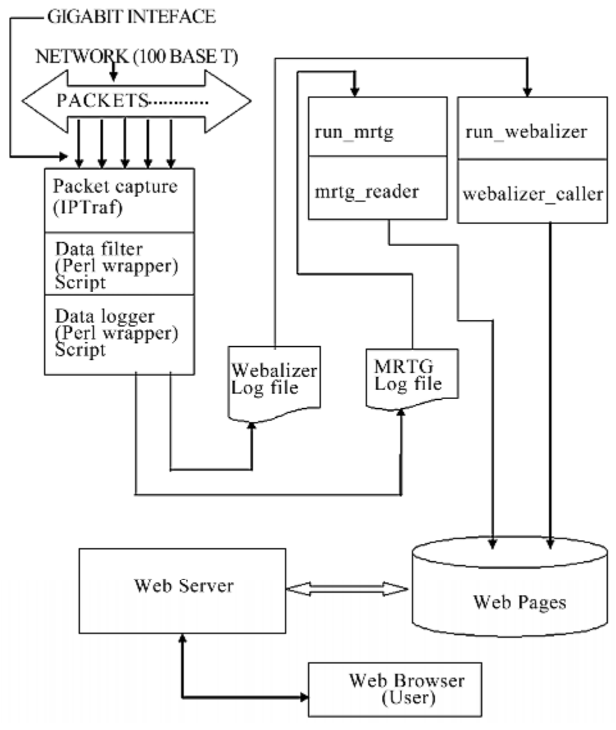


Figure 4. Scalable traffic monitoring architecture.

prises of hardware and software. The hardware component includes a Pentium 4 central processing unit, a gigabit switch, and a gigabit network interface card. The software components are free open source programs namely; Linux operating system, IP traffic monitoring program (IPTraf), MRTG, Webalizer and Perl script.

3.2.1. Hardware Setup

The Obafemi Awolowo University Network Monitoring experiment was conducted with the aim of monitoring live network without adversely imparting on the performance and also to identify and monitor traffic patterns (both speed and volume) on the basis of host (IP address), Protocol and time of the day. For this experiment we used open source software for the reason of cost and ease of modification. This work builds on an earlier work in [19]. On a high-speed network, the number of packets transmitted per second can be astronomical. Analyzing such data requires a great deal of processing time. Thus the efficiency of a packet capturing code is essential [11]. As a matter of fact, ensuring that all packets are captured is extremely challenging [20]. The original design of the LAN to INTERNET is shown in Figure 2. There are two separate networks sharing the INTERNET link (cyber cafes and academic network). The academic network comprises proxy 1, web server, and firewall. While the Nine (9) cyber cafes use proxy 2 to access the INTERNET. For the academic network, proxy 1 provides web/ftp services, firewall provides DNS/E-mail and other non-web services and the web server provides intranet facilities and also hosts the university web site. All were connected via fast Ethernet switch. A major constraint was that this design could not be changed or modified and it was not possible to install any application on the servers under the agreement reached with network/system administrators for conducting this experiment.

The proposal was to setup a single monitoring station that will monitor all the traffic. From Figure 2, the network has four (4) key servers transmitting and receiving signals at 100Mbps. In order to effectively capture all the packets from all the four (4) servers, the monitoring station must operate at 1gigabit speed. In Figure 3 the fast Ethernet switch in Figure 2, was replaced with a switch having two of its ports running at 1Gbps and also having port mirroring facility. The port mirroring facility allows concatenation of traffic from several ports to one of the gigabit ports where the monitoring station is attached. This implies that all traffic arriving at the ports for the four (4) OAUnet servers were also copied/mirrored to the gigabit port.

3.2.2. Software Setup

Figure 4 shows the architecture of the scalable network traffic monitoring system. The monitoring system runs on the Linux operating system. Monitoring was done using the open source IPTraf software [21]. A wrapper script was used to start the IPTraf program with suitable command line arguments to sort the output of IPTraf based on IP address and protocol and it creates suitable inputs for MRTG [22] and Webalizer [23]. After starting up IPTraf and initializing some variables, the Perl wrapper script is responsible for processing the summary information from IPTraf into suitable log formats both for MRTG and Webalizer. The software aims at doing the processing very fast and as such it is not complicated. In order to maintain system stability and minimize memory leaks, the software restarts every 30 minutes. Other scripts used are *run_mrtg, mrtg_reader*, *run_webalizer* and *webalizer_caller.*

A copy of the scripts used may be downloaded from the URLs http://www.ictp.it/~abionao. The scalable traffic monitoring and analysis system also runs a cron job that copies proxy log from proxy servers on the network every six hours. The log files were analyzed using a Webalizer configuration file; the result of the proxy log analysis is available through a web interface.

4. Implementation and Configuration

4.1. Implementation

A simple Perl script, *run_mrtg* was used to run MRTG. It basically checked if a MRTG configuration file and the right directories exist for the monitored host or protocol or create them if they do not exist. This script was run via a suitable cron job entry. A second script *mrtg_reader* was used to further adapt and present the output of the IPTraf wrapper in a format suitable to MRTG. MRTG requires input to be in log format, Figures 5 and 6 shows the flow chart for the wrapper script and *run_mrtg*.

Another Perl script *run_webalizer* was used to check (or create) suitable configurations file for Webalizer and also check that the right output directories exist and are accessible. This was also called in a cron job entry. The number of hosts or protocols to monitor was specified through the use of a central text file that holds a list of IP addresses and protocol names. Figure 7 shows the flow chart for *run_webalizer*. The scalable network monitoring system is capable of monitoring the Intranet and the Internet link simultaneously; all that is required to achieve this is the provision of additional network interface card installed in the monitoring system.

4.2. Installation and Configuration

A user is advised to choose names that are convenient for his network setup as it is the standard practice. In this write up the settings used are specific to our case study but can also be replicated. A copy of the Perl scripts may be downloaded from the URLs http://users.ictp.it/~abon-

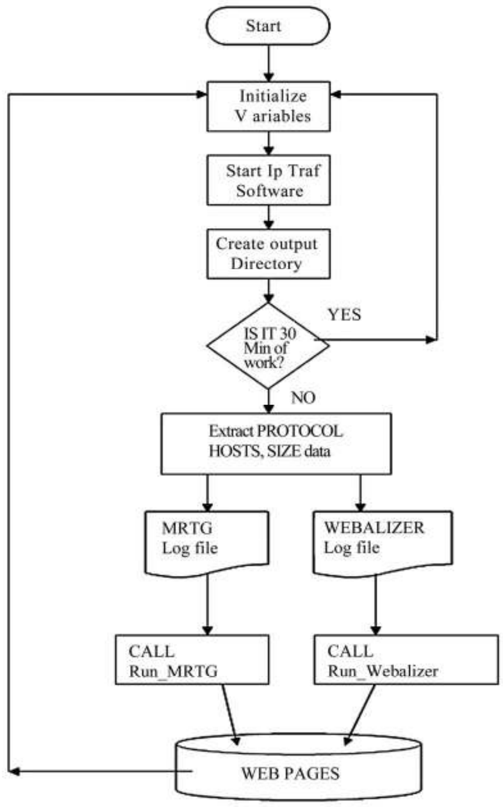


Figure 5. Wrapper script.

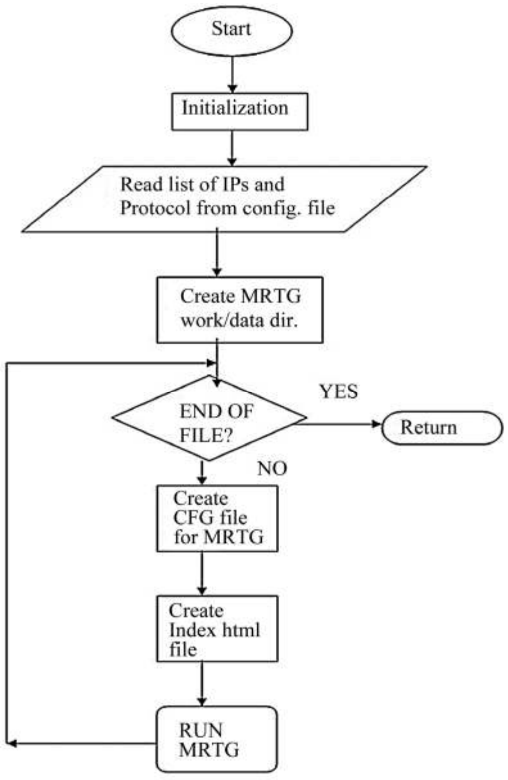


Figure 6. Run_mrtg script.

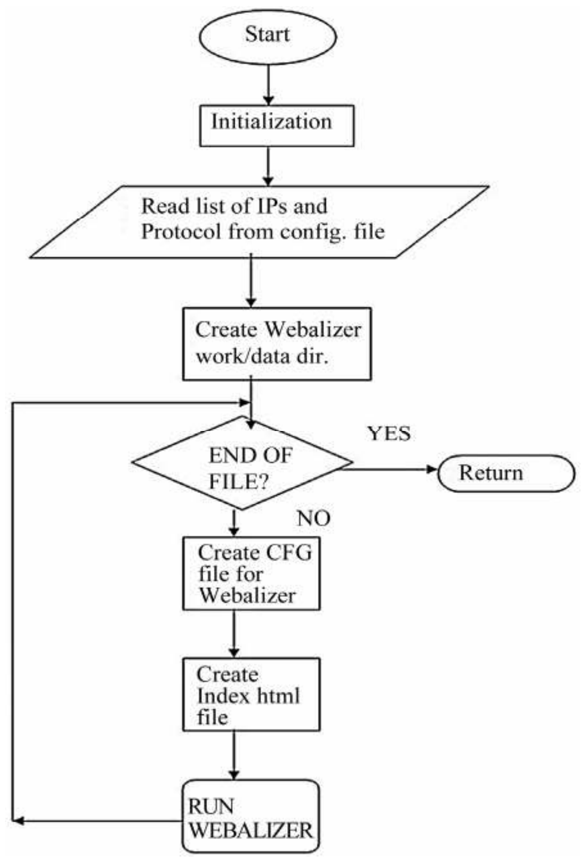

Start
Initialization
Read list of IPs and Protocol from config. file
Create Webalizer work/data dir.
END OF FILE?
YES
Return
NO
Create CFG file for Webalizer
Create Index html file
RUN WEBALIZER

Figure 7. Run_Webalizer script.

ao/ and copied to the directory *oaunetmon* in */usr/local*. Run the dos2unix command on each of the Perl script to convert them to UNIX program. Since the monitoring tool will process several packets, it is necessary to create a file containing the list of hosts and protocol on the network of interest. Using a text editor, type the IP addresses and protocols of interest in the file *OAUNETMON* in */etc* (vi */etc/OAUNETMON*), sample content of the file shown below.

Sample entry for OAUNETMON

```
29.20.23.1
29.20.23.2
29.20.23.8
http
domain
ftp
https
smtp
```

Make a cron job entry in */etc/cron.d* for the following files: *Asoju_proxy, Café_proxy, oaunetmon_mrtg* and oaunetmon_*webalizer*. This will enable the programs to run automatically when the system boots up or when restarted. Below are samples entries in the files. Users are advised to use valid IP address and correct path to the log files.

Sample Cron job entry

```
            Asoju_proxy
0 20,22,4,10,16 * * * root      /etc/asoju_squid_logs

            Café_proxy
0 20,22,4,10,16 * * * root      /etc/cafe_squid_logs

            oaunetmon_mrtg
*/5 * * * * root        /usr/local/oaunetmon/mrtg_reader.pl
/tmon/IPTRAF/mrtg/global/     /tmon/IPTRAF/mrtg/

*/15 * * * *    root        /usr/local/oaunetmon/run_mrtg.pl
/tmon/IPTRAF/mrtg
/logf/www/html/oaunetmon/mrtg
/logf/CONFIGS/oaunetmon/mrtg/

            oaunetmon_webalizer
5 */4    * * * root   /usr/local/oaunetmon/run_webalizer.pl
/tmon/IPTRAF/Webalizer/
/logf/www/html/oaunetmon/webalizer
/logf/CONFIGS/oaunetmon/webalizer/
```

To start the Perl wrapper script (iptraf.pl), the monitoring interface should be placed in promiscuous mode. In this mode, the monitoring interface will see all packets flowing in the network. This will enable it monitor all packets and then starts the traffic monitoring program automatically. The file rc.local in */etc/rc.d/* will have the following content*: /etc/rc.d/rc.local*,

Sample entry for rc.local

```
#!/bin/sh
/sbin/ifconfig eth0 promisc
/usr/local/oaunetmon/iptraf.pl /tmon/IPTRAF//tmon/
    iptraf   -i   qll    &touch

  /var/lock/subsys/local
```

The scalable monitoring system is also capable of analyzing proxy server log files. Log files were copied from the two proxy servers every six hours, the access logs were analyzed using a Webalizer configuration file and the results of the analysis can be viewed on a web browser. The results of traffic analysis combined with the proxy log analysis are useful for optimizing bandwidth usage. Below are sample entries for the script used to copy the proxy logs (asoju_squid_logs and café_squid_logs); the file resides in */etc* directory, and is called by a cron job (batch file). */etc/asoju_squid_logs* and */etc/café_squid_logs*.

Asoju_squid_logs Script

```
#!/bin/sh
PATH =   $ {PATH}: /usr/bin/
export   PATH
/usr/bin/rsync        –a        –e              ssh          root
@192.168.0.5:/var/log/squid/access.log    /logf/squid_log/
[ $?    -gt    0]    $$ exit 0
[-S    /logf/squid_log/access.log ] | | exit 0
/usr/bin/webalizer      -C      /etc/squid_webalizer.conf      >
/dev/null    2    > &1
```

café_squid_logs Script

```
#!/bin/sh
PATH =   $ {PATH}: /usr/bin/
export   PATH
/usr/bin/rsync        –a        –e              ssh          root
@192.168.0.6:/var/log/squid/access.log    /logf/cafe_log/
[ $?    -gt    0]    $$ exit 0
[-S    /logf/cafe_log/access.log ] | | exit 0
/usr/bin/webalizer      -C      /etc/cafe_webalizer.conf      >
/dev/null    2    > &1
```

5. Example Application of System

5.1. External Link Monitoring

Below are some output generated by the scalable network traffic monitoring and analysis system. Two types of outputs were generated, a graphical overview and a detailed statistics. MRTG generates the graphical overview while Webalizer produced the detailed statistics and proxy log analysis.

5.1.1. Sample MRTG Graphs

The MRTG graph is influenced by the socio-economic activities on the university campus. Academic activities start at 6.00am with some lectures and probably students trying to finish their assignments via online research. There is a pronounced increase at 8.00am when most lecturers arrive in the offices along with non academic staff; by 10am the campus is fully active and usage remains constant until 12.00noon, with a dip, when people leave their offices on lunch break or to pickup kids from school. A steady rise in traffic by 2.00pm, when the break is over and users are back in their offices and then a steady drop when users finally leave at about 6.00pm. Some academic staff and students typically return 8pm and work till midnight, while some students continue to work until 2.00am. Network usage is at its lowest at 4.00am and starts its cycle from 6.00am once again.

Note: the pattern on Wednesday 28 April 2004 indicates it is a holiday "workers remembrance and commemoration day" since activity started at 10.00am and continues until 4.00pm when users leave offices for social activities. The weekly graph correlates strongly the socio-economic influence as discussed above except for Saturday and Sunday, where there is a more pronounced late night activity on Saturday and a complete lack of afternoon and night time activity on Sundays.

Figure 8 shows a sample global MRTG graph. The average daily bandwidth usage was 60.3kBps downlink green line and 28.8kBps uplink blue line.

The graph above shows Internet traffic for an academic institution. Notice the relatively constant demand for network resources over time, as indicated by the horizontal pattern of the green, average-rate line. The green line also indicates that average utilization is about 60 KBps. This could be either a single user listening to a 60 KBps stream, or multiple users listening to slower streams that total 60 KBps.

The graph's peaks indicate some burstiness in the traffic that could periodically impact other applications' performance. However, given that this is a one week graph, bursting even once in an hour could cause the peak rate graph to go up. These peaks may just reflect heavy downloads or streaming by some users on the network.

5.1 2. Sample Webalizer Graphs

The Webalizer graph shows the hourly average for the entire month of May 2006. The graph shows a correlation with the discussion from the MRTG graph with respect to the socio-economic activities, that is, a gradual increase in traffic starting at 8.00am with a dip at 12.00pm and a dip at 4.00pm and a gradual decline until 8.00pm when there is a new rise, with a slight decline after midnight.

Unfortunately since this is an average over a month the peaks and troughs are less pronounced. Generally this graph is consistent with the expected usage pattern and indicates a constant usage on the average over a 20 hour period. Figure 9 shows a section of the generated output for hourly usage for May 2006.

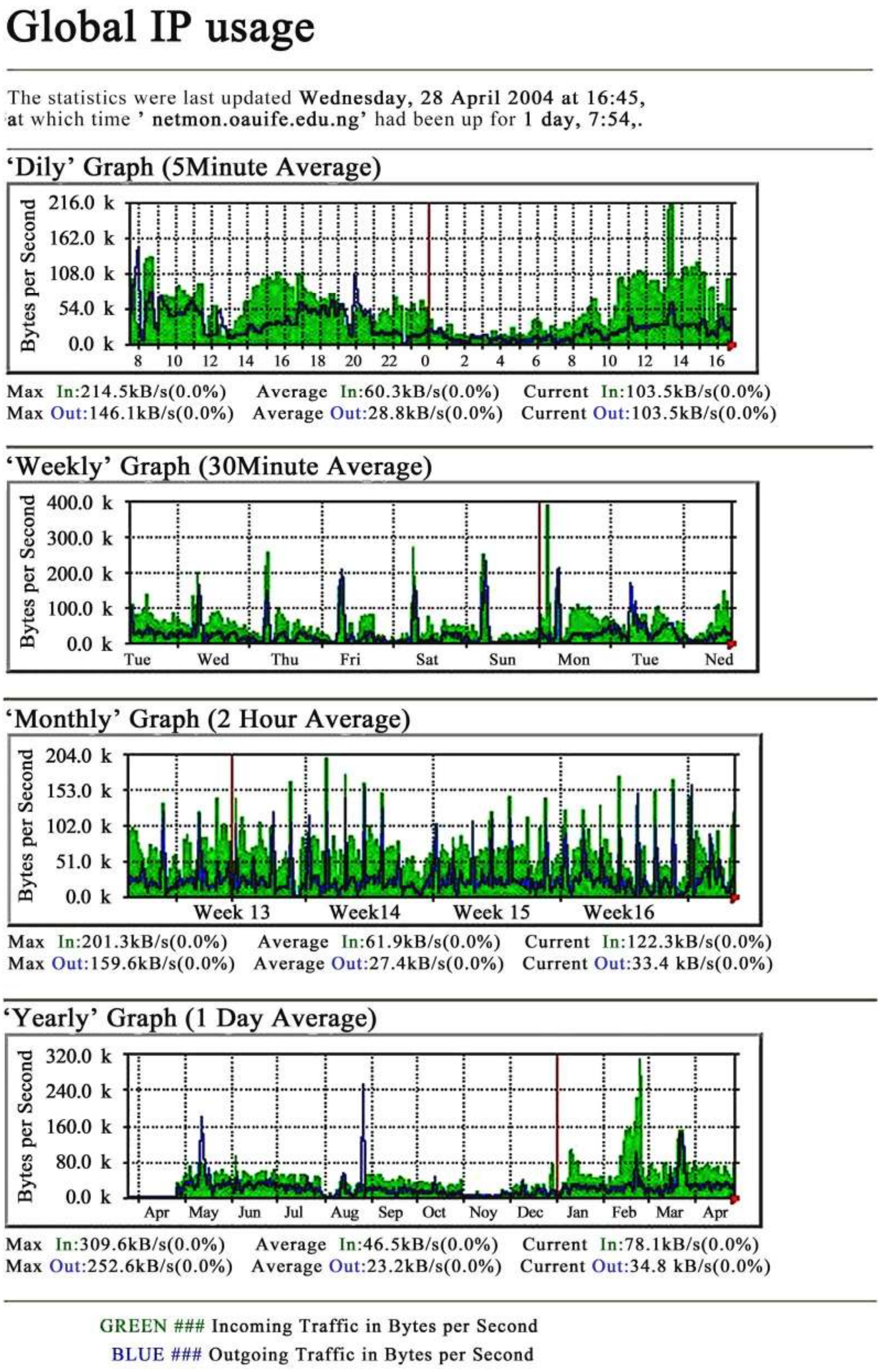
Other techniques including the hardware based network monitoring station (NMS) by CISCO for the monitoring and analysis of the INTERNET traffic exist; however this technique offers two interesting advantages:

Dual independent system (MRTG and Webalizer) makes correlation and deviation easier to detect and the short comings of either methods are overcome by the combination.

Ability to focus or ignore noise traffic i.e. HTTP traffic could be monitored alone or and as a relationship to the global traffic, with/without HTTPS traffic.

5.1.3. Proxy Log Analysis

The scalable network traffic monitoring and analysis system is also capable of proxy log analysis. This was


Global IP usage

The statistics were last updated **Wednesday, 28 April 2004 at 16:45,**
at which time **' netmon.oauife.edu.ng'** had been up for **1 day, 7:54,.**

'Dily' Graph (5Minute Average)

Max In:214.5kB/s(0.0%) Average In:60.3kB/s(0.0%) Current In:103.5kB/s(0.0%)
Max Out:146.1kB/s(0.0%) Average Out:28.8kB/s(0.0%) Current Out:103.5kB/s(0.0%)

'Weekly' Graph (30Minute Average)

'Monthly' Graph (2 Hour Average)

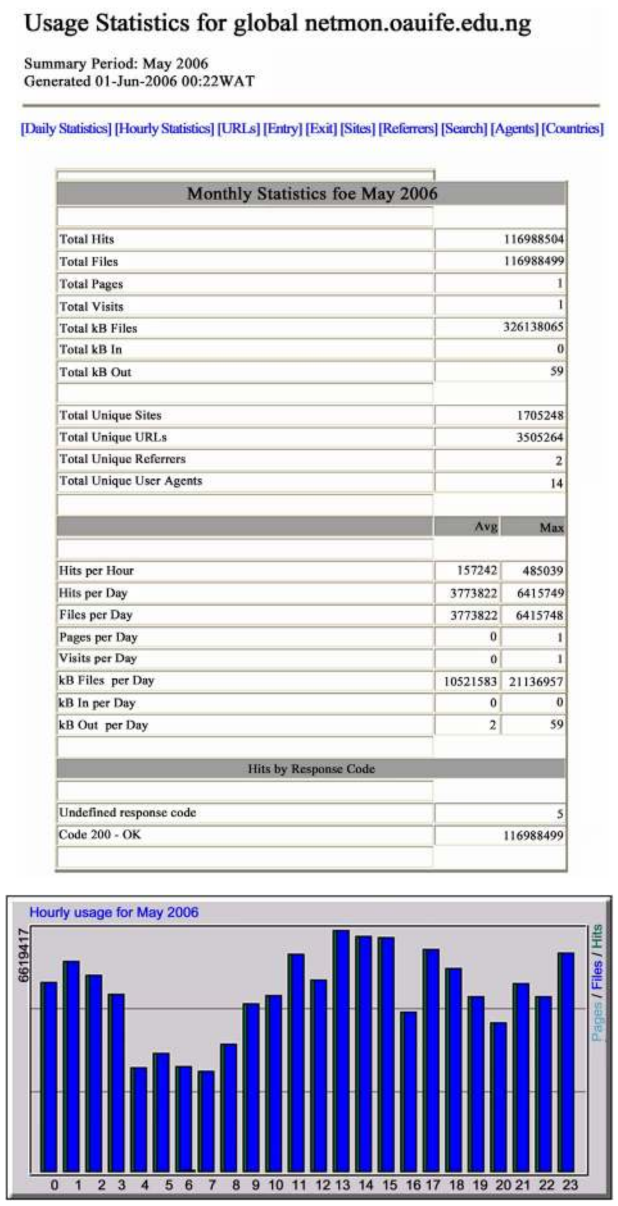
Max In:201.3kB/s(0.0%) Average In:61.9kB/s(0.0%) Current In:122.3kB/s(0.0%)
Max Out:159.6kB/s(0.0%) Average Out:27.4kB/s(0.0%) Current Out:33.4 kB/s(0.0%)

'Yearly' Graph (1 Day Average)

Max In:309.6kB/s(0.0%) Average In:46.5kB/s(0.0%) Current In:78.1kB/s(0.0%)
Max Out:252.6kB/s(0.0%) Average Out:23.2kB/s(0.0%) Current Out:34.8 kB/s(0.0%)

GREEN ### Incoming Traffic in Bytes per Second
BLUE ### Outgoing Traffic in Bytes per Second

Figure 8. Global usage statistics.



Usage Statistics for global netmon.oauife.edu.ng

Summary Period: May 2006
Generated 01-Jun-2006 00:22WAT

[Daily Statistics] [Hourly Statistics] [URLs] [Entry] [Exit] [Sites] [Referrers] [Search] [Agents] [Countries]

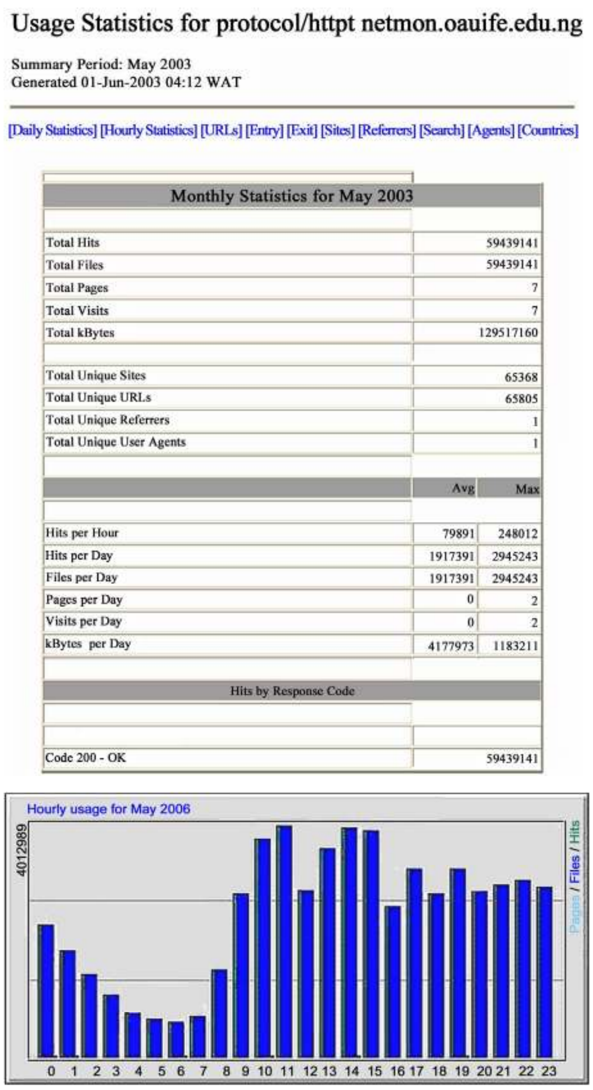
Monthly Statistics foe May 2006		
Total Hits		116988504
Total Files		116988499
Total Pages		1
Total Visits		1
Total kB Files		326138065
Total kB In		0
Total kB Out		59
Total Unique Sites		1705248
Total Unique URLs		3505264
Total Unique Referrers		2
Total Unique User Agents		14
	Avg	Max
Hits per Hour	157242	485039
Hits per Day	3773822	6415749
Files per Day	3773822	6415748
Pages per Day	0	1
Visits per Day	0	1
kB Files per Day	10521583	21136957
kB In per Day	0	0
kB Out per Day	2	59
Hits by Response Code		
Undefined response code		5
Code 200 - OK		116988499

Figure 9. Global usage statistics.

achieved by copying log files from the proxy servers at regular time interval and then running a Webalizer configuration file to analyze the proxy logs collected. The log files were collected every four hours from the proxy server and the analysis carried out. When a new analysis is completed, the graph is only updated, this reduces the amount of storage required to keep historical data. Figure 10 shows the protocol usage statistics.

Suppose a client using a proxy makes requests r_1, r_2...r_n to pages, if a page has F objects out of which C can be obtained from the cache and W from the origin server. Total request R will be:

$$R = \sum_{i=1}^{n} r_i$$



Usage Statistics for protocol/httpt netmon.oauife.edu.ng

Summary Period: May 2003
Generated 01-Jun-2003 04:12 WAT

[Daily Statistics] [Hourly Statistics] [URLs] [Entry] [Exit] [Sites] [Referrers] [Search] [Agents] [Countries]

Monthly Statistics for May 2003		
Total Hits		59439141
Total Files		59439141
Total Pages		7
Total Visits		7
Total kBytes		129517160
Total Unique Sites		65368
Total Unique URLs		65805
Total Unique Referrers		1
Total Unique User Agents		1
	Avg	Max
Hits per Hour	79891	248012
Hits per Day	1917391	2945243
Files per Day	1917391	2945243
Pages per Day	0	2
Visits per Day	0	2
kBytes per Day	4177973	1183211
Hits by Response Code		
Code 200 - OK		59439141

Figure 10. Protocol usage statistics.

But not all requests will bring back data. Hence, all requests that will result in data transfer will be:

$$F = \sum_{i=1}^{n} W_i + \sum_{i=1}^{n} C_i$$

So we can compute the Document hit ratio (DHR) and Byte Hit Ratio (BHR) as;

$$DHR = \frac{\sum_{i=1}^{n} C_i}{\sum_{i=1}^{n} W_i + \sum_{i=1}^{n} C_i} \quad , \quad BHR = \frac{\text{Cache byte}}{\text{Total byte}}$$

Cachebyte = the no of bytes transferred from the cache.
Totalbyte = the total no of bytes transferred.

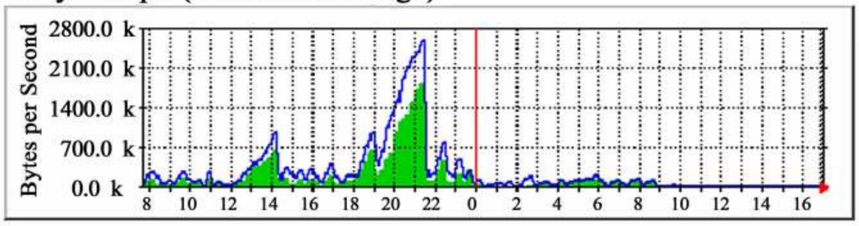
From the Squid Proxy Usage Statistics,

Total hits	= 59,035,756 (total requests)
Total Files	= 15,483,674 (internet content)
Difference	= 43,552,082 (cached content)

10.105.40.253 IP usage

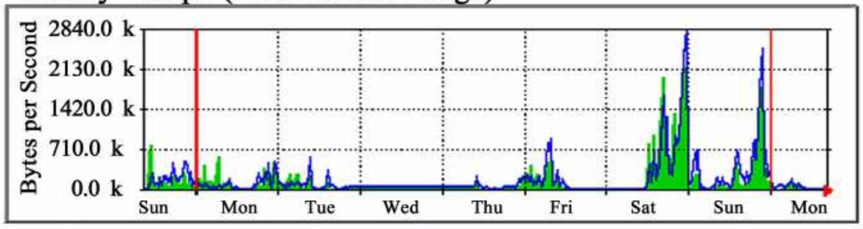
The statistics were last updated **Monday, 30 April 2007 at 17:02,**
at which time **' netmon.oauife.edu.ng'** had been up for **1:12, 2 use.**

'Dily' Graph (5 Minute Average)



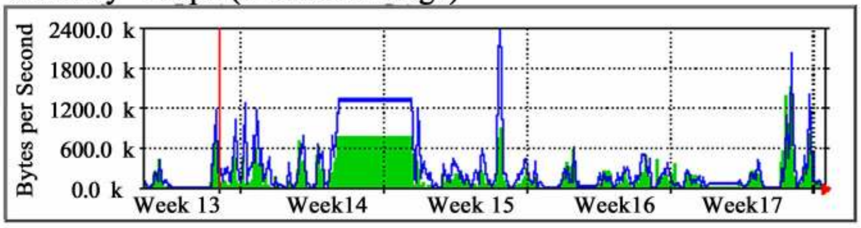
Max In:1847.8kB/s(0.0%) Average In:198.9B/s(0.0%) Current In:4684.0kB/s(0.0%)
Max Out:2587.0kB/s(0.0%) Average Out:309.2kB/s(0.0%) Current Out:11.8kB/s(0.0%)

'Weekly' Graph (30 Minute Average)



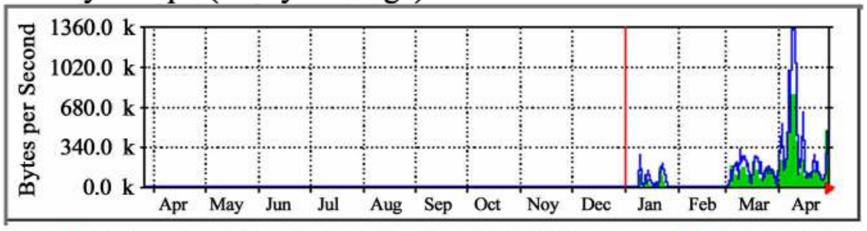
Max In:2184.0kB/s(0.0%) Average In:182.1kB/s(0.0%) Current In:7016.0 B/s(0.0%)
Max Out:2808.8kB/s(0.0%) Average Out:213.2kB/s(0.0%) Current Out:4243.0 B/s(0.0%)

'Monthly' Graph (2 Hour Average)



Max In:1519.6kB/s(0.0%) Average In:224.8kB/s(0.0%) Current In:0.0B/s(0.0%)
Max Out:2374.8kB/s(0.0%) Average Out:369.3kB/s(0.0%) Current Out:0.0 B/s(0.0%)

'Yearly' Graph (1 Day Average)



Max In:791.7kB/s(0.0%) Average In:126.7kB/s(0.0%) Current In:229.1kB/s(0.0%)
Max Out:1333.5kB/s(0.0%) Average Out:199.8kB/s(0.0%) Current Out:376.6 kB/s(0.0%)

GREEN ### Incoming Traffic in Bytes per Second
BLUE ### Outgoing Traffic in Bytes per Second

Figure 11. Hosts usage statistics.

The difference is a measure of cached content. Hence the cache hit rate is about 73%. Other useful information about the network such as no of users, static content, internal server error, total kilobytes transferred, numbers of web servers, top host generating traffic, top destination accessed, top users generating traffic etc can be obtained from the proxy log usage statistics. The cache hit rate calculated is an estimate; a more accurate value can be obtained by combining the data from the hit by response code and the monthly statistics. All the information provided by the monitoring system could be used by the system administrator for shaping the network traffic, improving network performance and predict future traffic trends.

5.2. Internal Network Monitoring

5.2.1. Sample MRTG Graphs

We have described how the scalable network traffic monitoring and analysis system can be used to monitor the external link; it can also be used to monitor the internal network. The type of output or environment in which the system is deployed will determine the setup and configuration used by the system. The system is flexible and extensible and can adapt to varying traffic load conditions. E.g. For a host on the internal network we are interested in traffic patterns, hosts and protocol related activity, but we are not interested in proxy log analysis, this implies that different configurations are used for different setup of the system. Figure 11 shows the MRTG host graph for the internal network. The flat line indicates power outages and periods the system was down. The graph shows high network traffic between 6.00pm and 9.00pm and low network traffic between 8.00am and 5.00pm. This is the pattern of traffic observed on Sundays.

5.2.2. Sample Webalizer Graphs

The Webalizer gives a detailed traffic statistics of the network; from the Webalizer graph we can see the global usage statistics, hourly usage statistics, hosts usage statistics and protocol usage statistics. Figure 12 shows the various historic Webalizer usage statistics.

The Protocol Usage Reports section includes reports that show bandwidth usage based on all the protocol groups generating traffic through the device. The Protocol Trend Reports section includes reports that show trends in the amount of traffic generated using different protocol groups. Protocol trends help in identifying peak usage times for each protocol group, understanding user trends, and enforcing better policies to allow traffic from each protocol group. The system is capable of keeping historic data for several months; this makes it suitable for generating trend reports.

5.3. Replicating the System in Other Environment

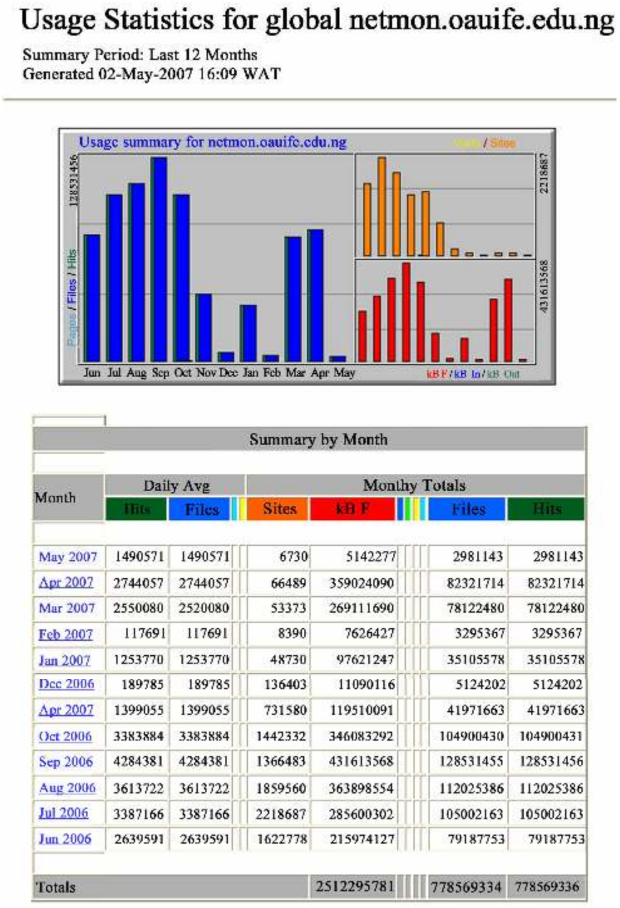
The system can be setup in two simple steps: Install any distribution of the Linux Operating system, this is freely available on the INTERNET and then follow Steps 1 and 2 to setup the hardware and software.

1) Hardware setups as in Subsection 3.2.1.
2) Software setups as in Subsection 3.2.2.

The software used to setup the scalable network traffic monitoring and analysis system and all configuration files are available for download on the author's website, the free open source programs are available at the developer's website, while all the Perl scripts developed are available for download at http://users.ictp.it/~abionao/. We welcome Feedback and contributions from the users of the software.

6. Conclusions

In this paper we have presented the development of a scalable network traffic monitoring and analysis system. The system is capable of monitoring and analyzing network


Usage Statistics for global netmon.oauife.edu.ng

Summary Period: Last 12 Months
Generated 02-May-2007 16:09 WAT

Usage summary for netmon.oauife.edu.ng
/ Sites
128531456
Pages / Files / Hits
2218687
431613568
Jun Jul Aug Sep Oct Nov Dec Jan Feb Mar Apr May
kB F / kB In / kB Out

Summary by Month

Month	Daily Avg				Monthy Totals							
	Hits	Files			Sites	kB F					Files	Hits
May 2007	1490571	1490571			6730	5142277					2981143	2981143
Apr 2007	2744057	2744057			66489	359024090					82321714	82321714
Mar 2007	2550080	2520080			53373	269111690					78122480	78122480
Feb 2007	117691	117691			8390	7626427					3295367	3295367
Jan 2007	1253770	1253770			48730	97621247					35105578	35105578
Dec 2006	189785	189785			136403	11090116					5124202	5124202
Apr 2007	1399055	1399055			731580	119510091					41971663	41971663
Oct 2006	3383884	3383884			1442332	346083292					104900430	104900431
Sep 2006	4284381	4284381			1366483	431613568					128531455	128531456
Aug 2006	3613722	3613722			1859560	363898554					112025386	112025386
Jul 2006	3387166	3387166			2218687	285600302					105002163	105002163
Jun 2006	2639591	2639591			1622778	215974127					79187753	79187753
Totals						2512295781					778569334	778569336

Figuer 12. Global usage statistics.

traffic for both the Intranet and the Internet traffic, the design employ free open source programs such as IPTraf, MRTG and Webalizer, and scripts written in Perl language. The system monitors network traffic passively, this implies that it is non intrusive and monitors network traffic continuously, using the lossy data storage technique to store the data as web pages. Graphical overviews of the traffic monitored were presented using MRTG, while detailed analyses of the traffic and proxy log analysis were presented using Webalizer. A user can view the results of the monitoring system using a web browser. We have described the hardware and software required to setup the system and how it can be replicated on any network. Using the system we were able to monitor the internal and external network, the scalability feature of the system, makes it attractive to both researchers and network managers. The output graph generated by the system provides details of the network dynamics and insight into problems that could lead to congestion and poor network performance. Future work will focus on intrusion detection monitoring module and preemptive intrusion control.

7. Acknowledgements

The authors will like to express our gratitude to the Office of External Activities (OEA) of the Abdus Salam International Centre for Theoretical Physics, Trieste, Italy, for funding the research through the donation of all the equipment used to conduct the study.

8. References

[1] V. Jacobson, C. Leres, and S. McCanne, "Libpcap," 1994, http://www-nrg.ee.lbl.gov/

[2] D. Paraas, "On the criteria to be used in decomposing systems modules," Communications of the ACM, Vol. 14, No. 1, pp. 221–227, 1972.

[3] A. Reid, M. Flatt, L. Stroller, J. Lepreau, and E. Eide, "Knit: Component composition for systems software," in Proceedings of the 4th Symposium on Operating Systems Design and Implementation, pp. 347–360, October 2000.

[4] N.C. Hutchinson and L. L. Peterson, "The X-Kernel: An architecture for implementing network protocols," IEEE Transactions on Software Engineering, Vol. 17, No. 1, pp. 64–76, 1991.

[5] E. Kohler, R. Morris, B. Chert, J. Jannotti, and M. Frans Kaashoek, "The click modular router," ACM Transactions on Computer Systems, Vol. 18, No. 3, pp. 263–197, August 2000.

[6] J. Allen, Cricket homepage, 2000, http://cricket.sourceforge.net.

[7] J. D. Case, M. Fedor, M. L. Schoffstall, and C. Davin, Simple Network Management Protocol (SNMP), May 1990, http://www.faqs.org/rfcs/rfc1157.html.

[8] V. Jacobson, C. Leres, and S. McCanne, "Tcpdump-the protocol packet capture and dumper program," http:// www. tcp dmp.org.

[9] T. Oetiker, "Monitoring your IT gear: The MRTG story," IEEE IT Profesionals, Vol. 3, No. 6, pp. 44–48, December 2001.

[10] G. Robert Malan and Farnam Jahanian, "An extensible probe for network protocol performance measurement," in Proceedings SIGCOMM'98, pp. 215–227, September 1998.

[11] J. Hong, S. Kwon, and J. Kim, "WebTrafMon: Web-based internetintranet network traffic monitoring and analysis system," Elsevier Computer Communications, Vol. 22, No. 14, pp. 1333–1342, September 1999.

[12] J. J. Chen, D. J. DeWitt, F. Tian, and Y. Wang, "A scalable continuous query system for internet databases," Proceedings of ACM SIGMOD'00, pp. 379–390, May 2000.

[13] S. Madden, M. Shah, J. M. HeUerstein, and V. Raman, "Continuously adaptive continuous queries over streams," Proceedings of ACM SIGMOD 2002, pp. 49–60, June 2002.

[14] M. Fisk and G. Varghese, "Agile and scalable analysis of network events," in Proceedings of 2nd ACM SIGCOMM Workshop on Internet Measurement IMW'02, pp. 285–290, November 2002.

[15] L. Deri and S. Suin, "Effective traffic measurement using ntop," IEEE Communication Magazine, Vol. 38, No. 5, pp. 138–143, May 2000.

[16] L. Deri, R. Carbone and S. Suin, "Monitoring networks using ntop," Proceedings of IEEE/IFIP International Sympo-sium on Integrated Network Management, pp. 199–212, May 2001.

[17] L. Deri and S. Suin, "Practical network security experiences with ntop," Computer Networks, Vol. 34, pp. 873–880, 2000.

[18] A. Hussain, G. Bartlett, Y. Pryadkin, J. Heidemann, C. Papadopoulos and J. Bannister, "Experiences with a continous network tracing infrastructure," in Proceedings of ACMSIGCOMM Workshop on Mining Network Data, pp. 185–190, August 2005.

[19] O. O. Abiona, C. E. Onime, A. I. Oluwaranti, E. R. Adagunodo, L. O. Kehinde, and S. M. Radicella, "Development of a non intrusive network traffic monitoring and analysis system," African Journal of Science and Technology (AJST) Science and Engineering series, Vol. 7, No. 2, pp. 54–69, December 2006.

[20] G. R. Wright and W. R. Stevens, "TCP/IP illustrated," 2 Addison-Wesley, Reading, M. A., 1994.

[21] G. P. Java, IPTraf : http://iptraf.seul.org/ 2001.

[22] T. Oetiker and D. Rand, "MRTG: Multi router traffic grapher," http://tobi.oetiker.ch/ 2008.

[23] B. L. Barrett, Webalizer home page, http://www.mrunix-. net /webalizer/ 2008.

[24] J. Vass, J. Harwell, H. Bharadvaj, and A. Joshi, "The world wide web: Everything you (n)ever wanted to know about its servers," IEEE Potentials, pp. 33–34, October/November 1998.