17/05/07

Deliverable DJ2.2.2,2: User and Test Report on NetFlow Probe

Deliverable DJ2.2.2,2

Contractual Date:	31/12/06
Actual Date:	17/05/07
Contract Number:	511082
Instrument type:	Integrated Infrastructure Initiative (I3)
Activity:	JRA2
Work Item:	2
Nature of Deliverable:	R (Report)
Dissemination Level	PU (Public)
Lead Partner	SURFnet
Document Code	GN2-07-073v5

Authors: Ladislav Lhotka (CESNET, Editor), Wim Biemolt (SURFnet), Peter Haag (SWITCH), Luchesar Iliev (ISTF), Petros Politopoulos (GRNET)

Abstract

Flow-based monitoring using Cisco NetFlow has been identified by GÉANT2 Joint Research Activity 2 (JRA2) as the principal source of data for security analysis and intrusion detection. In order to make NetFlow data available in all locations of both the trans-European and NREN backbone networks, JRA2 started the development of an autonomous NetFlow probe named FlowMon. This second version of the deliverable summarises test results and other experiences from the five GÉANT2 partners who have been testing both the prototype and the final version of the FlowMon probe with Gigabit Ethernet interfaces.

Table of Contents

0 Executive Summary

This deliverable describes the results of user testing on the NetFlow Probe, developed as part of Joint Research Activity 2 (JRA2, Security).

Because of the importance of backbone networks to both the GÉANT2 project and the National Research and Education Networks (NRENs), JRA2 identified flow-based monitoring as the principal source of data for security analysis and intrusion detection.

Cisco NetFlow technology is the universally accepted industry standard in flow-based monitoring. Because of this a hardware-accelerated IP flow probe (FlowMon) that utilises NetFlow is currently being developed within JRA2 Work Item 2 as a part of the Security Toolset. FlowMon is able to monitor bi-directional Gigabit Ethernet links at line rate without sampling, and export flow records in the NetFlow v5 and v9 formats. Support for 10GE interfaces is under development, with tests scheduled for April 2007.

Since the initial publication of this deliverable, four additional project partners have been involved in testing the prototype version of the probe in their networks. CESNET lent them the prototype cards in order to facilitate thorough testing. The partners' bug reports, feature requests and other feedback helped to improve the firmware and software of the probe considerably.

This deliverable describes:

- The general experience of using a stand-alone probe over the standard method of NetFlow data acquisition from IP routers.
- The steps for connecting the probe to the network, initialising the firmware and running the flow exporter program(s).
- A brief comparison of the prototype with the production version.
- The results of laboratory and field tests performed by five JRA2 partners.

Apart from several minor bugs, the overall experience (in terms of performance, stability and features) is very positive. The probe is recommended, together with the Nfdump/NfSen collector and visualisation suite, for inclusion in the JRA2 Security Toolset.

1 Introduction

Current methods for network security analysis and intrusion detection often rely on combining and correlating data from various sources, for example utilisation, statistics of IP flows, and detailed packet payload inspection. This integrated approach was also considered in the early phases of JRA2 activity. However, the focus of the GÉANT2 project and most of its partners is on backbone network operation, and an analysis showed that payload inspection on backbone links is beyond the reach of existing technology:

- Software tools (for example *Snort*[1]) can handle traffic rates only in the range of few hundred megabits per second (depending upon processor speed). This is obviously insufficient for network links with capacities of 1 Gb/s and higher.
- Methods for hardware-accelerated payload inspection and pattern matching on high-speed network links utilising Field-Programmable Gate Arrays (FPGA) are the subject of intensive research. However, neither commercial products nor functional prototypes are currently available.

Because of these factors, JRA2 decided to abandon the field of payload inspection and concentrate instead on the collection and analysis of data about IP flows using Cisco Netflow, the recognised industry standard. Unlike payload inspection, flow-based monitoring is best deployed in backbone networks, and methods for collecting and processing flow data from multi-gigabit links are relatively well understood.

Typically, Netflow data is acquired and exported by IP routers. However, this has several drawbacks, especially with respect to security related monitoring:

1. Routers are, by definition, visible Layer 3 devices that are easily discovered by simple tools such as *traceroute*. Consequently, they can become targets for all kinds of attacks and intrusions.

2. The main task of routers is to forward datagrams and exchange routing information with their neighbours. The available processing power is therefore rather limited, and so operations on Netflow data do not usually go much beyond simple export of raw flow records.

3. As a special case of the previous item, some routers impose sampling on the input traffic. Even if sampling is not mandatory, in some cases it is the only way of keeping the router operational, and/or NetFlow data reliable, especially when monitoring high speed interfaces.

[1] http://www.snort.org

This deliverable describes the hardware-accelerated FlowMon probe that is being developed within JRA2 as a part of the Work Item 2 toolset for security monitoring.

The probe was designed to reuse programmable hardware cards of the COMBO family, which were built for earlier FP5 IST projects 6NET[1] and SCAMPI[2]. The FlowMon firmware and software is being developed under free software licences, as a result of the GN2 project. A BSD-like licence is used for firmware and certain parts of low-level software, while GPL version 2 is applied to Linux kernel modules and some applications (such as flow exporters and control front-ends).

At present, the probe is able to monitor both directions of a Gigabit Ethernet link at line rate without sampling. Several sampling methods are provided but they are strictly optional and should generally be avoided for security-related applications. This GE version of the FlowMon probe is aimed primarily at NREN and university backbones. Support for 10GE interfaces is under development, expected to be finished in summer 2007. This 10GE card will be also suitable for deployment in the GN2 backbone,although the throughput of the current hardware is limited to about 3 million packets per second. For higher data rates, sampling will have to be turned on.

The prototype version of the probe was finished in September 2005. In early 2006, CESNET lent four cards to JRA2 partners who expressed interest in testing the prototype. Consequently, the FlowMon prototype was deployed in five NREN backbone networks (CESNET, GRNET, ISTF, SURFnet and SWITCH). The development of the probe continued in parallel to these field tests. This has resulted in the final version of the FlowMon probe with GE interfaces. It is planned that ten pieces of this probe will be purchased in year 3 of the GN2 project on a standard commercial basis (including support and maintenance) from a start-up company that CESNET helped to establish. Eight of the new cards will then be distributed to GN2 partners as a part of the Security Toolset and two kept for testing purposes.

This deliverable contains:

- An overview of the probe from the user perspective (without going into excessive technical detail). The hardware and firmware design of the probe is described in [ZPK05] (prototype version) and [Cel06] (final version).
- Step-by-step instructions for connecting the probe to the network and getting it up and running.
- Laboratory tests with the final version that were carried out as a part of the probe development.
- The results of field tests performed by five project partners in their production networks.

[1] FP5-IST-2001-32602, http://www.6net.org
[2] FP5-IST-2001-32404, http://www.ist-scampi.org

2 FlowMon Probe

The FlowMon probe consists of a PC (running Linux) that is equipped with an add-on hardware card for accelerated flow processing.

The probe is currently able to monitor Gigabit Ethernet links. Support for 10GE is under development.

In a typical case, the host PC also has a separate standard network interface card used for sending Netflow records to a collector over the network, and also for remote configuration and management. However, it is also possible to run the collector and presentation back-end directly on the PC hosting the probe.

Tests started in 2005 with a prototype version of the probe. In early 2006, CESNET lent these prototypes to four JRA2 partners (GRNET, ISTF, SURFNet, and SWITCH) who performed further tests in their NRENs. In the meantime, CESNET continued the development that eventually led to the final version of the probe, which is now fully operational and has been extensively tested by CESNET. JRA2 plans to distribute eight probes to GN2 partners as a part of the Security Toolset in 2007.

The prototype and final version of the probe use different hardware. The firmware design of the final version has also been considerably improved. However, most of the system software is common for both versions.

Both versions of the probe and their major differences are described below.

2.1 Prototype version

2.1.1 Hardware

The FlowMon prototype was based on the COMBO family of programmable hardware cards. It is a combination of two interconnected cards that fit into a single 32-bit/33 MHz PCI slot (see Figure 1). This combination consists of:

- A *COMBO6 motherboard* that houses most of the processing logic and connects the accelerator to the PCI bus.

- The accelerator is connected to the network through the ports on an *interface card*. A choice of two Gigabit Ethernet interface cards are currently available: (i) COMBO-4MTX with four metallic ports, and (ii) COMBO-4SFP with four cages for standard SFP transceivers.

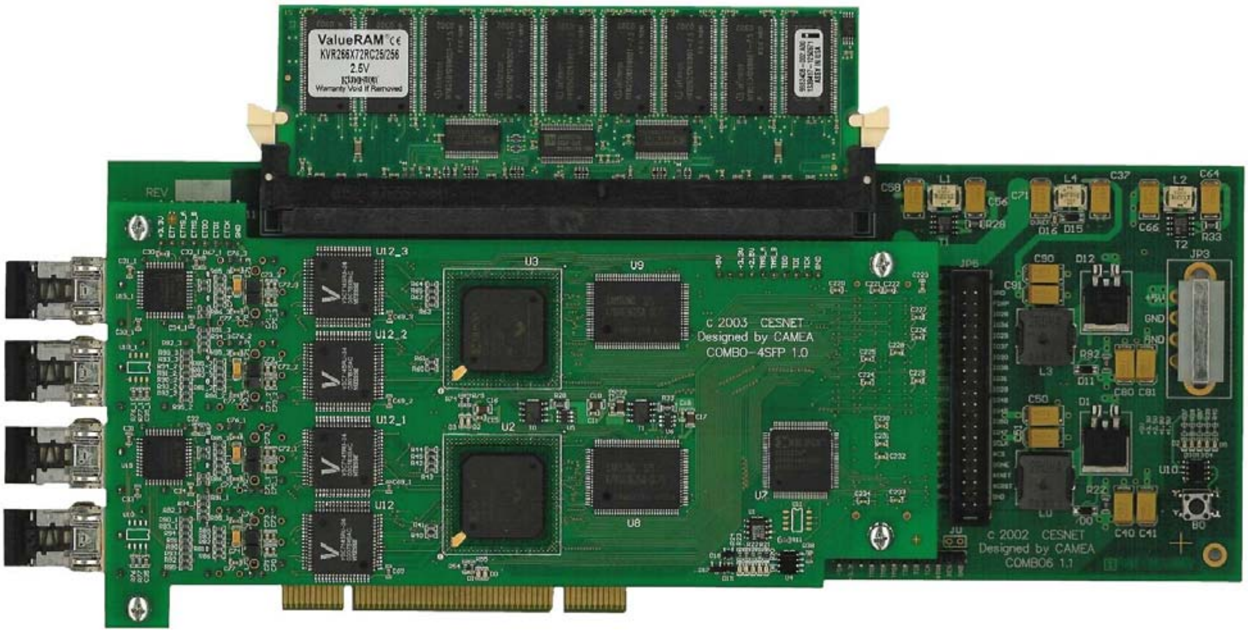


Figure 1: Prototype version of the FlowMon probe: COMBO6 motherboard (bottom) and COMBO-4SFP (top).

2.1.2 Firmware

Firmware for the FlowMon cards is written in VHDL (VHSIC Hardware Description Language, where VHSIC stands for Very-High-Speed Integrated Circuit).

The prototype version of this probe is able to simultaneously process IPv4 and IPv6 traffic. From the viewpoint of the monitored link, the probe acts as a repeater: The ingress traffic received on port 0 of the interface card is immediately transmitted to port 1 while a copy of the traffic is passed to the firmware for flow processing.

The firmware is able to maintain records of up to 2^{16} = 65536 flows (approximately) at the same time.

The processing pipeline is depicted in Figure 2. This works as follows:

Packets received by the Input Buffer (IBUF) are passed to the Header Field Extractor (HFE). This unit parses L2, L3 and L4 headers, extracts all relevant fields and records them in a fixed data structure named *Unified Header* (UH) that is stored in a RAM-based queue (Statistical FIFO).

In parallel, certain *key fields* from the UH are used as input to the CRC-64 hash function implemented by the HASH unit.

From the 64-bit result of the CRC-64 function, 57 bits are used as the unique identifier of the flow. By default, the next five items from the IP and TCP/UDP headers are used as key fields that uniquely identify the flow: source and destination IP addresses, source and destination ports, and L3 protocol. The firmware can be configured to mask an arbitrary selection of bits from these key fields so that only the remaining bits are used for computing the hash function. The flows are more aggregated in this case.

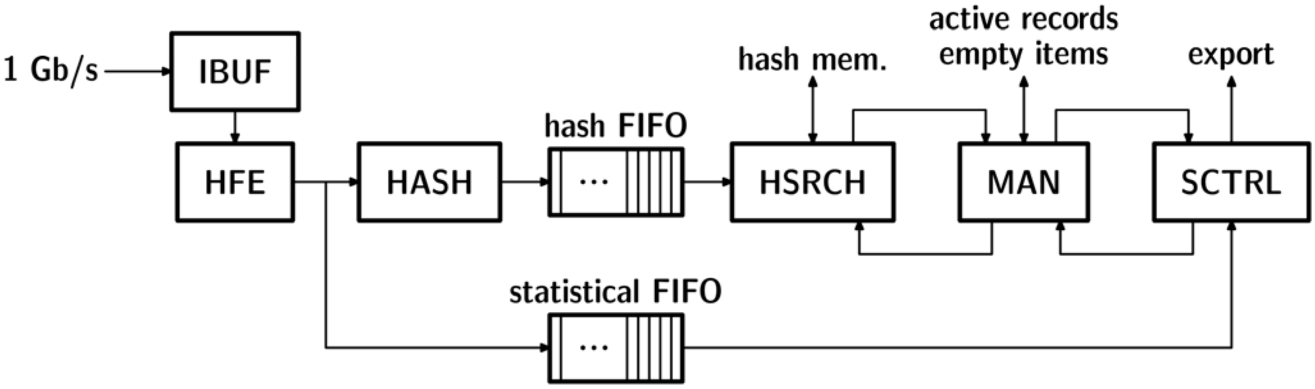


Figure 2: Block diagram of the FlowMon prototype firmware

The use of a 57-bit hash value as the flow identifier means that packets with differing key fields may occasionally map to the same hash value, resulting in two packets being incorrectly classified as belonging to the same flow. However, given the statistically uniform distribution of CRC-64 values, the probability of such an *undetected* collision of flows is $N \cdot 2^{-57,}$ where N is the actual number of flows in the cache. As the maximum number of flows in the cache is $N_{max} = 2^{16}$, the collision probability cannot be higher than $2^{-41} \bullet 4{,}55 \cdot 10^{-13}$. With the maximum throughput of half million packet per second, this means that seven collisions are likely per year on average. While this probability is sufficiently low for most purposes, we have also taken into account the possibility of an attacker generating a hash collision on purpose (for example by injecting carefully forged traffic and then launching hostile traffic that will be classified as the previous forged flow due to the hash collision). This scenario is not extremely difficult to put into practise given that the hash function is known. To prevent this threat, the hash unit is initialised with a random seed so that the values of the hash function are not predictable.

The hash value is stored in the Hash FIFO (First In First Out). On the opposite end of this queue, the Hash Search Unit (HSRCH) takes the hash values one-by-one and searches the Hash Memory in order to find out whether the hash value is already present. If it is already present, the SCTRL unit is instructed to update the statistics of the existing flow. Otherwise, a new entry in the Hash Memory is created.

The Manager unit (MAN) looks after all flows entries in the Hash Memory, and also manages the list of free memory locations. The flow entries are kept in a bidirectional list sorted by the timestamp of the flow entries. This way, inactive flows are easily recognised when their age exceeds a given threshold (inactive timeout).

Finally, the Storage unit (SCTRL) collects the statistics on active flows and exports flow records according to instructions obtained from the MAN and HSRCH units.

The prototype firmware includes two methods of packet sampling:

- *Statistical sampling*, implemented in the IBUF unit.
- *Sample-and-hold* [EV02]. This method is similar to the statistical-sampling, however sampling is avoided for flows that already have entries in the flow cache. This way, it can provide very precise information about large flows. Sample-and-hold is applied at the HSRCH unit.

It is worth pointing out that the probe is able to process data in both directions of a Gigabit Ethernet link at line rate. Sampling is therefore always optional.

Several parameters governing firmware behaviour can be modified dynamically, without disrupting the operation of the probe:

- Active timeout in the range 0-1200 seconds.
- Inactive timeout in the range 0-60 seconds.
- Sampling rate (separately for the statistical sampling and sample-and-hold) in the range 1-65535.

2.2 Final version

Apart from major improvements in performance and capacity, the final version of the prototype provides, several new functions, including address anonymisation, adaptive control of parameters, and traffic mirroring.

2.2.1 Hardware

The final version (see Figure 3) uses newer cards from the COMBO family:

- *COMBO6X motherboard* with support for either 64-bit/66 Mhz PCI or PCI-X.
- *COMBO-4SFPRO interface card* with four cages for SFP transceivers.

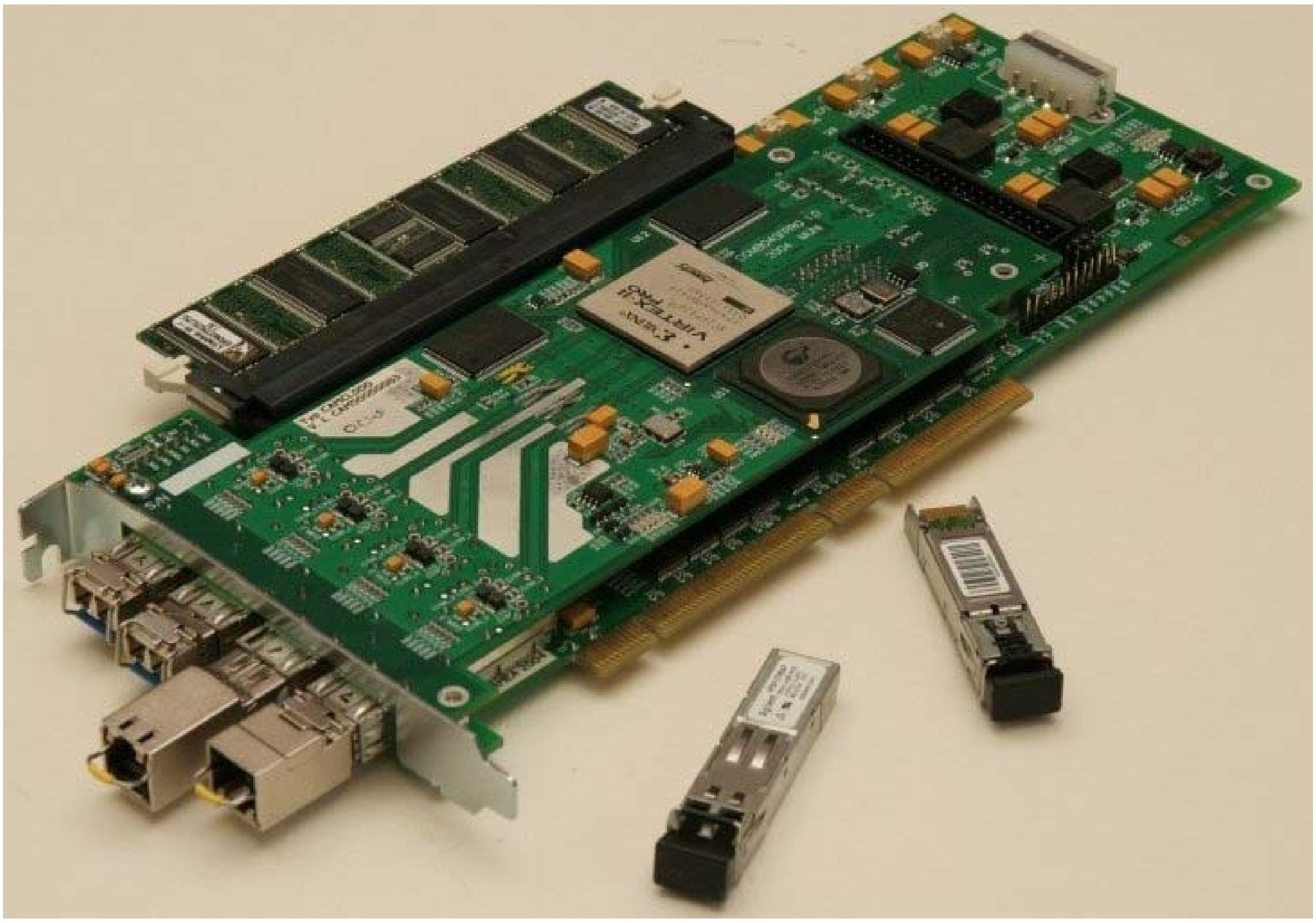


Figure 3: Final version of the FlowMon probe – COMBO6X motherboard (bottom) and COMBO-4SFPRO interface card (top).

Both cards have considerably more resources than their predecessors, which provide new functions, performance improvements and an almost ten-fold increase in flow cache size.

2.2.2 Firmware

The new firmware design is composed of the same building blocks as in Figure 2. However, the input part of the design, which was the bottleneck of the prototype, has been improved so that each monitoring interface uses two processing pipelines, with separate IBUF and HFE blocks, in parallel. Also, the HFE unit has been optimised to increase its throughput.

These improvements mean that the final version is able to simultaneously monitor both directions of GE links at line rate, and that the theoretical throughput limit of this configuration has been raised to 3.2 Gb/s. The bottleneck is now the connector between the two cards.

The final version is able to process MPLS-encapsulated traffic in a simplified way: all MPLS shim headers are removed (that is, the information about MPLS labels is not included in the flow records).

The flow cache was redesigned to use a cascade of various types of memories including dynamic RAM (DRAM). The cache is therefore able to accommodate approximately half a million simultaneous flows, which is more than most high-end routers.

Since the probability of a CRC hash value collision is directly proportional to the number of flows in the cache, increasing the size of the flow cache has the adverse effect of making collisions more likely. To avoid this, the hashing procedure now uses 61 bits of the CRC-64 value (in comparison to the prototype version using only 57). The resulting collision probability is therefore even lower than in the old version.

Another feature added to the new firmware is the hardware anonymisation of IP addresses in packet headers. For this purpose, the CRC-64 hash function is used again.

The new firmware also provides basic means for adaptive control of the two most important parameters: sampling rate and inactive timeout. The administrator can specify up to 16 values for each of these two parameters, and specify ranges of flow cache occupancy where each of the values are to be applied (with hysteresis). This mechanism can be used for avoiding critical situations such as flow cache overflow during a denial-of-service attack.

Finally, since the interface card has four ports and only two of them (0 and 1) are used for flow monitoring, the probe can now also optionally replicate the traffic. In this mode, data arriving on port 0 are sent out not only from port 1 (repeater function), but also from port 2, and similarly for the other pair of interfaces.

2.3 FlowMon Software

The structure of the system software for the FlowMon probe is shown in Figure 4. It is being developed for the Linux operating system. The layer directly above the COMBO hardware is handled by the device drivers that make the hardware accessible to the operating system. Due to the differences in the underlying hardware, there is a special variant for the prototype (Phase 1) and the final version of the probe (Phase 2).

The remaining user-space software is identical for both the prototype and final version. Another layer was inserted between the driver and application programs; the *Libcsflow* library. Its role is to hide low-level system calls and provide a consistent API for the C language. It also aids in debugging the applications by allowing them to run without the COMBO hardware (the data about flows may be served from a disk file).

The exporter programs shown in Figure 4 read the flow records by calling appropriate functions in Libcsflow and sending them to collectors using one of the supported flow export protocols (NetFlow version 5 and 9, IPFIX in the future). The collectors are not part of the FlowMon software, although they can also run on the same computer that hosts the probe.

The three software modules shown in the upper right corner of Figure 4 are currently under development and will provide a remote configuration and management interface. The *Flowmond* daemon is intended as the only entry and exit point for configuration requests and system messages, respectively. At present, the probe is configured through a terminal (SSH) session directly at the machine hosting the probe.

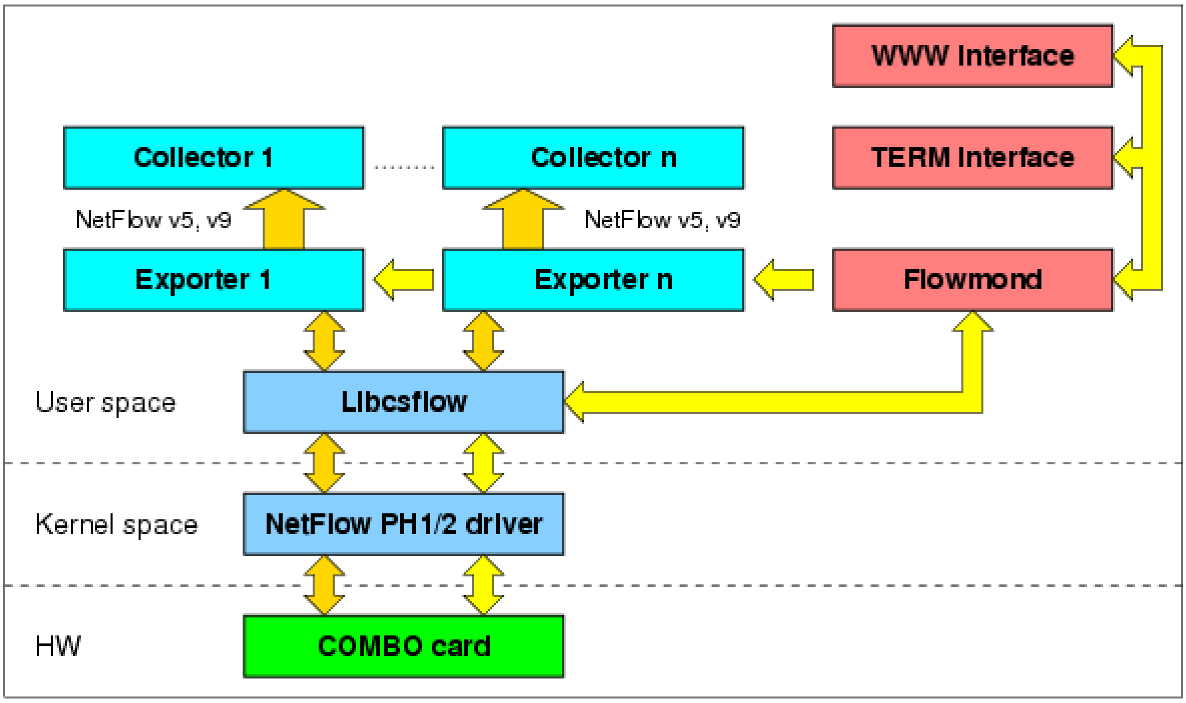


Figure 4: Structure of FlowMon system software

2.3.1 Device driver

The device driver for the FlowMon probe supports recent Linux 2.4 and 2.6 kernels. It allows for concurrent access of multiple exporters or other applications to the flow records. This is accomplished by using a single shared memory block for storing flow records in the circular buffer structure. When the buffer becomes full, the oldest records get overwritten by the new ones.

Each application keeps its own pointer to the buffer and can also lock up to 1024 records in order to prevent them from being overwritten before the application has read them.

For the FlowMon prototype, the size of 16K records was sufficient in most cases. However, the final version (with a larger flow cache) requires a buffer size of at least 65K records.

The tests of the prototype version revealed problems with DMA memory allocation. After several restarts of the driver, the memory became too fragmented and no DMA page was available, meaning that the host computer had to be rebooted. To alleviate this problem, a new kernel module was developed to preserve the allocated DMA pages across restarts of the FlowMon driver.

Support for the *sysfs* file system has been added. It makes use of the *udev*[1] mechanism, which is becoming standard in most Linux distributions, for creating the necessary device files in the `/dev` directory automatically at system startup. In the future, it could also be used for other purposes (such as loading the firmware).

The driver consists of the following Linux kernel modules (some of them are intended only for certain hardware and firmware configurations):

combo6.ko	Core for the COMBO6 card (with PLX PCI bridge), used by the prototype version.
combo6x.ko	Core for the COMBO6X card (with PCI bridge as IP core in the FPGA), used by the final version.
combo6core.ko	Core of the driver.
libermemalloc.ko	Module to keep DMA pages reserved for FlowMon. It is loaded only once, at system boot.
szedata.ko	Straight zero-copy data interface API.
szedatax-c6pcr.ko	Universal SZEDATA driver for COMBO6X with DMA support.
netflow-ph1.ko	Phase 1 FlowMon driver for COMBO6.
netflowx-ph1.ko	Phase 1 FlowMon driver for COMBO6X without DMA support.

2.3.2 Flow Exporter Programs

While it is perfectly possible to process the flow records supplied by the driver directly on the computer hosting the probe, a better scenario is to pack them into a standardised format and send over the network to a remote machine that runs the “collector software”. Since most of the functionality is common to implementations of all export protocols, a generic framework was developed that currently supports two specific instances: NetFlow version 5 and 9.

The framework allows the flow records to be transported in either IPv4 or IPv6 packets. The only L4 protocol that is currently implemented for transport of flow records is UDP. Other options (TCP, SCTP) will be added along with IPFIX support.

[1] See http://www.kernel.org/pub/linux/utils/kernel/hotplug/udev.html

Flow records of NetFlow v5 cover a fixed set of 18 information elements about each flow, including source and destination IP addresses, ports and interfaces, type-of-service, and cumulative TCP flags. Although this approach is rigid and inefficient, NetFlow v5 is still important as it is the only protocol supported by many collectors.

NetFlow v9 [RFC3954] adds considerable flexibility by using templates to describe the layout of exported flow records. The templates are sent in band, and each device exporting NetFlow v9 typically supports several of them. The receiving collector must first read the templates in order to be able to parse the flow data. The **flowmon_nf9** exporter program uses eight NetFlow v9 templates for all combinations of L3 (IPv4, IPv6) and L4 (TCP, UDP, ICMP, OTHER) protocols. Detailed layout of these templates can be found in the Appendix. NetFlow v9 options are not supported yet.

Since version 1 of this deliverable, new modules have been added to the exporter framework:

1. Flow filtering: Flow records sent to a particular collector can be filtered using one or more ranges of IPv4 or IPv6 addresses that can be matched against source, destination or both addresses in the flow. This way, for example, every collector can be offered a specific subset of available data it is entitled to see for policy or privacy reasons.

2. Flow anonymisation: As an alternative to the packet-level anonymisation performed by the FlowMon firmware, an exporter can be instructed to anonymise the IP addresses in the flow records. This module provides either trivial anonymisation of IPv4 addresses by changing the first two octets to 192.168, or a more sophisticated method based on the AES block cipher, which can be applied to both IPv4 and IPv6 addresses, and also to both port numbers.

3 Probe Installation and Setup

The distribution package containing the firmware and software necessary for running the FlowMon probe is available to JRA2 partners on registration under open source licences[1]. Although the software is distributed as source code that must be compiled and installed on the computer hosting the probe, the procedure is quite simple and can be easily carried out even by non-experts. The following subsections describe typical steps that are the necessary minimum for getting the probe up and running. Additional details are provided in the comprehensive README file that is included in the distribution package.

3.1 Host Computer

The probe can be hosted by any standard PC. However, in order to guarantee sufficient heat dissipation, a full-size case with efficient fans is recommended. The combination of the COMBO6(X) card and an appropriate interface card should be plugged into a PCI slot.

In most cases, another network interface card (standard Ethernet) will be needed for sending the export packets to a remote collector, and also to establish a remote connection to the host computer for configuring and monitoring the probe.

3.2 Network Connection

The probe should be inserted as a repeater into the monitored link so that traffic entering port 0 (adjacent to the PCI connector) leaves port 1, and vice-versa. Only Gigabit Ethernet is currently supported. In particular, plain or Fast Ethernet will not work.

The repeater function is robust, and tests show it is safe to use in production links. The probe as a repeater does not depend on any software, and therefore should remain operational even during and after rebooting the host machine, with approximately one second outage. However, as an active device it will not survive a power loss. Therefore, in order to eliminate the risk of the probe causing network problems, use either of the following arrangements:

[1] URL: http://www.liberouter.org/clients/

- A passive optical splitter inserted into the active link and the probe connected to it. This is by far the safest configuration. However this needs two splitters for monitoring the link in both directions.
- Some routers and switches offer the “mirror” or “SPAN” function that copies traffic from a live port to another port where the probe can be connected. However, in many devices this feature is not very reliable and not intended to be used for extended periods of time.

3.3 Software Prerequisites

All software for the probe (driver, control utilities and the flow exporter) is currently supported on Linux only. So far, most tests have been performed using the Debian Linux, but any up-to-date distribution should be suitable.

The software is distributed as a source package and must be compiled on the host computer. To be able to do this, the C development environment must be installed. The GNU C Compiler of major version 2 or 3 is recommended. In addition, *autoconf* and *automake* programs are also required. Most Linux distributions provide these as add-on packages.

3.4 Compiling and Installing the Software

Firmware and software is distributed using the standard Linux and Unix utilities **tar** and **gzip. They are contained within** a file named `flowmon-`*version*`.tgz` (where *version* denotes the version of the package, such as `1.1.0`). This archive can be extracted in any appropriate location by the command:

```
$ tar xzvf flowmon-version.tgz
```

This will create a new subdirectory in the current directory of the format `flowmon-`*version*.

The next three steps can be:

```
$ cd flowmon-version/base
```

```
$ ./pkgtool --build
```

```
# ./pkgtool --install --udev
```

Note: The last command must be executed with superuser privileges.

The above sequence builds the Linux kernel modules and all remaining FlowMon software, and installs it under `/usr/local` (this can be changed by giving the `--prefix=...` option to both `pkgtool` commands). In the last command, the `--udev` option indicates that the host operating system supports sysfs/udev; this is now standard in virtually all major Linux distributions based on kernel 2.6.

Depending on the configuration of the host operating system and the choice of the installation prefix above, some environment variables (`$PATH`) and configuration files (`/etc/ld.so.conf`) may need to be updated. Refer to the README file for details.

3.5 Kernel Modules

Four Linux kernel modules must be loaded before the FlowMon probe can be started:

- For the prototype version: combo6core.ko, combo.ko, szedata.ko and netflow-ph1.ko
- For the final version: combo6core.ko, combo6x.ko, szedata.ko and szedatax-c6pcr.ko

The distribution package contains a script named `flowmonlkm` that can be used to insert or remove the modules dynamically to and from a running kernel. However, in most cases it is preferable to have the modules loaded at system startup. To achieve this on a Debian or Ubuntu system, enter the module names line-by-line to the file `/etc/modules` (other Linux distributions may use different conventions).

Note: Due to the dependencies among the modules, the DMA memory management module `libermemalloc.ko` will also be automatically loaded. This module must not be removed even if the other modules are removed.

With all kernel modules properly loaded, you can check that the cards are detected by running the following command:

```
# csid
```

This results in output as shown below, which is for the final version. The last string identifies the model of the FPGA chip on the COMBO motherboard.

```
combo6x sfpro xc2vp20
```

3.6 Configuration File

The scripts that start the probe set the parameters of FlowMon firmware and software, such as sampling rate, timeouts, export protocol, collector address and others, according to the variables specified in the configuration file `/etc/liberouter/flowmon.conf`. The variable names are mostly self-explanatory and the sample file in the distribution is heavily commented, so it is usually quite easy to modify the file to suit. For further details, see the **flowmon.conf(5)** manual.

3.7 Starting the probe

After installation and configuration, the probe is started by running the `flowmon` script that is included in the distribution package. It is not necessary to run this script with superuser privileges; the only requirement is that the user executing this script (and other commands below) can read and write the device files `/dev/combosix/0` and `/dev/szedata/0`.

The script can be run without any command-line options (all parameters are then read from the `flowmon.conf` file) but each configuration setting can be easily overridden from the command line. For example, the command:

```
$ flowmon -ec collector.geant2.net:63780
```

enables the export of flow records and specifies the host name and port of the collector.

The preferred method for operating the probe in a production environment is through the use of init scripts included in the package. These can be installed and activated in order to start the probe as a part of the start-up sequence of the host operating system.

4 Tests

The addition of four new JRA2 partners (besides CESNET) after the first version of this deliverable, considerably expanded the range of tests. This expansion revealed a number of new issues, partly because of the different environments involved and partly because of the well-known phenomenon of “developer blindness” (developers often proving resistant to certain types of errors).

In the following subsections the tests and their results performed by each of the partners are described. As CESNET is the only partner with access to the final version of the FlowMon probe, its results also include performance tests of the final version.

4.1 CESNET

4.1.1 Throughput

The performance of both the prototype and final version of the FlowMon probe in the full-duplex mode (traffic arriving in both directions simultaneously) was tested. The measured quantity was *throughput*, defined as the highest packet rate captured by the probe without a single packet being lost.

The Spirent AX/4000 traffic analyser was used to generate 250 flows in each direction. The throughput was determined by varying the packet rate of the generated flows and performing a binary search to find the closest approximation of the actual throughput. For both versions of the probe, a series of measurements with the same range of frame sizes was taken. The results are shown in Table 1 and Figure 5.

Frame size [bytes]	Prototype [packets/s]	Final version [packet/s]
66	494 963	2 985 271
128	381 641	1 695 562
256	256 396	905 808
512	154 785	469 930
1024	86 047	239 466
1518	61 192	162 340

Table 1: Throughput of the prototype and final version as a function of frame size

The improved performance of the final version over the prototype is particularly remarkable for small packet sizes. This is due to the redesign and parallelisation of the input part of the processing pipeline. The line for the final version (full circles in Figure 5) now essentially reaches the practical throughput limits of Gigabit Ethernet in full duplex.

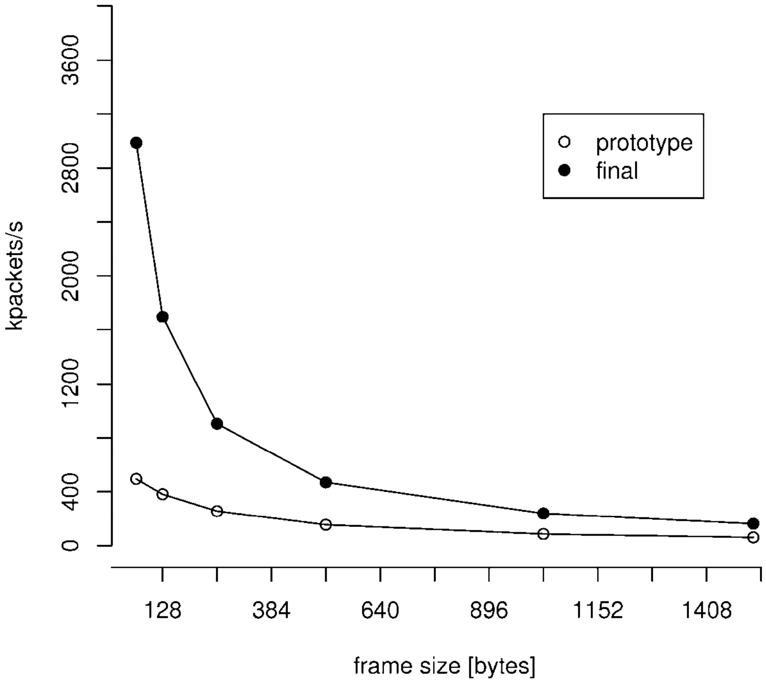


Figure 5: Graph of throughput of the prototype and final version

4.1.2 Comparison with a software probe

The performance of the FlowMon probe was compared to the common software probes, *fprobe* and *nprobe*. Only 1500-byte packets were used for this test, as the performance of the software probes necessarily degrades for smaller packet sizes (or realistic packet mixes) due to the throughput limits of the PC architecture (memory bus and interrupt system). A GE line fully saturated by the Spirent AX/4000 analyser was used. The results are shown in Figure 6, where the blue line represents the traffic rate reported by the final version of FlowMon and the pink line the same parameter obtained from the software probes. fprobe results are shown on the left, and nprobe results on the right.

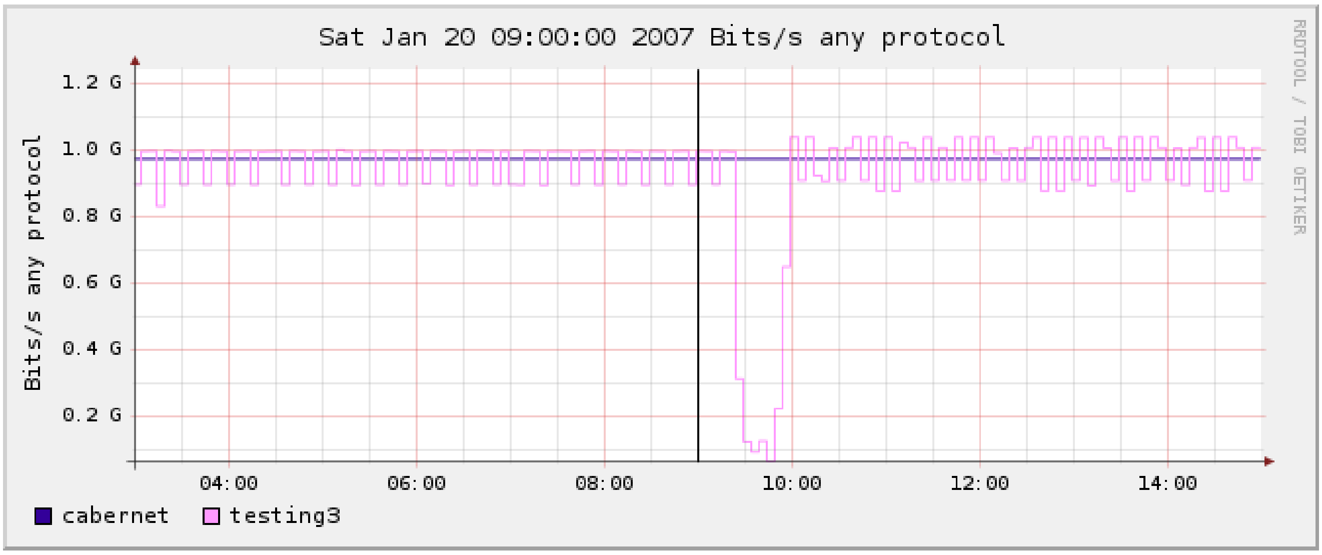


Figure 6: Comparison of FlowMon (blue line) with fprobe (left half) and nprobe (right half)

The results show that the overall performance of the software probes is comparable to FlowMon, although the latter is much more stable.

However, the measurements of both software probes (nprobe in particular) are apparently flawed, as their maxima are slightly higher than the throughput limit of Gigabit Ethernet.

4.1.3 Tests with production traffic

Four FlowMon probes (three prototypes and one final version) were deployed to various locations of the CESNET2 backbone and networks of CESNET customers to monitor:

1. Connection of Akamai servers hosted by CESNET

2. Traffic between CESNET backbone and one of the PoPs (Liberec)

3. Traffic of one of the schools of Masaryk University in Brno

In all cases, the probes were connected to mirror ports on Cisco routers. In case 1 both the prototype and final version of the probe were attached to the same port by using the repeater function, as shown in Figure 7. This way the exported data from both versions could be compared directly as they were acting on exactly the same traffic. The reverse connection from port 1 on COMBO6 (C6) in *kopr* to an Ethernet adapter in *jetel* was occasionally used for inspecting raw packet dumps.

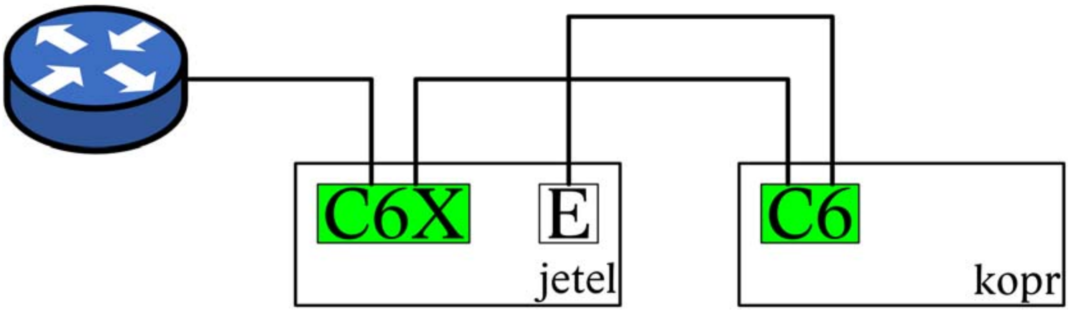


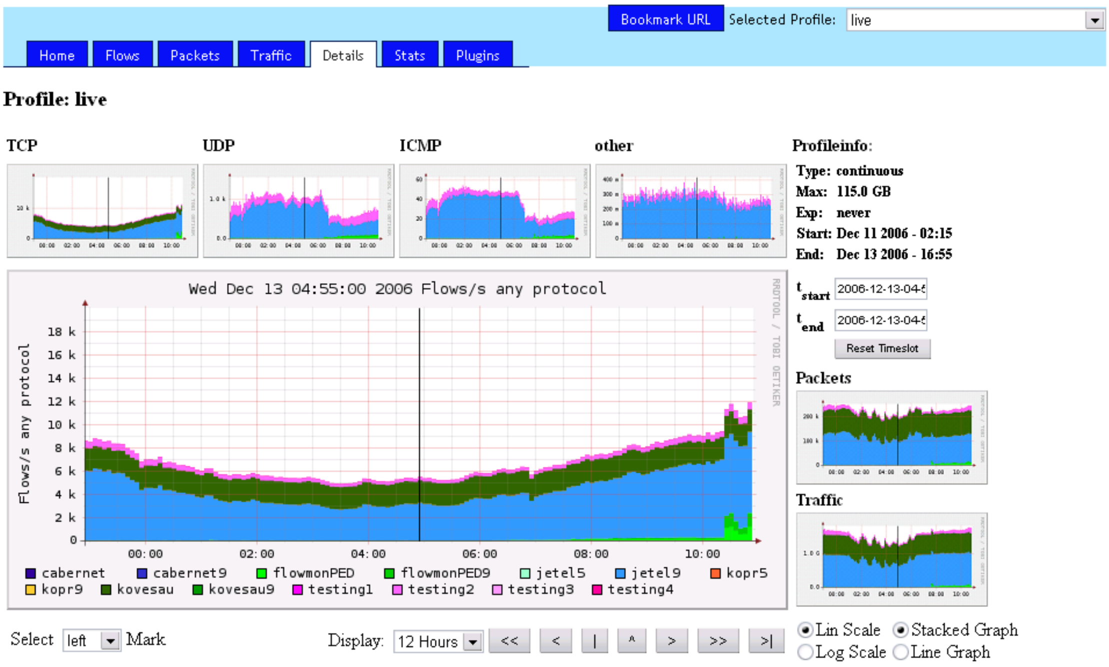
Figure 7: Interconnection of two FlowMon probes monitoring the Akamai link (kopr-prototype, jetel-final)

From all probes both v5 and v9 of NetFlow were exported. The data were processed by two suites of collectors and visualisation programs: FTAS [Kos04] and Nfdump/NfSen (the latter was selected for the JRA2 Security Toolset). Several compatibility problems and bugs in both the FlowMon probe and the processing suites were identified, including:

- NfSen used 1024-based coefficients for “mega” and “giga” in bitrates. The correct coefficient is 1000, so that e.g., 1 Mb/s = 1 000 000 b/s[1].
- FTAS mangled IPv6 addresses
- Wrong timestamps due to a race condition in the firmware
- Wrong values of the sampling rate reported by the firmware
- Corrupted flow records due to various firmware bugs

Figure 8 shows graphs produced by NfSen for a number of FlowMon probes (both lab tests and field deployments). Figure 9 shows an example of FTAS output from FlowMon data (cumulative volumes of most frequent services).

[1] http://en.wikipedia.org/wiki/Binary_prefix



Selected Profile: live

Profile: live

Profileinfo:

Type: continuous
Max: 115.0 GB
Exp: never
Start: Dec 11 2006 - 02:15
End: Dec 13 2006 - 16:55

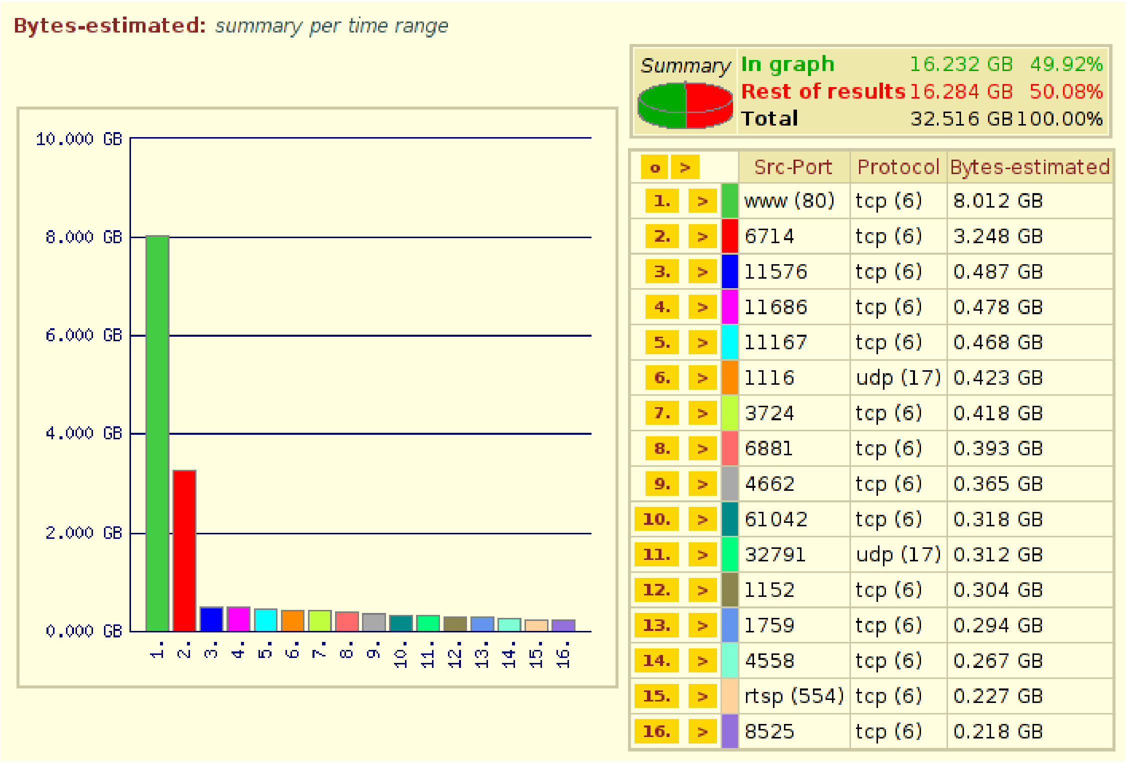
t_{start} 2006-12-13-04-5
t_{end} 2006-12-13-04-5

Select left Mark

Display: 12 Hours

● Lin Scale ● Stacked Graph
○ Log Scale ○ Line Graph

Figure 8: Sample NfSen "Details" page



Bytes-estimated: *summary per time range*

Summary			
	In graph	16.232 GB	49.92%
	Rest of results	16.284 GB	50.08%
	Total	32.516 GB	100.00%

o	>		Src-Port	Protocol	Bytes-estimated
1.	>	🟩	www (80)	tcp (6)	8.012 GB
2.	>	🟥	6714	tcp (6)	3.248 GB
3.	>	🟦	11576	tcp (6)	0.487 GB
4.	>	🟪	11686	tcp (6)	0.478 GB
5.	>	🟦	11167	tcp (6)	0.468 GB
6.	>	🟧	1116	udp (17)	0.423 GB
7.	>	🟩	3724	tcp (6)	0.418 GB
8.	>	🟥	6881	tcp (6)	0.393 GB
9.	>	⬜	4662	tcp (6)	0.365 GB
10.	>	🟦	61042	tcp (6)	0.318 GB
11.	>	🟩	32791	udp (17)	0.312 GB
12.	>	🟫	1152	tcp (6)	0.304 GB
13.	>	🟦	1759	tcp (6)	0.294 GB
14.	>	🟩	4558	tcp (6)	0.267 GB
15.	>	🟧	rtsp (554)	tcp (6)	0.227 GB
16.	>	🟪	8525	tcp (6)	0.218 GB

Figure 9: Sample output from FTAS: cumulative volumes of most frequent services

4.1.4 Summary

We found that the final version of the FlowMon probe is able to process unsampled bi-directional traffic on a GE link at line rate.

The compatibility of the probe with the Nfdump/NfSen NetFlow processing and visualisation suite is very good. We can therefore recommend this combination for use in the JRA2 Security Toolset.

4.2 GRNET

4.2.1 FlowMon setup

The FlowMon card was installed in a single CPU Xeon server with 1GB RAM running at 2.4 GHz. The host operating system is GNU/Linux (Slackware 10.1 with 2.4.29 kernel). As seen in Figure 10, the network part of the setup was relatively simple. FlowMon was used to monitor incoming and outgoing traffic of one of GRNET customers; The University of Crete. The FlowMon probe was connected to an unused port of the Cisco 6509 switch and the SPAN feature on this switch was configured. The interface to which the router was connected was set as the source for SPAN monitoring and the interface connected to FlowMon as the destination.

In addition, existing passive traffic analysis equipment was attached to port 1 on the FlowMon card, which allowed testing of the reliability of the repeater function between ports 0 and 1.

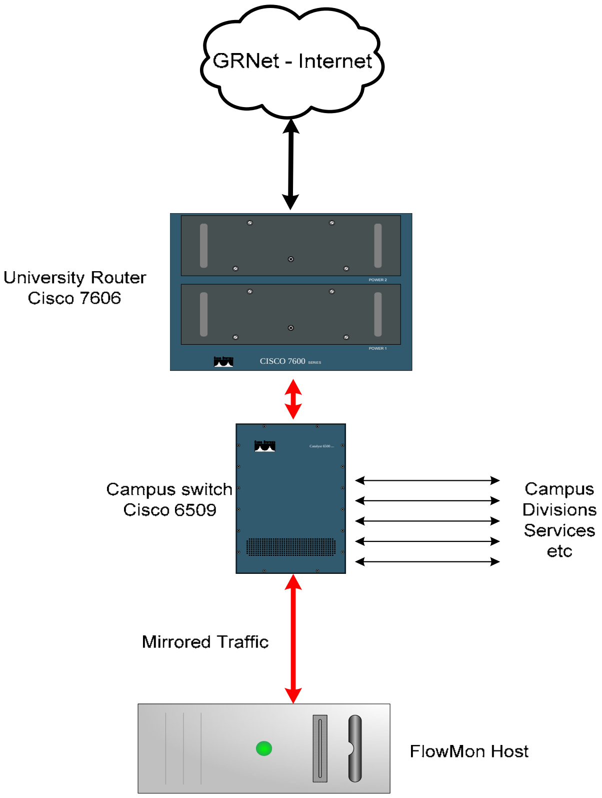


Figure 10: Test setup at GRNET

4.2.2 Measured Traffic

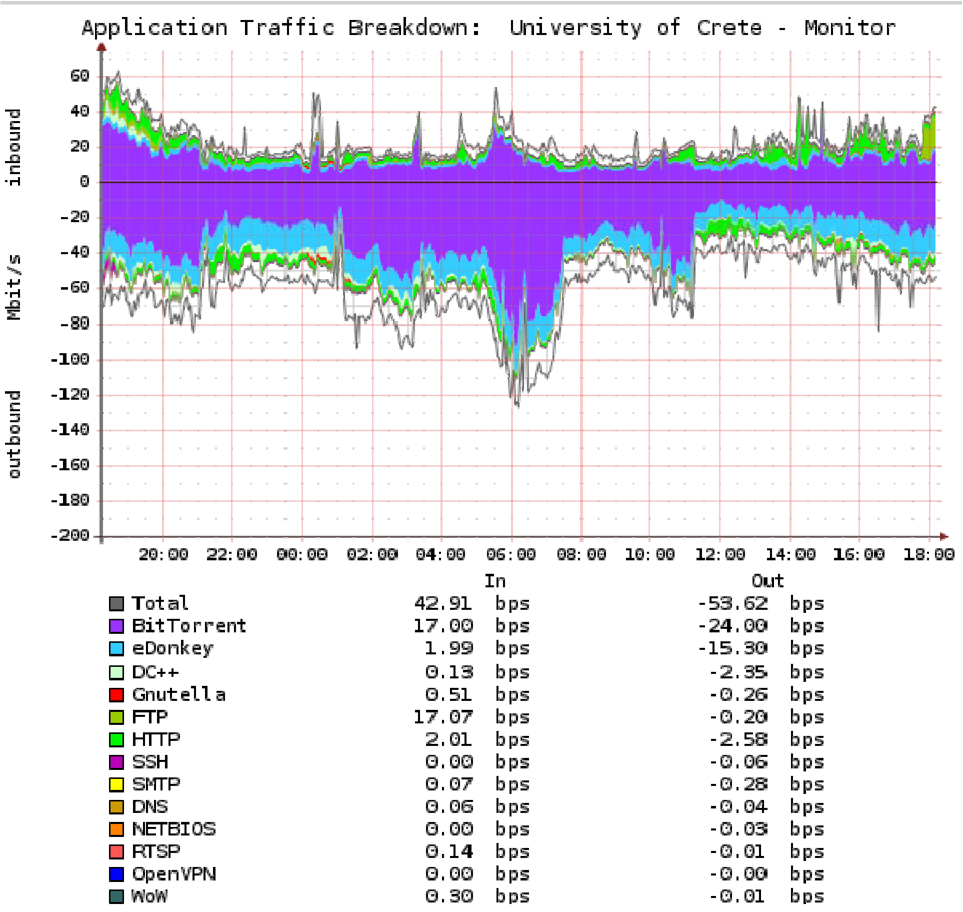
Due to technical restrictions and lack of ports, only the traffic of the main campus was monitored. This accounts, on average, for about 60 % of the total volume of university traffic. The graph in Figure 11 shows a representative one-day sample (units on the y-axis are Mbit/s). As can be seen, even the peak traffic rates are many times less than the maximum throughput of the

FlowMon probe. The same is true for the observed packet rates; 9000 packets per second on the average. The fraction of special flow types, such as IPv6, is also not significant.

4.2.3 Tests performed

In the GRNET environment, NetFlow plays a major role in network administration. Before the uplink to GRNET was recently upgraded to 1 Gbit/s, the IP flow information from the core router was used to drive a “penalty box” system that was limiting the bandwidth for rogue peer-to-peer programs, or huge FTP transfers.

While restricting users is no longer necessary, data on flows can provide valuable information on network utilisation. For this purpose the ntop[1] network traffic probe was installed and tested together with FlowMon.



	In		Out	
⬛ Total	42.91	bps	-53.62	bps
🟪 BitTorrent	17.00	bps	-24.00	bps
🟦 eDonkey	1.99	bps	-15.30	bps
□ DC++	0.13	bps	-2.35	bps
🟥 Gnutella	0.51	bps	-0.26	bps
🟩 FTP	17.07	bps	-0.20	bps
🟩 HTTP	2.01	bps	-2.58	bps
🟪 SSH	0.00	bps	-0.06	bps
🟨 SMTP	0.07	bps	-0.28	bps
🟫 DNS	0.06	bps	-0.04	bps
🟧 NETBIOS	0.00	bps	-0.03	bps
🟥 RTSP	0.14	bps	-0.01	bps
🟦 OpenVPN	0.00	bps	-0.00	bps
⬛ WoW	0.30	bps	-0.01	bps

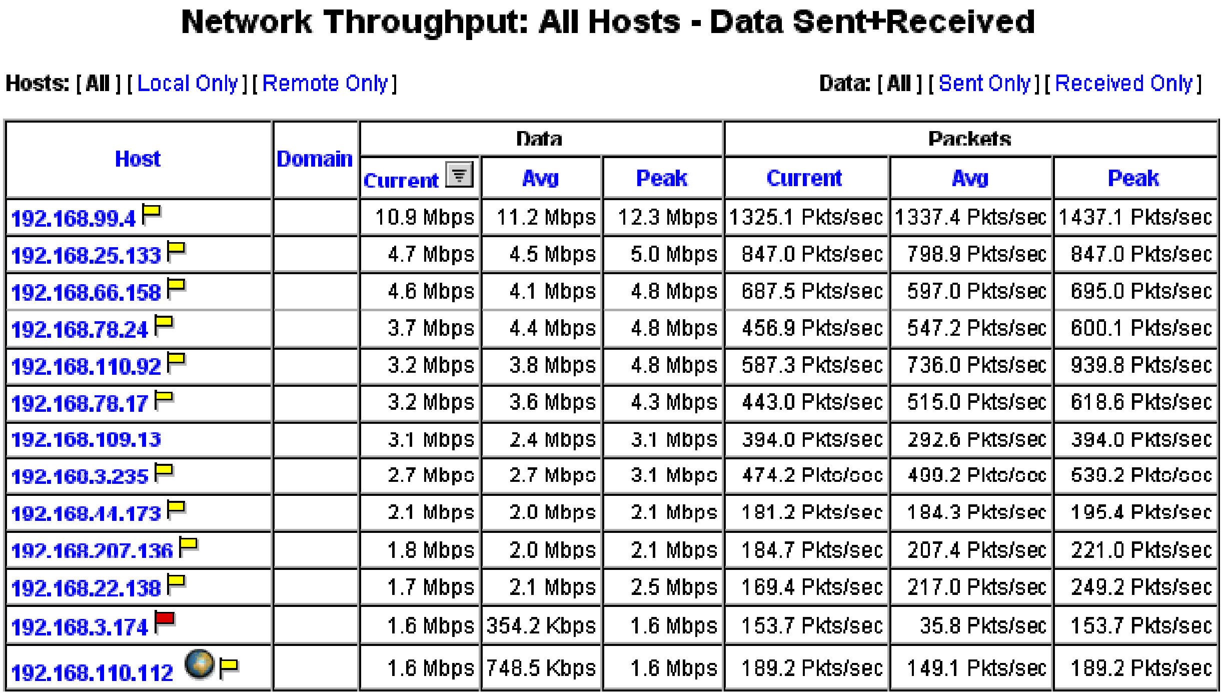
Figure 11: Sample of daily traffic

[1] http://www.ntop.org/

Figure 12 shows a sample ntop report anonymised with the “192.168” option of the FlowMon exporter to protect sensitive data.

A more interesting test was based on the comparison of data provided by the FlowMon card with those exported by the Cisco 6509 switch for the same traffic. For large-scale aggregated statistics, both sources produced similar results that were more accurate than the raw data from the router.

In order to make the test more specific, four ICMP Echo Requests from a host inside the campus were sent to an external network. After the active and inactive timeout periods expired, ntop was checked to see whether the corresponding flows are present. This test was repeated forty times under varying load conditions. However, in all cases the load was small compared to the total capacity of the link.



Network Throughput: All Hosts - Data Sent+Received

Hosts: [All] [Local Only] [Remote Only]   Data: [All] [Sent Only] [Received Only]

Host	Domain	Data			Packets		
		Current	Avg	Peak	Current	Avg	Peak
192.168.99.4		10.9 Mbps	11.2 Mbps	12.3 Mbps	1325.1 Pkts/sec	1337.4 Pkts/sec	1437.1 Pkts/sec
192.168.25.133		4.7 Mbps	4.5 Mbps	5.0 Mbps	847.0 Pkts/sec	798.9 Pkts/sec	847.0 Pkts/sec
192.168.66.158		4.6 Mbps	4.1 Mbps	4.8 Mbps	687.5 Pkts/sec	597.0 Pkts/sec	695.0 Pkts/sec
192.168.78.24		3.7 Mbps	4.4 Mbps	4.8 Mbps	456.9 Pkts/sec	547.2 Pkts/sec	600.1 Pkts/sec
192.168.110.92		3.2 Mbps	3.8 Mbps	4.8 Mbps	587.3 Pkts/sec	736.0 Pkts/sec	939.8 Pkts/sec
192.168.78.17		3.2 Mbps	3.6 Mbps	4.3 Mbps	443.0 Pkts/sec	515.0 Pkts/sec	618.6 Pkts/sec
192.168.109.13		3.1 Mbps	2.4 Mbps	3.1 Mbps	394.0 Pkts/sec	292.6 Pkts/sec	394.0 Pkts/sec
192.168.3.235		2.7 Mbps	2.7 Mbps	3.1 Mbps	474.2 Pkts/sec	499.2 Pkts/sec	539.2 Pkts/sec
192.168.44.173		2.1 Mbps	2.0 Mbps	2.1 Mbps	181.2 Pkts/sec	184.3 Pkts/sec	195.4 Pkts/sec
192.168.207.136		1.8 Mbps	2.0 Mbps	2.1 Mbps	184.7 Pkts/sec	207.4 Pkts/sec	221.0 Pkts/sec
192.168.22.138		1.7 Mbps	2.1 Mbps	2.5 Mbps	169.4 Pkts/sec	217.0 Pkts/sec	249.2 Pkts/sec
192.168.3.174		1.6 Mbps	354.2 Kbps	1.6 Mbps	153.7 Pkts/sec	35.8 Pkts/sec	153.7 Pkts/sec
192.168.110.112		1.6 Mbps	748.5 Kbps	1.6 Mbps	189.2 Pkts/sec	149.1 Pkts/sec	189.2 Pkts/sec

Figure 12: Ntop report with anonymised addresses

	4 pings with 32 bytes payload	36 MB FTP file transfer
Cisco NetFlow export	75 % detected	1 of 4 transfers done at 4 MB/s rate completely missed. The rest detected with slightly imprecise byte counts.
FlowMon card	100 % detected	All transfers detected correctly and exported immediately after the active timeout expired.

The results in the above table suggest that the FlowMon card is far more precise and faster in exporting the flows compared to the router.

4.2.4 Problems encountered

The size of the card was an initial problem. In order to fit it to our 5U server chassis, the PCI slot air guide had to be removed.

From the documentation it was assumed that the repeater function would be firmware independent. This was not to be the case.

Finally, the current flow analysis tools were not 100% compatible with NetFlow v9 as exported by the probe. Specifically, ntop failed to accept v9 templates from the exporter, reporting “wrong template size”, even after the latest CVS version was used. The custom-built “penalty box” system also had problems with handling v9 export. These problems could not be analysed in more detail, but both appeared to be caused by software incompatibilities and not by the FlowMon card itself.

4.2.5 Summary

The FlowMon card provides a means for exporting IP flows without imposing additional load on networking equipment. Also, the repeater feature of FlowMon allowed to connect two traffic analysis devices to a single interface, saving both CPU cycles and ports on the switch. More importantly, expensive splitter/repeater equipment is no longer necessary with this configuration.

All results obtained from the probe appeared to be accurate and no lossy sampling was needed, which means that even certain known single-packet attacks can be traced (theoretically).

Even though the current throughput allows for payload inspection, it may not be the case in the future. The FlowMon card (provided its development will continue and encompass the new open standards such as IPFIX) could become a useful and cost-effective tool in the area of IP flow analysis.

4.3 ISTF

4.3.1 FlowMon Setup

The simplified physical topology of the core network is shown in Figure 13. All traffic in the network passes through the central switch (Cisco 3550-12T) and is mirrored to the FlowMon probe. The probe then exports the generated NetFlow data to a monitoring server. Using the repeater function, it also sends the raw packet stream to the same server, which is used when a more detailed analysis is needed.

The FlowMon probe is installed in a Dell 1600SC server (2x 2 GHz Intel P4 Xeon CPU, 1.75 GB registered ECC SDRAM PC2100, 73.5 GB + 36.7 GB SCSI Ultra 320 10000 RPM HDD, Intel 82540EM Gigabit Ethernet Controller). As the host operating system, two Linux distributions were used: Red Hat Linux 9.0 and, more recently, Slackware Linux 11.0. On RH9, the tests were performed with kernels 2.4.20-31.9, 2.4.32 and 2.6.16.11, both in uniprocessor and symmetric multiprocessing mode. On Slackware, 2.4.33.3 and 2.6.17.13 in uniprocessor mode only were used.

Considering the logical topology, the setup is actually slightly more complicated as it involves some filtering and selective mirroring. For example, the customers' connections terminate on one of the routers, so part of the traffic might pass twice across the core switch, being counted twice in the monitoring system.

4.3.2 Monitored Traffic

Depending on the particular time of day and/or day of week, the traffic rate varied between 50 and 250 Mb/s, with peaks above 300 Mb/s. The average was around 150 Mb/s. IPv6 traffic accounts for about 0.04 % of the total volume and multicast for about 0.1 %. Over 80 % packets are either 64-128 bytes long or around 1500 bytes.

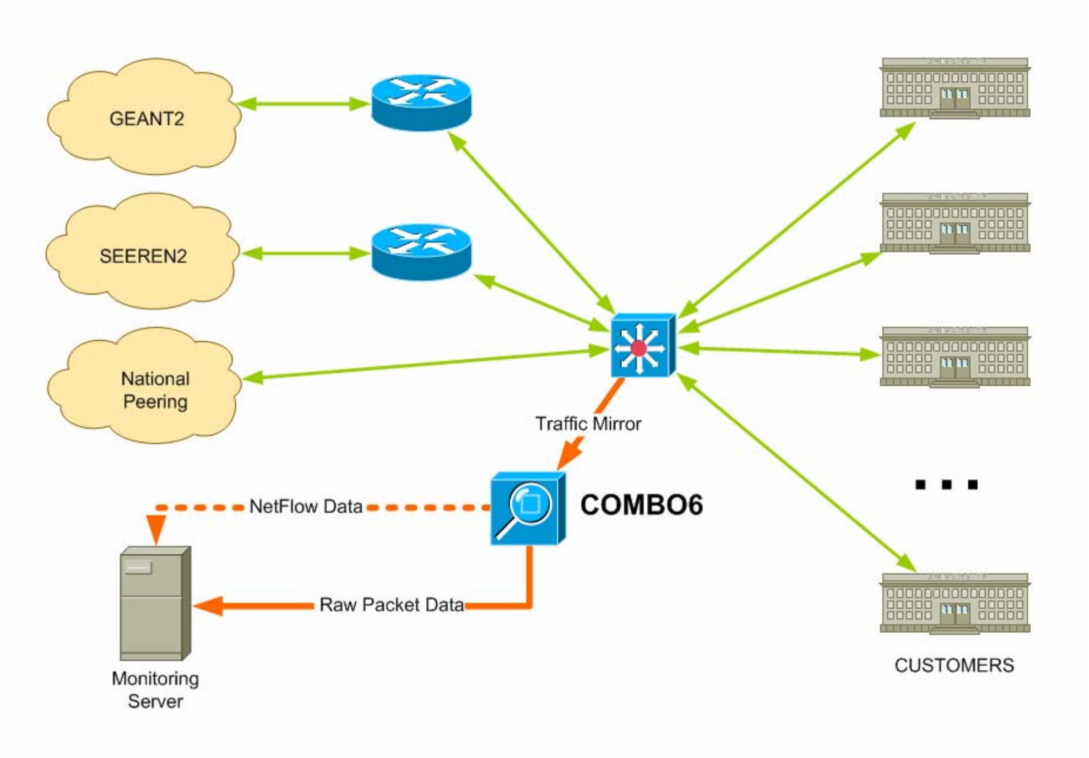


Figure 13: Test setup at ISTF

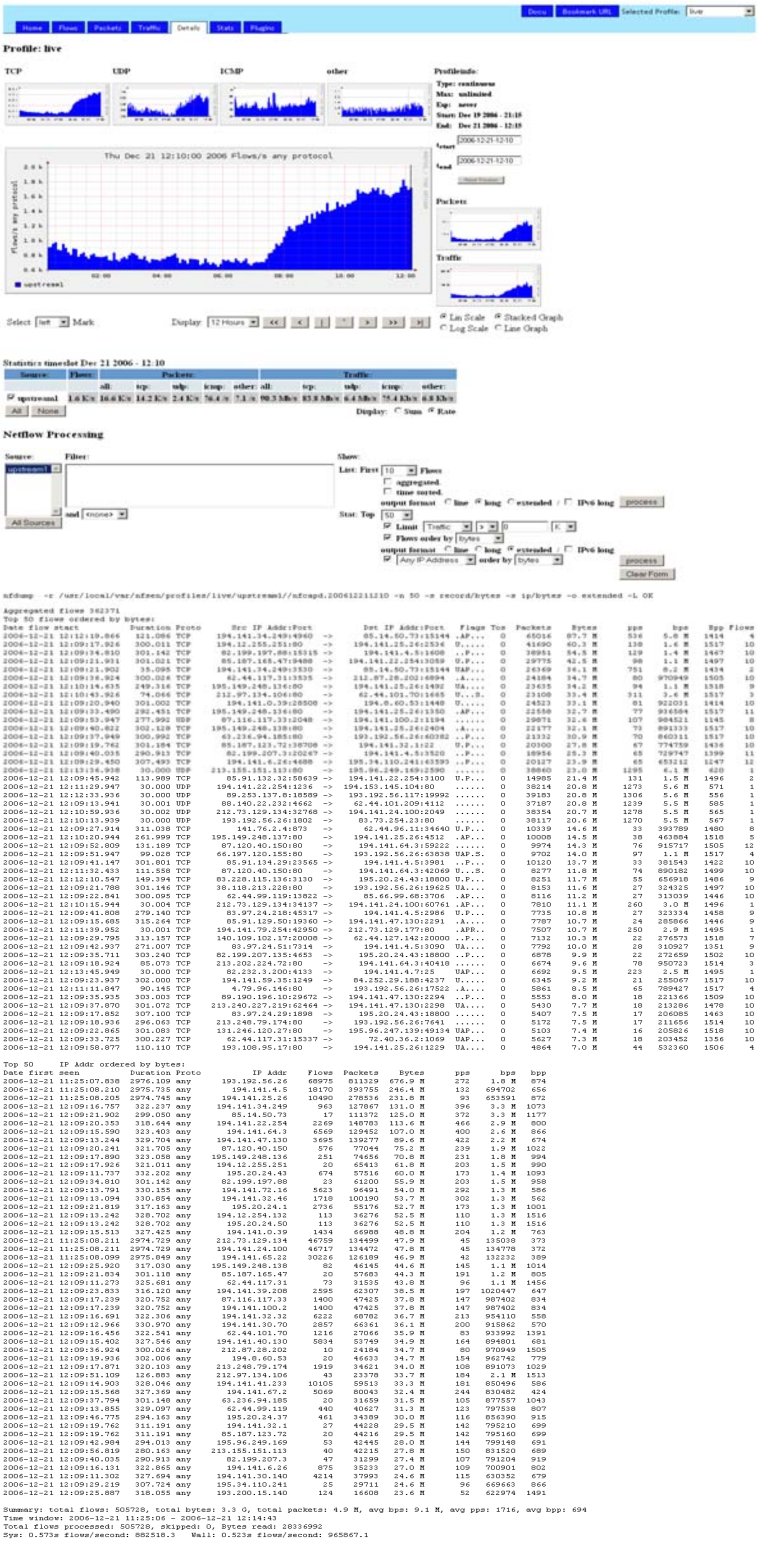
4.3.3 Tests Performed

ntop and NfSen were used for visualisation and analysis of the NetFlow data. While ntop tends to be more user-friendly and visually attractive, NfSen is a “heavy-duty” tool targeted at dedicated network professionals. Two screenshots are shown below: Figure 14 is from NfSen and Figure 15 from ntop.

Generally, the whole process of installing the FlowMon probe and its software was smooth and flawless on all uniprocessor kernels, with only a few minor problems. The SMP configurations proved to be rather unstable, but this was to be expected as the documentation warns against such a setup.

The documentation deserves a special mention: It is simple but very helpful, with a nice step-by-step layout.

Before FlowMon, packet sniffing was used (either with ntop or, in some cases, with tcpdump and similar tools) when more detailed information was needed. Although it is certainly possible to export NetFlow from routers, it puts quite a burden on them, especially considering the already existing high load resulting from complex routing rules and policies. Furthermore, the traffic would have to be mirrored anyway, which means wasting resources.



Profile: live

Profileinfo:
Type: continuous
Max: unlimited
Exp: never
Start: Dec 19 2006 - 21:15
End: Dec 21 2006 - 12:15
tstart 2006-12-21-12-10
tend 2006-12-21-12-10

◉ Lin Scale ◉ Stacked Graph
○ Log Scale ○ Line Graph

Statistics timeslot Dec 21 2006 - 12:10

Source:	Flows:	Packets:					Traffic:				
		all:	tcp:	udp:	icmp:	other:	all:	tcp:	udp:	icmp:	other:
☑ upstream1	1.6 K/s	16.6 K/s	14.2 K/s	2.4 K/s	76.4 /s	7.1 /s	90.3 Mb/s	83.8 Mb/s	6.4 Mb/s	75.4 Kb/s	6.8 Kb/s

Display: ○ Sum ◉ Rate

Netflow Processing

Source: upstream1
Filter:
and <none>

Show:
List: First 10 Flows
□ aggregated
□ time sorted
output format ○ line ◉ long ○ extended / □ IPv6 long
Stat: Top 50
☑ Limit Traffic > 0 K
☑ Flows order by bytes
output format ○ line ○ long ◉ extended / □ IPv6 long
☑ Any IP Address order by bytes

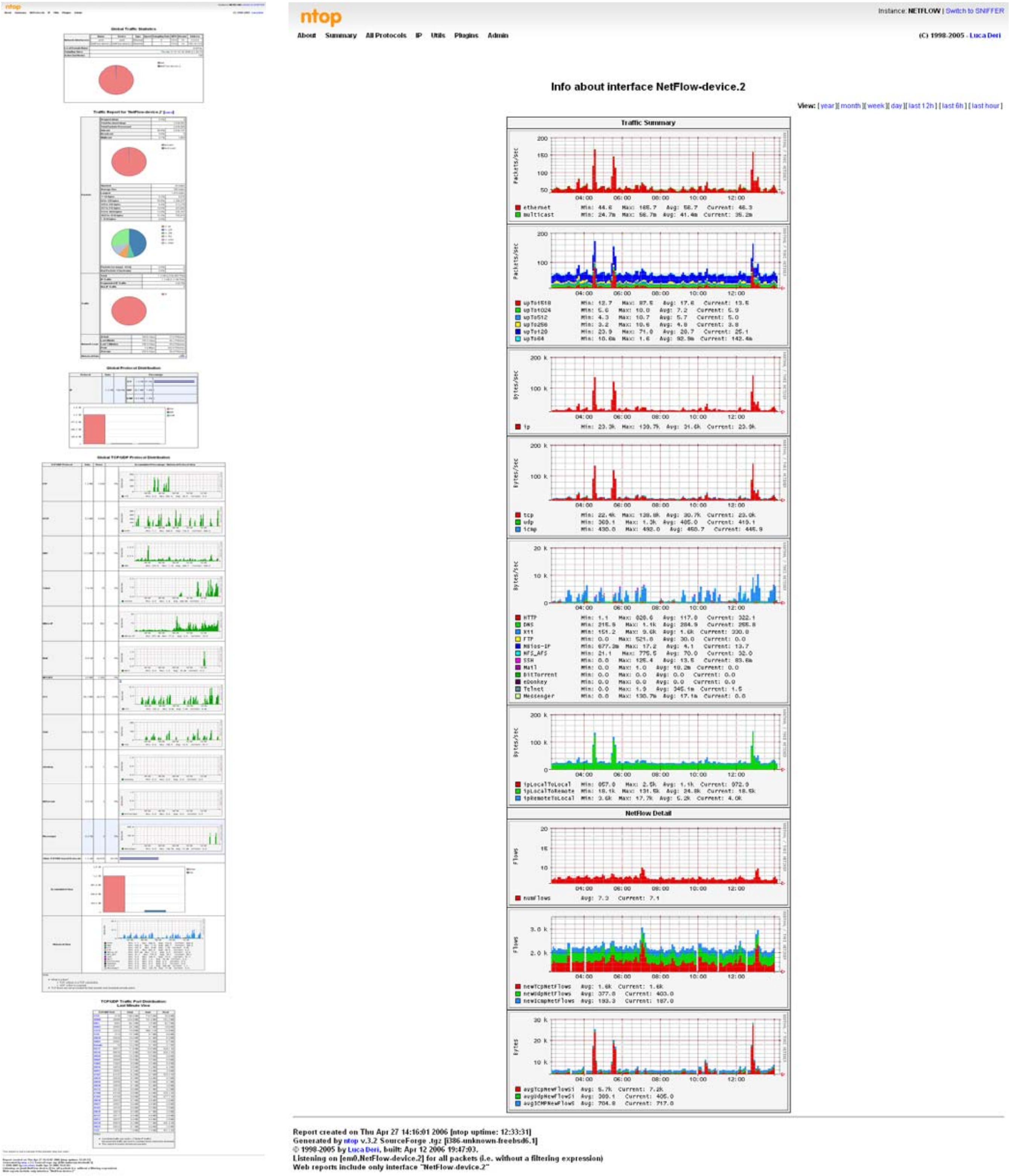
```
nfdump  -r /usr/local/var/nfsen/profiles/live/upstream1//nfcapd.200612211210 -n 50 -s record/bytes -s ip/bytes -o extended -L OK

Aggregated flows 362371
Top 50 flows ordered by bytes:
Date flow start          Duration Proto      Src IP Addr:Port          Dst IP Addr:Port   Flags Tos  Packets    Bytes      pps      bps    Bpp Flows
2006-12-21 12:12:10.866   121.086 TCP     194.141.34.249:4960  ->      85.14.50.73:15144 .AP...   0    65016   87.7 M      536    5.8 M   1414     4
2006-12-21 12:09:17.926   300.011 TCP     194.12.255.251:80    ->    194.141.25.26:2536  U.....   0    41690   60.3 M      138    1.6 M   1517    10
2006-12-21 12:09:34.810   301.142 TCP     82.199.197.88:15315 ->      194.141.4.5:1608  ..P...   0    38951   54.5 M      129    1.4 M   1467    10
2006-12-21 12:09:21.931   301.021 TCP     85.187.165.47:9488   ->   194.141.22.254:3059  U.P...   0    29775   42.5 M       98    1.1 M   1497    10
2006-12-21 12:09:21.902    35.095 TCP     194.141.34.249:3530  ->      85.14.50.73:15144 UAP...   0    26369   36.1 M      751    8.2 M   1434     2
2006-12-21 12:09:36.924   300.026 TCP       62.44.117.31:3535  ->    212.87.28.202:6894  .A....   0    24184   34.7 M       80   970949   1505    10
2006-12-21 12:10:14.635   249.316 TCP    195.149.248.136:80    ->    194.141.25.26:1492  UA....   0    23635   34.2 M       94    1.1 M   1518     9
2006-12-21 12:10:43.926    74.066 TCP     212.97.134.106:80    ->     62.44.101.70:1665  U...S.   0    23108   33.4 M      311    3.6 M   1517     3
2006-12-21 12:09:20.940   301.002 TCP      194.141.0.39:28508 ->       194.8.60.53:1448  U.....   0    24523   33.1 M       81   922031   1414    10
2006-12-21 12:09:33.490   292.451 TCP    195.149.248.136:80    ->    194.141.25.26:1350  .AP...   0    22558   32.7 M       77   936584   1517    11
2006-12-21 12:09:53.947   277.992 UDP      87.116.117.33:2048  ->    194.141.100.2:1194  ......   0    29871   32.6 M      107   984521   1145     8
2006-12-21 12:09:40.822   302.128 TCP    195.149.248.138:80    ->    194.141.25.26:2404  .A....   0    22177   32.1 M       73   891333   1517    10
2006-12-21 12:09:37.949   300.992 TCP      63.236.94.185:80    ->    193.192.56.26:60382 ..P...   0    21332   30.9 M       70   860311   1517    10
2006-12-21 12:09:19.762   301.184 TCP     85.187.123.72:38708 ->      194.141.32.1:22    U.P...   0    20300   27.8 M       67   774759   1436    10
2006-12-21 12:09:40.035   290.913 TCP      82.199.207.3:20247 ->       194.141.4.5:3520  ..P...   0    18956   25.3 M       65   729747   1399    11
2006-12-21 12:09:29.450   307.493 TCP       194.141.6.26:4688  ->   195.34.110.241:63593 ..P...   0    20127   23.9 M       65   653212   1247    12
2006-12-21 12:13:36.938    30.000 UDP    213.155.151.113:80    ->   195.96.249.169:2590  ......   0    38860   23.0 M     1295    6.1 M    620     1
2006-12-21 12:09:45.942   113.989 TCP      85.91.132.32:58639 ->   194.141.22.254:3100  U.P...   0    14985   21.4 M      131    1.5 M   1496     2
2006-12-21 12:11:29.947    30.000 UDP     194.141.22.254:1236  ->  194.153.145.104:80    ......   0    38214   20.8 M     1273    5.6 M    571     1
2006-12-21 12:12:33.936    30.000 UDP      89.253.137.8:18589 ->   193.192.56.117:19992 ......   0    39183   20.8 M     1306    5.6 M    556     1
2006-12-21 12:09:13.941    30.001 UDP     88.140.22.232:4662   ->    62.44.101.209:4112  ......   0    37187   20.8 M     1239    5.5 M    585     1
2006-12-21 12:10:59.936    30.002 UDP    212.73.129.134:32768 ->   194.141.24.100:2049  ......   0    38354   20.7 M     1278    5.5 M    565     1
2006-12-21 12:10:13.939    30.000 UDP     193.192.56.26:1802   ->     83.73.254.23:80    ......   0    38117   20.6 M     1270    5.5 M    567     1
2006-12-21 12:09:27.914   311.038 TCP        141.76.2.4:873    ->     62.44.96.11:34640 U.P...   0    10339   14.6 M       33   393789   1480     8
2006-12-21 12:10:20.944   261.999 TCP    195.149.248.137:80    ->    194.141.25.26:4512  .AP...   0    10008   14.5 M       38   463884   1518     5
2006-12-21 12:09:52.809   131.189 TCP     87.120.40.150:80     ->    194.141.64.3:59222  ......   0     9974   14.3 M       76   915717   1505    12
2006-12-21 12:09:51.947    99.028 TCP    66.197.120.155:80     ->    193.192.56.26:63838 UAP.S.   0     9702   14.0 M       97    1.1 M   1517     4
2006-12-21 12:09:41.147   301.801 TCP      85.91.134.29:23565 ->      194.141.4.5:3981  ..P...   0    10120   13.7 M       33   381543   1422    10
2006-12-21 12:11:32.433   111.558 TCP     87.120.40.150:80     ->    194.141.64.3:42069  U...S.   0     8277   11.8 M       74   890182   1499    10
2006-12-21 12:12:10.547   149.394 TCP    83.228.115.136:3130   ->    195.20.24.43:18800  U.P...   0     8251   11.7 M       55   656918   1486     9
2006-12-21 12:09:21.788   301.146 TCP    38.118.213.228:80     ->    193.192.56.26:19625 UA....   0     8153   11.6 M       27   324325   1497    10
2006-12-21 12:09:22.841   300.095 TCP      62.44.99.119:13822 ->      85.66.99.68:3706  .AP...   0     8116   11.2 M       27   313039   1446    10
2006-12-21 12:10:15.944    30.004 TCP    212.73.129.134:34137 ->   194.141.24.100:60761 .AP...   0     7810   11.1 M      260    3.0 M   1496     1
2006-12-21 12:09:41.808   279.140 TCP      83.97.24.218:45317 ->      194.141.4.5:2986  U.P...   0     7735   10.8 M       27   323334   1458     9
2006-12-21 12:09:15.685   315.264 TCP      85.91.129.50:19360 ->   194.141.47.130:2291  .A....   0     7787   10.7 M       24   285866   1446     9
2006-12-21 12:11:39.952    30.001 TCP     194.141.79.254:42950 ->   212.73.129.177:80    .APR..   0     7507   10.7 M      250    2.9 M   1495     1
2006-12-21 12:09:29.795   313.157 TCP    140.109.102.17:20008 ->    62.44.127.142:20000 ..P...   0     7132   10.3 M       22   276573   1518     7
2006-12-21 12:09:42.937   271.007 TCP       83.97.24.51:7314   ->      194.141.4.5:3090  UA....   0     7792   10.0 M       28   310927   1351     9
2006-12-21 12:09:35.711   303.240 TCP    82.199.207.135:4653   ->    195.20.24.43:18800 ..P...   0     6878    9.9 M       22   272659   1502    10
2006-12-21 12:09:18.924    85.073 TCP    213.202.224.72:80     ->    194.141.64.3:40418  ......   0     6674    9.6 M       78   950723   1514     3
2006-12-21 12:13:45.949    30.000 TCP      82.232.3.200:4133   ->      194.141.4.7:25    UAP...   0     6692    9.5 M      223    2.5 M   1495     1
2006-12-21 12:09:23.937   302.000 TCP     194.141.59.35:1249   ->   84.252.29.188:4237  U.....   0     6345    9.2 M       21   255067   1517    10
2006-12-21 12:11:11.847    90.145 TCP       4.79.96.146:80     ->    193.192.56.26:17522 .A....   0     5861    8.5 M       65   789427   1517     4
2006-12-21 12:09:35.935   303.003 TCP     89.190.196.10:29672 ->   194.141.47.130:2294  ..P...   0     5553    8.0 M       18   221366   1509    10
2006-12-21 12:09:37.870   301.072 TCP   213.240.227.219:62464 ->   194.141.47.130:2298  UA....   0     5430    7.7 M       18   213286   1478    10
2006-12-21 12:09:17.852   307.100 TCP       83.97.24.29:1898   ->    195.20.24.43:18800 ......   0     5407    7.5 M       17   206085   1463    10
2006-12-21 12:09:18.936   296.063 TCP    213.248.79.174:80     ->    193.192.56.26:7641  ......   0     5172    7.5 M       17   211656   1514    10
2006-12-21 12:09:22.865   301.083 TCP    131.246.120.27:80     ->  195.96.247.139:49134 UAP...   0     5103    7.4 M       16   205826   1518    10
2006-12-21 12:09:33.725   300.227 TCP      62.44.117.31:15337 ->       72.40.36.2:1069  UAP...   0     5627    7.3 M       18   203452   1356    10
2006-12-21 12:09:58.877   110.110 TCP     193.108.95.17:80     ->    194.141.25.26:1229  UA....   0     4864    7.0 M       44   532360   1506     4

Top 50     IP Addr ordered by bytes:
Date first seen          Duration Proto          IP Addr    Flows  Packets    Bytes      pps      bps   bpp
2006-12-21 11:25:07.838  2976.109 any      193.192.56.26    68975   811329  676.9 M      272    1.8 M   874
2006-12-21 11:25:08.210  2975.735 any        194.141.4.5    18170   393755  246.4 M      132   694702   656
2006-12-21 11:25:08.205  2974.745 any      194.141.25.26    10490   278536  231.8 M       93   653591   872
2006-12-21 12:09:16.757   322.237 any     194.141.34.249      963   127867  131.0 M      396    3.3 M  1073
2006-12-21 12:09:21.902   299.050 any        85.14.50.73       17   111372  125.0 M      372    3.3 M  1177
2006-12-21 12:09:20.353   318.644 any     194.141.22.254     2269   148783  113.6 M      466    2.9 M   800
2006-12-21 12:09:15.590   323.403 any       194.141.64.3     6569   129452  107.0 M      400    2.6 M   866
2006-12-21 12:09:13.244   329.704 any     194.141.47.130     3695   139277   89.6 M      422    2.2 M   674
2006-12-21 12:09:20.241   321.705 any      87.120.40.150      576    77044   75.2 M      239    1.9 M  1022
2006-12-21 12:09:17.890   323.058 any    195.149.248.136      251    74656   70.8 M      231    1.8 M   994
2006-12-21 12:09:17.926   321.011 any     194.12.255.251       20    65413   61.8 M      203    1.5 M   990
2006-12-21 12:09:11.737   332.202 any       195.20.24.43      674    57516   60.0 M      173    1.4 M  1093
2006-12-21 12:09:34.810   301.142 any      82.199.197.88       23    61200   55.9 M      203    1.5 M   958
2006-12-21 12:09:13.791   330.155 any      194.141.72.16     5623    96491   54.0 M      292    1.3 M   586
2006-12-21 12:09:13.094   330.854 any      194.141.32.46     1718   100190   53.7 M      302    1.3 M   562
2006-12-21 12:09:21.819   317.163 any        195.20.24.1     2736    55176   52.7 M      173    1.3 M  1001
2006-12-21 12:09:13.242   328.702 any     194.12.254.132      113    36276   52.5 M      110    1.3 M  1516
2006-12-21 12:09:13.242   328.702 any       195.20.24.50      113    36276   52.5 M      110    1.3 M  1516
2006-12-21 12:09:15.513   327.425 any       194.141.0.39     1434    66988   48.8 M      204    1.2 M   763
2006-12-21 11:25:08.211  2974.729 any     212.73.129.134    46759   134499   47.9 M       45   135038   373
2006-12-21 11:25:08.211  2974.729 any     194.141.24.100    46717   134472   47.8 M       45   134778   372
2006-12-21 11:25:08.099  2975.849 any      194.141.65.22    30226   126189   46.9 M       42   132232   389
2006-12-21 12:09:25.920   317.030 any    195.149.248.138       82    46145   44.6 M      145    1.1 M  1014
2006-12-21 12:09:21.834   301.118 any      85.187.165.47       20    57683   44.3 M      191    1.2 M   805
2006-12-21 12:09:11.273   325.681 any       62.44.117.31       73    31535   43.8 M       96    1.1 M  1456
2006-12-21 12:09:23.833   316.120 any     194.141.39.208     2595    62307   38.5 M      197  1020447   647
2006-12-21 12:09:17.239   320.752 any      87.116.117.33     1400    47425   37.8 M      147   987402   834
2006-12-21 12:09:17.239   320.752 any      194.141.100.2     1400    47425   37.8 M      147   987402   834
2006-12-21 12:09:16.691   322.306 any      194.141.32.32     6222    68782   36.7 M      213   954110   558
2006-12-21 12:09:12.966   330.970 any      194.141.30.70     2857    66361   36.1 M      200   915862   570
2006-12-21 12:09:16.456   322.541 any       62.44.101.70     1216    27066   35.9 M       83   933992  1391
2006-12-21 12:09:15.402   327.546 any     194.141.40.130     5834    53749   34.9 M      164   894801   681
2006-12-21 12:09:36.924   300.026 any      212.87.28.202       10    24184   34.7 M       80   970949  1505
2006-12-21 12:09:19.936   302.006 any        194.8.60.53       20    46633   34.7 M      154   962742   779
2006-12-21 12:09:17.871   320.103 any     213.248.79.174     1919    34621   34.0 M      108   891073  1029
2006-12-21 12:09:51.109   126.883 any     212.97.134.106       43    23378   33.7 M      184    2.1 M  1513
2006-12-21 12:09:14.903   328.046 any     194.141.41.233    10105    59513   33.3 M      181   850496   586
2006-12-21 12:09:15.568   327.369 any       194.141.67.2     5069    80043   32.4 M      244   830482   424
2006-12-21 12:09:37.794   301.148 any      63.236.94.185       20    31659   31.5 M      105   877557  1043
2006-12-21 12:09:13.855   329.097 any       62.44.99.119      440    40627   31.3 M      123   797538   807
2006-12-21 12:09:46.775   294.163 any       195.20.24.37      461    34389   30.0 M      116   856390   915
2006-12-21 12:09:19.762   311.191 any       194.141.32.1       27    44228   29.5 M      142   795210   699
2006-12-21 12:09:19.762   311.191 any      85.187.123.72       20    44216   29.5 M      142   795160   699
2006-12-21 12:09:42.984   294.013 any     195.96.249.169       53    42445   28.0 M      144   799148   691
2006-12-21 12:09:56.819   280.163 any    213.155.151.113       40    42215   27.8 M      150   831520   689
2006-12-21 12:09:40.035   290.913 any       82.199.207.3       47    31299   27.4 M      107   791204   919
2006-12-21 12:09:16.131   322.865 any       194.141.6.26      875    35233   27.0 M      109   700901   802
2006-12-21 12:09:11.302   327.694 any     194.141.30.140     4214    37993   24.6 M      115   630352   679
2006-12-21 12:09:29.219   307.724 any     195.34.110.241       25    29711   24.6 M       96   669663   866
2006-12-21 12:09:25.887   318.055 any     193.200.15.140      124    16608   23.6 M       52   622974  1491

Summary: total flows: 505728, total bytes: 3.3 G, total packets: 4.9 M, avg bps: 9.1 M, avg pps: 1716, avg bpp: 694
Time window: 2006-12-21 11:25:06 - 2006-12-21 12:14:43
Total flows processed: 505728, skipped: 0, Bytes read: 28336992
Sys: 0.573s flows/second: 882518.3   Wall: 0.523s flows/second: 965867.1
```

Figure 14: Sample output from NfSen



ntop
Instance: NETFLOW | Switch to SNIFFER
About Summary All Protocols IP Utils Plugins Admin
(C) 1998-2005 - Luca Deri

Info about interface NetFlow-device.2

View: [year][month][week][day][last 12h] [last 6h] [last hour]

Traffic Summary

```
ethernet          Min: 44.6    Max: 165.7   Avg: 58.7    Current: 46.3
multicast         Min: 24.7m   Max: 56.7m   Avg: 41.4m   Current: 35.2m

upTo1518          Min: 12.7    Max: 87.5    Avg: 17.6    Current: 13.5
upTo1024          Min: 5.6     Max: 10.0    Avg: 7.2     Current: 5.9
upTo512           Min: 4.3     Max: 10.7    Avg: 5.7     Current: 5.0
upTo256           Min: 3.2     Max: 10.6    Avg: 4.8     Current: 3.8
upTo128           Min: 23.9    Max: 71.0    Avg: 20.7    Current: 25.1
upTo64            Min: 10.6m   Max: 1.6     Avg: 92.9m   Current: 142.4m

ip                Min: 23.3k   Max: 139.7k  Avg: 31.6k   Current: 23.9k

tcp               Min: 22.4k   Max: 138.8k  Avg: 30.7k   Current: 23.0k
udp               Min: 309.1   Max: 1.3k    Avg: 405.0   Current: 419.1
icmp              Min: 430.0   Max: 492.0   Avg: 460.7   Current: 445.9

HTTP              Min: 1.1     Max: 828.6   Avg: 117.0   Current: 322.1
DNS               Min: 215.9   Max: 1.1k    Avg: 284.9   Current: 255.8
X11               Min: 151.2   Max: 9.6k    Avg: 1.6k    Current: 330.0
FTP               Min: 0.0     Max: 521.0   Avg: 30.0    Current: 0.0
NBios-IP          Min: 677.3m  Max: 17.2    Avg: 4.1     Current: 13.7
NFS_AFS           Min: 21.1    Max: 776.5   Avg: 70.0    Current: 32.0
SSH               Min: 0.0     Max: 125.4   Avg: 13.6    Current: 83.6m
Mail              Min: 0.0     Max: 1.0     Avg: 10.2m   Current: 0.0
BitTorrent        Min: 0.0     Max: 0.0     Avg: 0.0     Current: 0.0
eDonkey           Min: 0.0     Max: 0.0     Avg: 0.0     Current: 0.0
Telnet            Min: 0.0     Max: 1.9     Avg: 346.1m  Current: 1.5
Messenger         Min: 0.0     Max: 130.7m  Avg: 17.1m   Current: 0.0

ipLocalToLocal    Min: 857.0   Max: 2.5k    Avg: 1.1k    Current: 972.9
ipLocalToRemote   Min: 18.1k   Max: 131.5k  Avg: 24.8k   Current: 18.5k
ipRemoteToLocal   Min: 3.6k    Max: 17.7k   Avg: 5.2k    Current: 4.0k
```

NetFlow Detail

```
numFlows          Avg: 7.3     Current: 7.1

newTcpNetFlows    Avg: 1.6k    Current: 1.6k
newUdpNetFlows    Avg: 377.8   Current: 403.0
newIcmpNetFlows   Avg: 193.3   Current: 187.0

avgTcpNewFlowSi   Avg: 5.7k    Current: 7.2k
avgUdpNewFlowSi   Avg: 309.1   Current: 405.0
avgICMPNewFlowS   Avg: 704.8   Current: 717.0
```

Report created on Thu Apr 27 14:16:01 2006 [ntop uptime: 12:33:31]
Generated by ntop v.3.2 SourceForge .tgz [i386-unknown-freebsd6.1]
© 1998-2005 by Luca Deri, built: Apr 12 2006 19:47:03.
Listening on [em0,NetFlow-device.2] for all packets (i.e. without a filtering expression)
Web reports include only interface "NetFlow-device.2"

Figure 15: Sample output from ntop

4.3.4 Summary

The FlowMon probe was a very welcome addition to the monitoring toolbox. It made continuous monitoring possible, not just “when needed”.

With respect to future improvements, the primary suggestion is to solve the problems with SMP. Another would be to port the software to the BSD family of Unices.

4.4 SURFnet

4.4.1 FlowMon Setup

SURFnet deployed the FlowMon probe in the office network. A Cisco Catalyst 6509 with Supervisor Engine 720 connects the internal network to the SURFnet6[1] network via a 1 Gb/s link. The Cisco Catalyst 6509 has a mirror port that passes a copy of all traffic to the FlowMon probe, but also exports its own NetFlow v5 data. Another mirror port is used for LANGuardian[2]. The setup, with additional details, is shown in Figure 16.

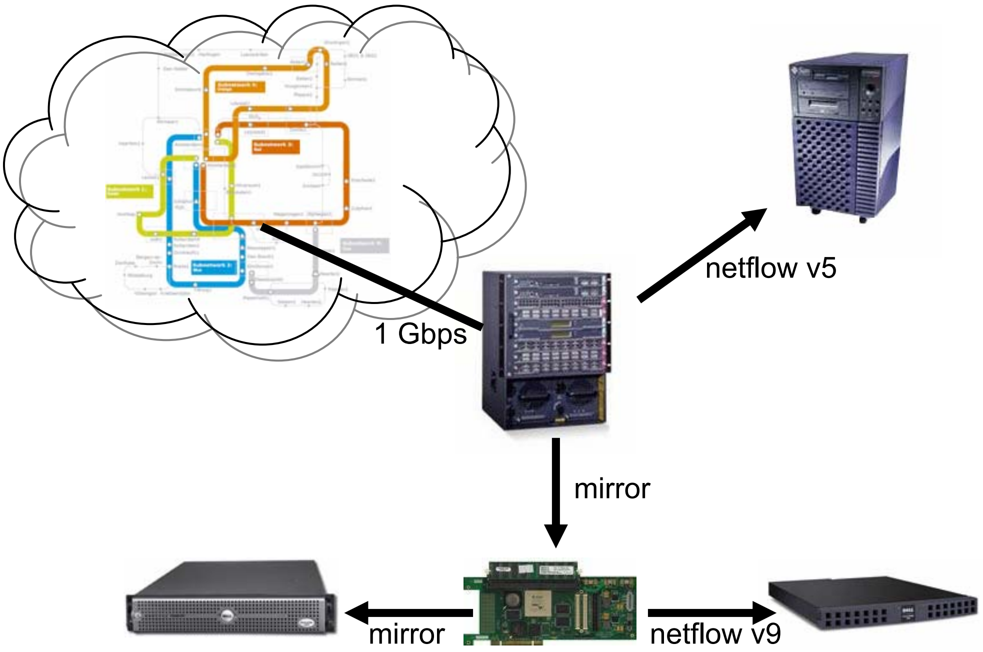


Figure 16: Test setup at SURFnet

[1] http://www.surfnet.nl/info/en/network/home.jsp
[2] http://www.netforttechnologies.com/languardian.html

The probe exports NetFlow v9 data to a monitoring server. The repeater function was used for sending raw packet stream for further analysis whenever more information was needed.

The FlowMon probe is currently installed in a old Dell PowerEdge 350 server (Intel PIII 850 MHz CPU, 1 GB RAM, 40 GB + 300 GB ATA disks and 2 Fast Ethernet interfaces). The operating system is Debian 4.0 and for the tests kernels 2.4.27, 2.6.16 and 2.6.17 were used, together with both the old NETFLOW (01_03 and 01_04) and new flowmon (1.0.0, 1.0.2 and 1.0.3) packages.

4.4.2 Monitored Traffic

Figure 17 provides some background regarding the volume of traffic flowing between the SURFnet office and SURFnet6 network. The peaks are due to off-site backup.

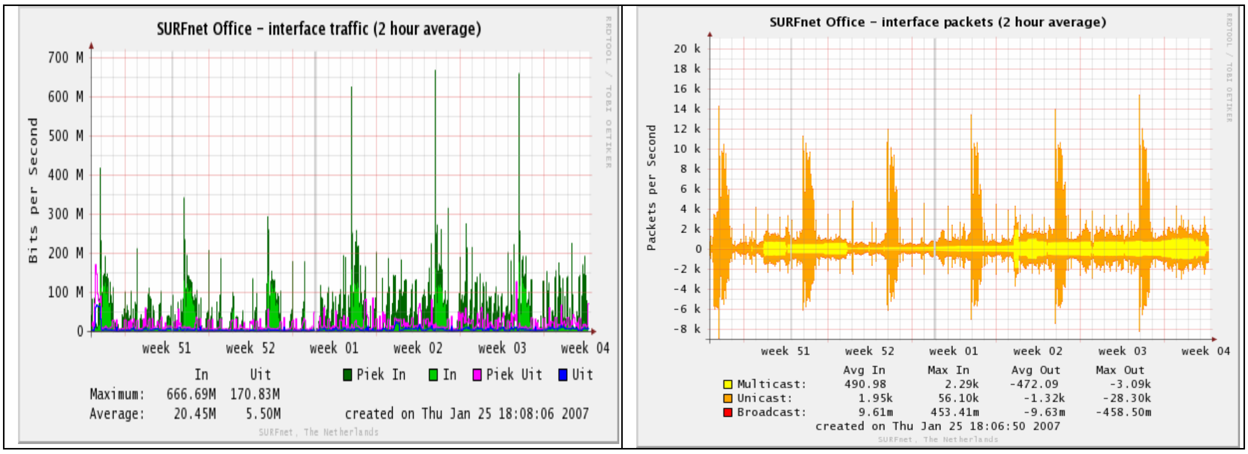
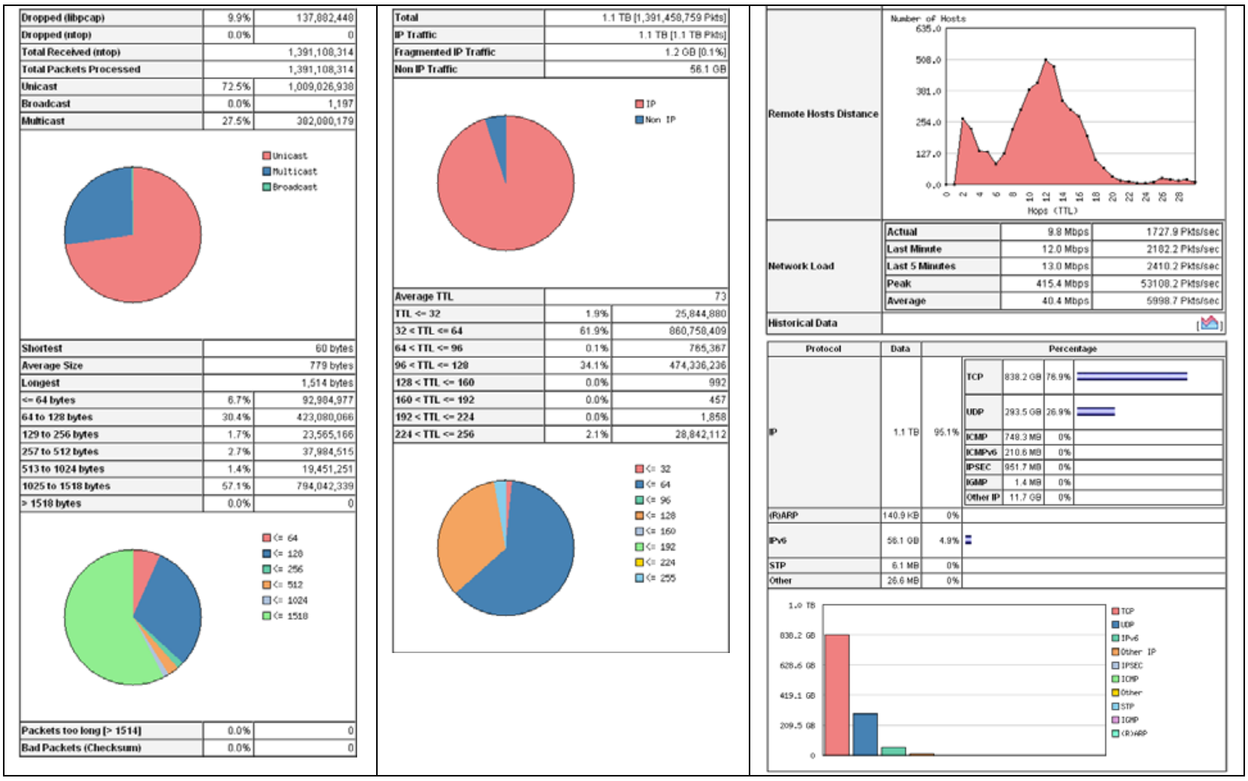


Figure 17: Interface statistics (SNMP measurement) for SURFnet office

Depending on the time of day and day of week, the traffic rate varies between 10 and 150 Mb/s, with peaks above 600 Mb/s and an average around 20 Mb/s. The IPv6 traffic accounts for around 5 % of the total volume and multicast for around 5 %. Over 90 % of packets are either 64-128 bytes long or around 1500 bytes. See Figure 18 for a visualisation from ntop.



Dropped (libpcap)	9.9%	137,882,448
Dropped (ntop)	0.0%	0
Total Received (ntop)		1,391,108,314
Total Packets Processed		1,391,108,314
Unicast	72.5%	1,009,026,938
Broadcast	0.0%	1,197
Multicast	27.5%	382,080,179

Shortest		60 bytes
Average Size		779 bytes
Longest		1,514 bytes
<= 64 bytes	6.7%	92,984,977
64 to 128 bytes	30.4%	423,080,066
129 to 256 bytes	1.7%	23,565,166
257 to 512 bytes	2.7%	37,984,515
513 to 1024 bytes	1.4%	19,451,251
1025 to 1518 bytes	57.1%	794,042,339
> 1518 bytes	0.0%	0

Packets too long [> 1514]	0.0%	0
Bad Packets (Checksum)	0.0%	0

Total	1.1 TB [1,391,458,759 Pkts]
IP Traffic	1.1 TB [1.1 TB Pkts]
Fragmented IP Traffic	1.2 GB [0.1%]
Non IP Traffic	56.1 GB

Average TTL		73
TTL <= 32	1.9%	25,844,880
32 < TTL <= 64	61.9%	860,758,409
64 < TTL <= 96	0.1%	765,367
96 < TTL <= 128	34.1%	474,336,236
128 < TTL <= 160	0.0%	992
160 < TTL <= 192	0.0%	457
192 < TTL <= 224	0.0%	1,858
224 < TTL <= 256	2.1%	28,842,112

Remote Hosts Distance

Network Load	Actual	9.8 Mbps	1727.9 Pkts/sec
	Last Minute	12.0 Mbps	2182.2 Pkts/sec
	Last 5 Minutes	13.0 Mbps	2410.2 Pkts/sec
	Peak	415.4 Mbps	53108.2 Pkts/sec
	Average	40.4 Mbps	5998.7 Pkts/sec
Historical Data			

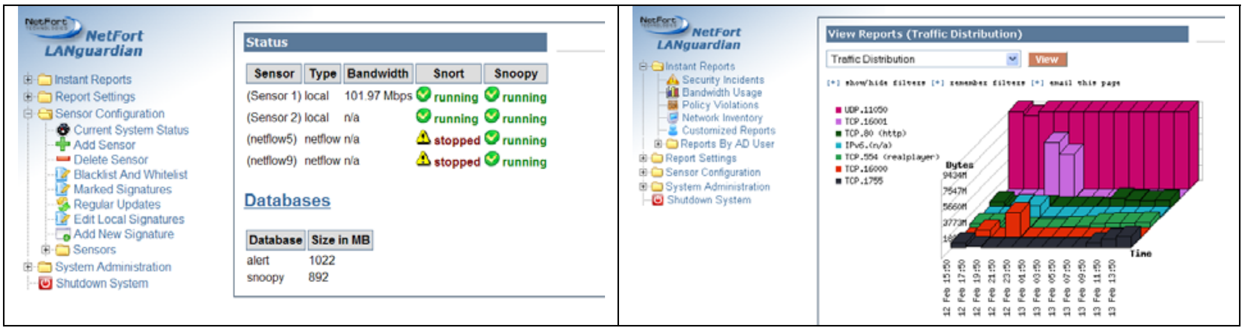
Protocol	Data	Percentage			
IP	1.1 TB	95.1%	TCP	838.2 GB	76.9%
			UDP	293.5 GB	26.9%
			ICMP	748.3 MB	0%
			ICMPv6	210.6 MB	0%
			IPSEC	951.7 MB	0%
			IGMP	1.4 MB	0%
			Other IP	11.7 GB	0%
(R)ARP	140.9 KB	0%			
IPv6	56.1 GB	4.9%			
STP	6.1 MB	0%			
Other	26.6 MB	0%			

Figure 18: ntop traffic report

4.4.3 Tests Performed

Installation, configuration and startup of the FlowMon probe and the software in our environment was easy and straightforward, thanks to the step-by-step documentation and scripts supplied.

The main applications for NetFlow and raw data inspection are related to security. For example, raw data as well as NetFlow data is fed to LANguardian with the aim of protecting the network against malicious traffic with known signatures. LANguardian currently supports only NetFlow version 5. Adding a sensor for NetFlow version 9 generated by the FlowMon probe was unsuccessful.



Status

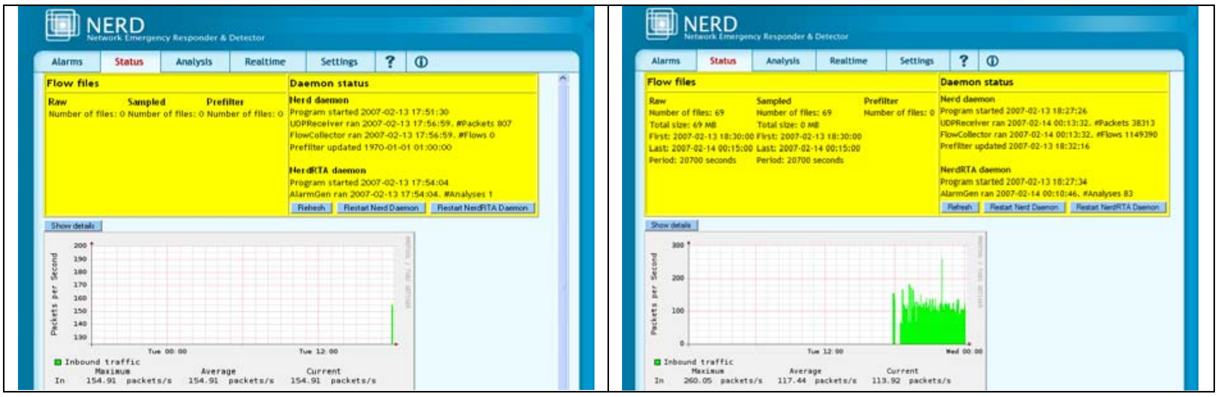
Sensor	Type	Bandwidth	Snort	Snoopy
(Sensor 1)	local	101.97 Mbps	✓ running	✓ running
(Sensor 2)	local	n/a	✓ running	✓ running
(netflow5)	netflow	n/a	⚠ stopped	✓ running
(netflow9)	netflow	n/a	⚠ stopped	✓ running

Databases

Database	Size in MB
alert	1022
snoopy	892

Figure 19: Sensor status and traffic distribution from LANguardian

Apart from this commercial solution, SURFnet also tried to develop its own NetFlow monitoring system initially based upon cflowd[1]. called NERD. Since it didn't support NetFlow version 9, an extra collector was written in C++. As Figure 20 shows, this achieved little success. Zero flows are displayed when the FlowMon probe exports NetFlow version 9.



Flow files

Raw
Number of files: 0

Sampled
Number of files: 0

Prefilter
Number of files: 0

Daemon status

Nerd daemon
Program started 2007-02-13 17:51:30
UDPReceiver ran 2007-02-13 17:56:59. #Packets 807
FlowCollector ran 2007-02-13 17:56:59. #Flows 0
Prefilter updated 1970-01-01 01:00:00

NerdRTA daemon
Program started 2007-02-13 17:54:04
AlarmGen ran 2007-02-13 17:54:04. #Analyses 1

```
■ Inbound traffic
         Maximum              Average              Current
 In    154.91  packets/s   154.91  packets/s   154.91  packets/s
```

Flow files

Raw
Number of files: 69
Total size: 69 MB
First: 2007-02-13 18:30:00
Last: 2007-02-14 00:15:00
Period: 20700 seconds

Sampled
Number of files: 69
Total size: 0 MB
First: 2007-02-13 18:30:00
Last: 2007-02-14 00:15:00
Period: 20700 seconds

Prefilter
Number of files: 0

Daemon status

Nerd daemon
Program started 2007-02-13 18:27:26
UDPReceiver ran 2007-02-14 00:13:32. #Packets 38313
FlowCollector ran 2007-02-14 00:13:32. #Flows 1149390
Prefilter updated 2007-02-13 18:32:16

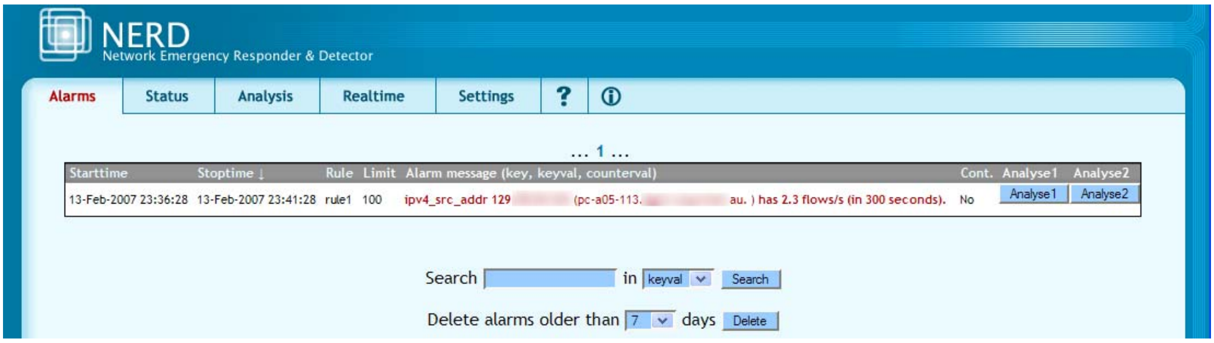
NerdRTA daemon
Program started 2007-02-13 18:27:34
AlarmGen ran 2007-02-14 00:10:46. #Analyses 83

```
■ Inbound traffic
         Maximum              Average              Current
 In    260.05  packets/s   117.44  packets/s   113.92  packets/s
```

Figure 20: NetFlow v9 (left) and v5 (right) analysis by NERD

For a brief period, the probe was configured to export NetFlow v5. Figure 21 shows that this could be used successfully. However, due to many issues, further development of NERD was terminated.

[1] http://www.caida.org/tools/measurement/cflowd/



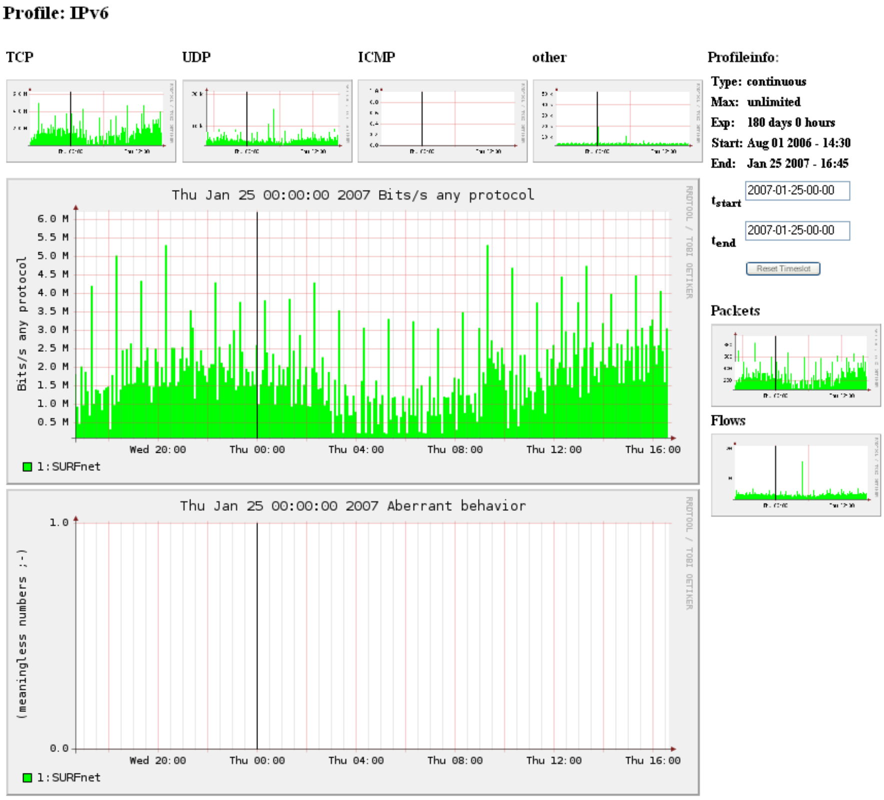
Starttime	Stoptime ↓	Rule	Limit	Alarm message (key, keyval, counterval)	Cont.	Analyse1	Analyse2
13-Feb-2007 23:36:28	13-Feb-2007 23:41:28	rule1	100	ipv4_src_addr 129[redacted] (pc-a05-113.[redacted] au.) has 2.3 flows/s (in 300 seconds).	No	Analyse1	Analyse2

Search [ ] in keyval Search

Delete alarms older than 7 days Delete

Figure 21: Alarm by NERD for an SSH scan based on NetFlow

The main tool used for visualisation and analysis of NetFlow data within SURFnet is now Nfdump[1]/NfSen[2]. As an example, Figure 22 shows some details for IPv6 traffic obtained from NetFlow v9 data generated by the FlowMon probe, including aberrant behaviour detection based on the built-in Holt-Winters algorithm of RRDtool[3].



Profileinfo:
Type: continuous
Max: unlimited
Exp: 180 days 0 hours
Start: Aug 01 2006 - 14:30
End: Jan 25 2007 - 16:45
t_{start} 2007-01-25-00-00
t_{end} 2007-01-25-00-00
Reset Timeslot

Figure 22: Example of NfSen details (including Holt-Winters patch)

[1] http://nfdump.sourceforge.net/
[2] http://nfsen.sourceforge.net/
[3] http://bakacsin.ki.iif.hu/~kissg/project/nfsen-hw/

As can be seen from Figure 23, NfSen initially reported unbelievable results for packet and bit rates from NetFlow data generated by FlowMon – too many packets and unlikely traffic distribution. After our bug report, this problem was fixed.

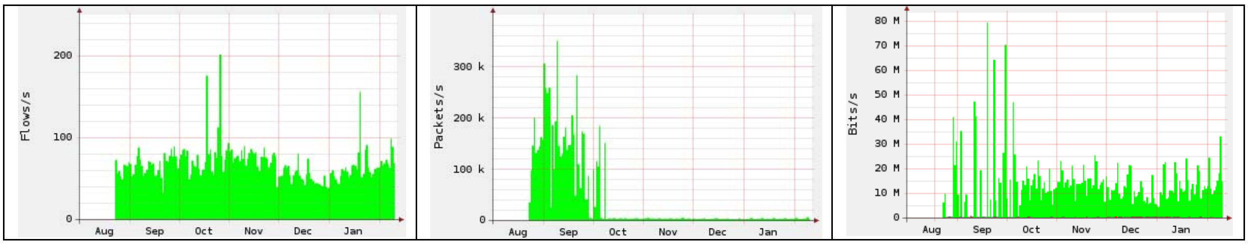


Figure 23: Corrupted NetFlow data, flows/s, packets/s, bits/s

To compare the flow data from the Catalyst 6509 (NetFlow v5) with those from the FlowMon probe (NetFlow v9), a five-minute time frame of traffic to port 53 was used. The anonymised results are shown in the following two tables.

First seen	Duration	Proto	Src IP Addr	Flows	Packets	Bytes
15:59:57.445	598.475	any	10.168.108.22	3198	4332	336344
16:00:00.762	563.945	any	10.168.110.29	737	968	70279
15:59:59.810	298.128	any	10.168.109.158	91	140	8925
15:59:58.594	480.030	any	10.168.108.186	83	86	6104
15:59:56.678	292.171	any	10.168.110.54	66	66	4318
16:01:11.044	513.614	any	10.168.108.12	63	162	13109
16:00:01.393	182.763	any	10.168.117.239	28	28	1856
16:03:04.664	0.781	any	10.168.110.11	25	25	2009
15:59:58.915	296.833	any	10.168.110.67	22	22	1758
16:00:47.893	266.109	any	10.168.110.73	20	113	7823

Table 2: Top 10 source IP addresses reported by Cisco 6509

First seen	Duration	Proto	Src IP Addr	Flows	Packets	Bytes
15:59:26.363	327.349	any	10.168.108.22	3756	4752	335484
15:59:48.440	250.143	any	10.168.109.158	103	177	10083
16:01:26.082	184.003	any	2001:61..7:109:4	103	198	15070
15:59:49.712	299.370	any	10.168.108.12	97	145	10962
15:59:50.059	290.578	any	10.168.108.186	89	89	5967
15:59:56.806	290.932	any	10.168.108.61	64	68	3060
15:59:57.516	289.198	any	10.168.110.54	63	63	3870
15:59:58.693	289.367	any	10.168.110.29	39	57	14418
16:00:00.022	177.058	any	2001:61..c2:c3c8	36	72	6268
16:03:05.831	1.213	any	10.168.110.11	25	25	1909

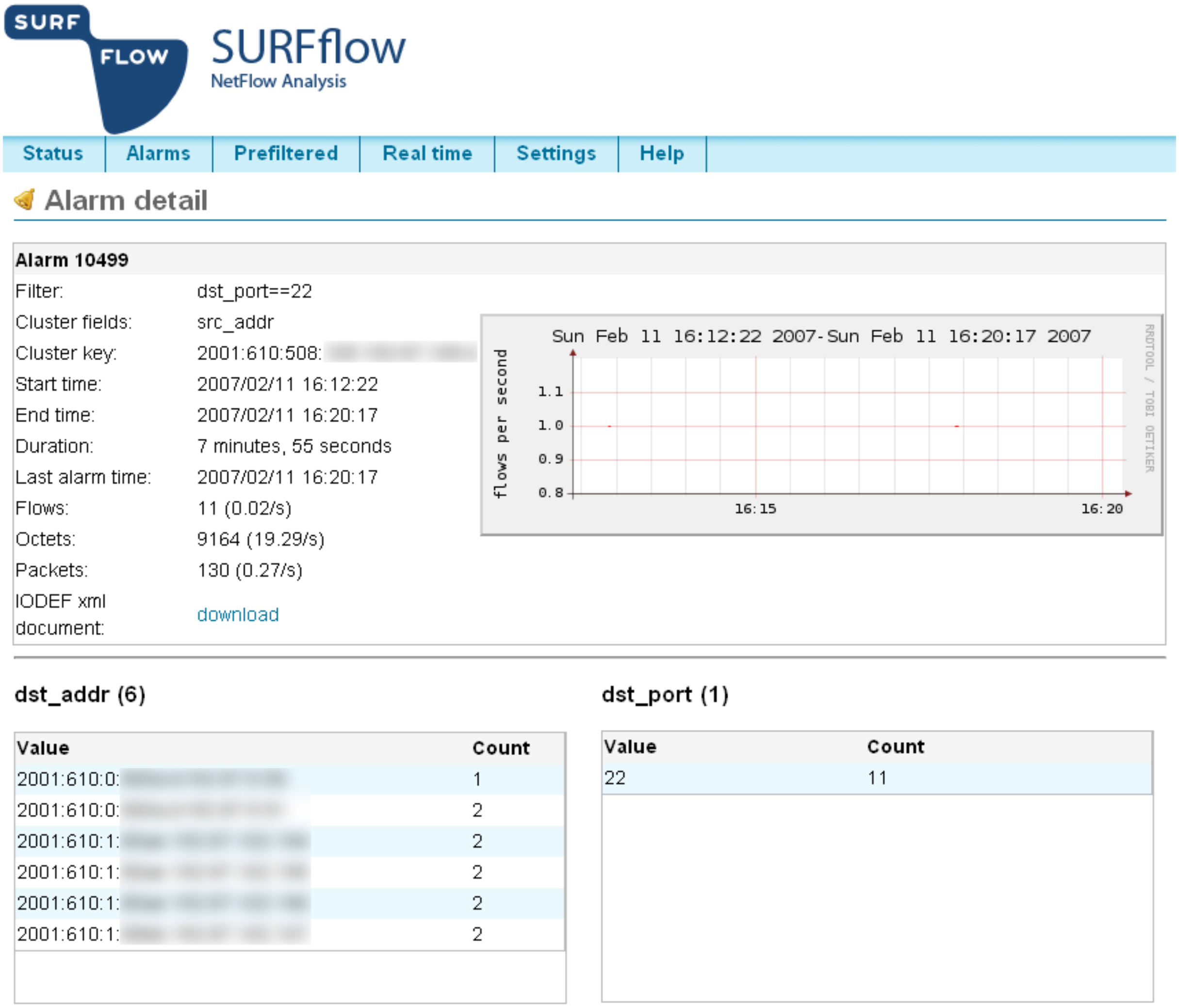
Table 3: Top 10 source IP addresses reported by FlowMon

The obvious difference between Table 2 and Table 3 is the appearance of IPv6 flows reported by NetFlow v9. Another significant difference is the amount of flows seen for source 10.168.110.29. Subsequent analysis proved that the Cisco switch was showing additional internal traffic that didn't make it to the mirror port. After adjusting for this, the numbers of flows were much closer. The number of bytes seems to differ by four per packet.

After terminating the development of our own collector, we used flowd[1] for building an anomaly detector named SURFflow[2] using the Python language and web-based frontend. Figure 24 shows again an example of what appears to be an SSH scan inferred from the NetFlow data generated by the FlowMon card.

The Peakflow SP[3] pilot software (version 3.4.2, build 6D2K) was also tested, together with FlowMon (version 1.0.2), but unfortunately without much success, see Figure 25. For unknown reasons, the number of flows detected by Peakflow SP when using NetFlow v9 from the FlowMon probe kept increasing. While the normal level is 100 flows per second, over 100 million flows per second (a) were soon observed. After that, the Peakflow system issued a “NO HEARTBEAT” message (b) and raised an alarm “collector down” (c). After a while, the messages would stop and the system would start again. With version 5 such behaviour wasn’t observed. Next to the v9 data collected earlier from the SURFnet Avici[4] routers, it is hard to see the resulting network summary (d).

[1] http://www.mindrot.org/projects/flowd/
[2] http://www.uitwisselplatform.nl/projects/surfflow/
[3] http://www.arbornetworks.com/products_sp.php
[4] http://www.avici.com/



Alarm detail

Alarm 10499	
Filter:	dst_port==22
Cluster fields:	src_addr
Cluster key:	2001:610:508:[redacted]
Start time:	2007/02/11 16:12:22
End time:	2007/02/11 16:20:17
Duration:	7 minutes, 55 seconds
Last alarm time:	2007/02/11 16:20:17
Flows:	11 (0.02/s)
Octets:	9164 (19.29/s)
Packets:	130 (0.27/s)
IODEF xml document:	download

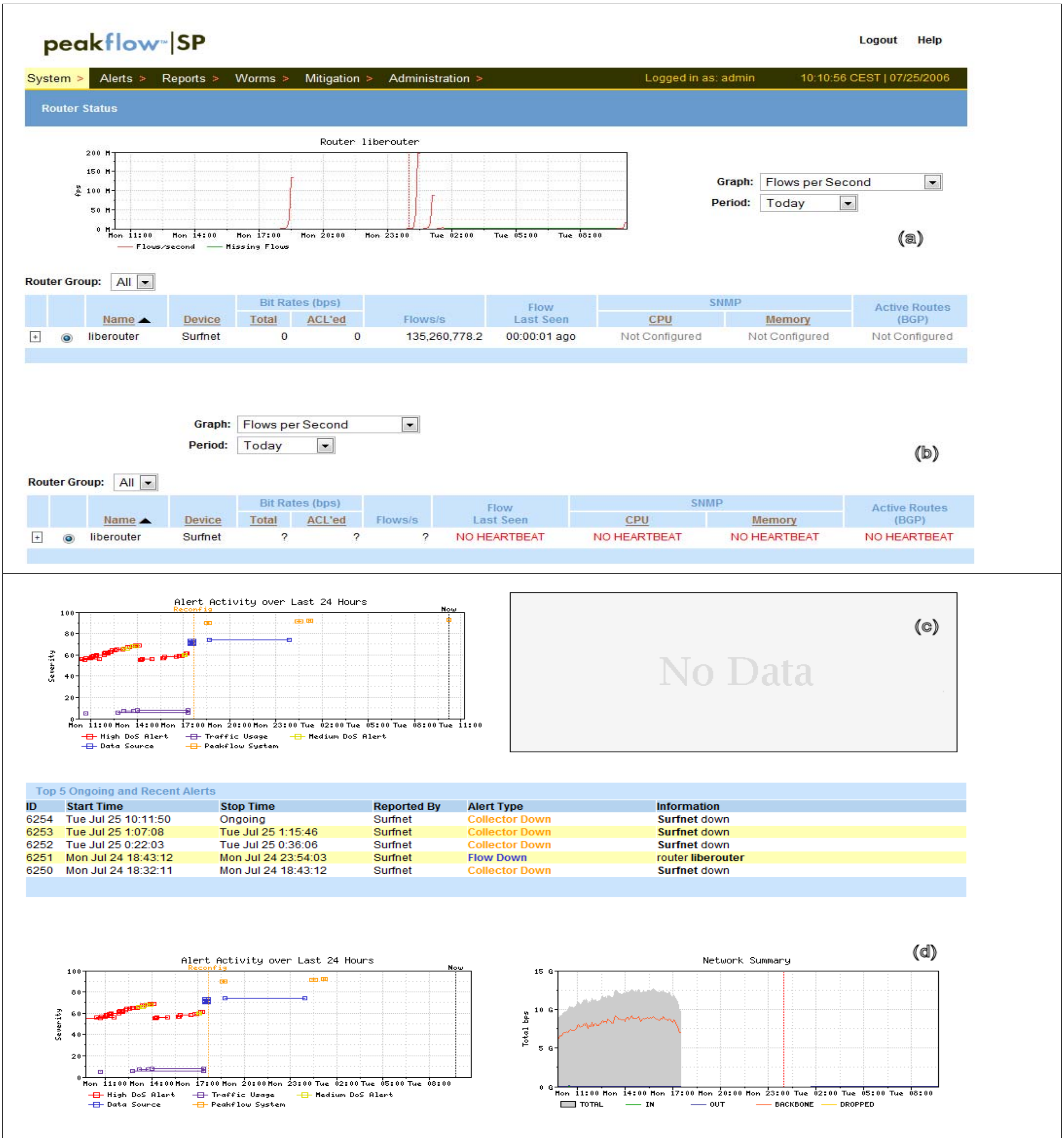
dst_addr (6)

Value	Count
2001:610:0:[redacted]	1
2001:610:0:[redacted]	2
2001:610:1:[redacted]	2
2001:610:1:[redacted]	2
2001:610:1:[redacted]	2
2001:610:1:[redacted]	2

dst_port (1)

Value	Count
22	11

Figure 24: SSH scan alarm issued by SURFflow based on NetFlow data



Graph: Flows per Second

Period: Today

Router Group: All

		Name ▲	Device	Bit Rates (bps) Total	Bit Rates (bps) ACL'ed	Flows/s	Flow Last Seen	SNMP CPU	SNMP Memory	Active Routes (BGP)
⊞	◉	liberouter	Surfnet	0	0	135,260,778.2	00:00:01 ago	Not Configured	Not Configured	Not Configured

Graph: Flows per Second

Period: Today

Router Group: All

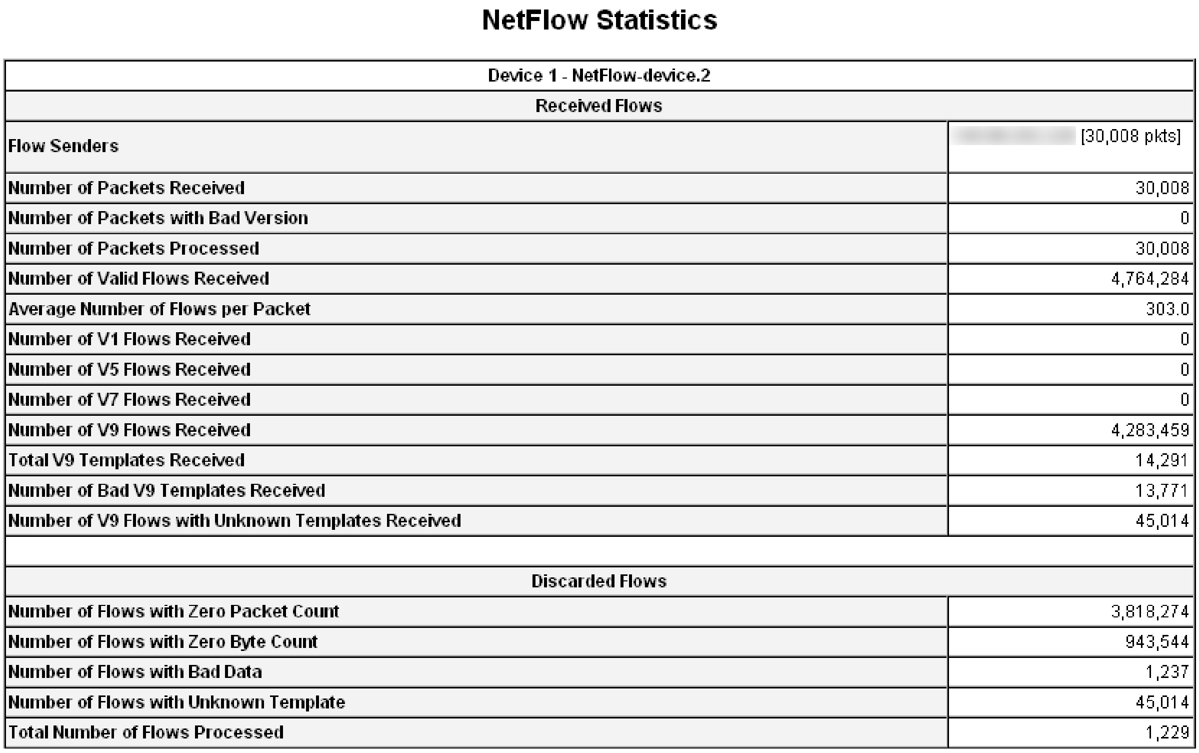
		Name ▲	Device	Bit Rates (bps) Total	Bit Rates (bps) ACL'ed	Flows/s	Flow Last Seen	SNMP CPU	SNMP Memory	Active Routes (BGP)
⊞	◉	liberouter	Surfnet	?	?	?	NO HEARTBEAT	NO HEARTBEAT	NO HEARTBEAT	NO HEARTBEAT

Top 5 Ongoing and Recent Alerts

ID	Start Time	Stop Time	Reported By	Alert Type	Information
6254	Tue Jul 25 10:11:50	Ongoing	Surfnet	Collector Down	**Surfnet** down
6253	Tue Jul 25 1:07:08	Tue Jul 25 1:15:46	Surfnet	Collector Down	**Surfnet** down
6252	Tue Jul 25 0:22:03	Tue Jul 25 0:36:06	Surfnet	Collector Down	**Surfnet** down
6251	Mon Jul 24 18:43:12	Mon Jul 24 23:54:03	Surfnet	Flow Down	router **liberouter**
6250	Mon Jul 24 18:32:11	Mon Jul 24 18:43:12	Surfnet	Collector Down	**Surfnet** down

Figure 25: Screenshots from Arbor Peakflow SP

NetFlow version 9 generated by the FlowMon probe was also used as input for ntop. Figure 26 shows an example of the NetFlow statistics as received by the NetFlow plug-in of ntop. Unfortunately, from this data it was not possible to obtain the same kind of information as shown in Figure 18.



NetFlow Statistics

Device 1 - NetFlow-device.2	
Received Flows	
Flow Senders	[redacted] [30,008 pkts]
Number of Packets Received	30,008
Number of Packets with Bad Version	0
Number of Packets Processed	30,008
Number of Valid Flows Received	4,764,284
Average Number of Flows per Packet	303.0
Number of V1 Flows Received	0
Number of V5 Flows Received	0
Number of V7 Flows Received	0
Number of V9 Flows Received	4,283,459
Total V9 Templates Received	14,291
Number of Bad V9 Templates Received	13,771
Number of V9 Flows with Unknown Templates Received	45,014
Discarded Flows	
Number of Flows with Zero Packet Count	3,818,274
Number of Flows with Zero Byte Count	943,544
Number of Flows with Bad Data	1,237
Number of Flows with Unknown Template	45,014
Total Number of Flows Processed	1,229

Figure 26: Sample of NetFlow statistics from ntop

4.4.4 Summary

While SURFnet routinely collects NetFlow data from its routers, the FlowMon probe proved to be a useful device for our environment. In particular, in the early phase of testing, it was the only tool capable of reporting IPv6 flows.

From the visualisation and analysis programs that we tested, Nfdump/NfSen seems to be the most appropriate system for flow analysis.

4.5 SWITCH

4.5.1 FlowMon Setup

The FlowMon hardware probe, provided by CESNET, was installed at SWITCH-CERT on 15 May 2006. It was plugged into a dedicated PC with Intel Pentium 4 CPU 1.60GHz and 256MB RAM. The installation of the probe itself was straightforward.

The input of the FlowMon probe was connected to a monitoring port of the SWITCH server zone, which delivers traffic between 900 Mb/s and 1 Gb/s traffic sustained. The generated v9 netflow traffic was sent to a dedicated NfSen test installation. The repeater output of the probe was not used.

4.5.2 Test results

After starting the probe using the supplied script, the following information was displayed:

```
nfprobe% csid -s
Board    : combo6
Addon    : mtx2
Chip     : xcv1000
Firmware : ok
SW       : 0xf1010002
HW       : 0x2
Text     : NETFLOW_1Gbps_Probe

Device [combo6] netflow-ph1
  (0xf1010002-0xf10100ff) {NETFLOW_1Gbps_Probe}: active
Device [combo6] netflow-ph1
  (0xf1010001-0xf1010001) {NETFLOW 1Gbps Probe}: inactive

nfprobe% cat /proc/driver/combo6/card0/netflowph1-regs
SW_ID:          f1010002
HW_ID:          00000002
IRQ mask:       00000001
IRQ pending:    00000000
FPGA intercomm: 00000001
Input items:    00384604
Received items: 00384604
Memory config:  02000008
Write pointer:  00000004
Read pointer:   00000001
IRQ items:      00000020
IRQ timeout:    0000000a
```

Exported flow records were collected and processed. A few minor bugs in the NetFlow v9 protocol implementation were reported back to CESNET. These issues included:

- Report octets as 4bytes in the out_bytes instead of 5bytes in in_bytes field due to compatibility with CISCO.
- Wrong TCP flags decoding.
- Impossible packet lengths (> 1500 bytes per packet) on Ethernet.

All issues were fixed by CESNET and the probe now works as expected.

However, it turned out that this prototype version of the probe was not able to cope with sustained traffic of about 1 Gb/s.

About 30 minutes of run time resulted in:

```
nfprobe% csbus -c 10 2000
0c9398c1 00000000 00005555 00000007 0000003e 00000006 00000000 09bf7a3d
00000001 00000000
```

This reads as

```
0c9398c1 successfully received packets
00000000 packets received with error
09bf7a3d lost packets
```

When we compared the traffic as reported by the router (Figure 27) with data reported by NfSen from flow records sent by the FlowMon probe, it was clear that the 1 Gb/s sustained traffic resulted in about 470 Mb/s according to the probe. Furthermore, after the probe runs for some time with a high traffic rate, it seems to increasingly drop traffic. Figure 28 shows a point in time when the exporter software was restarted after it had been running for about one month. The traffic rate immediately increased by 80 Mb/s.

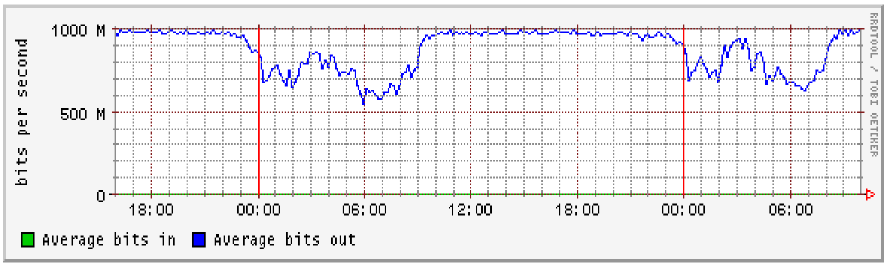


Figure 27: Traffic rates as reported by the router

The probe has now been running for about eight months, and seems to be stable within the limits described above. Three complete freezes of the software occurred during that time, when no traffic was exported and the software had to be restarted.

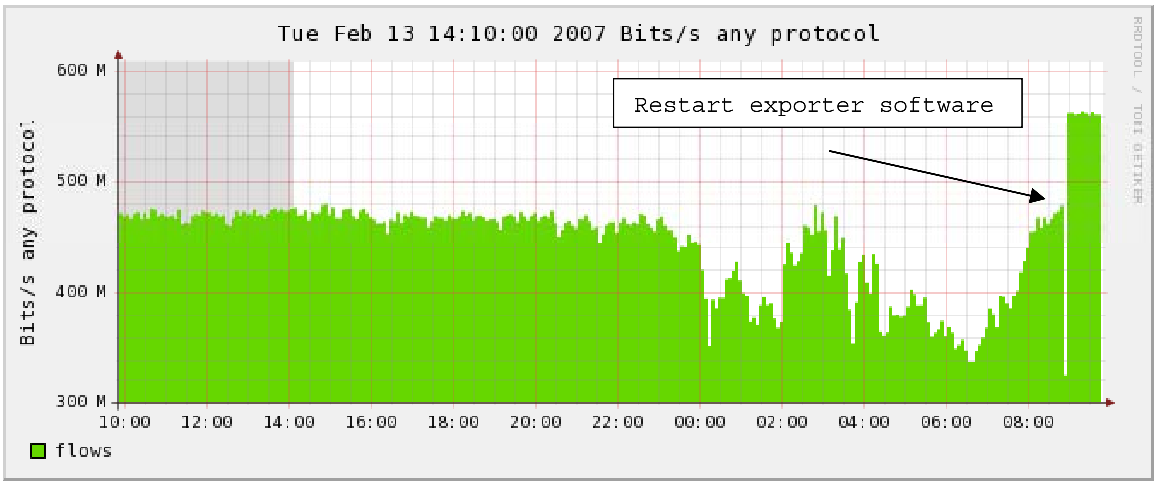


Figure 28: Traffic rates as reported by NfSen

4.5.3 Summary

If the next/final version of hardware overcomes the performance limits demonstrated above and is able to cope with 1 Gb/s sustained unsampled traffic, the FlowMon probe should be of a very high value to any NREN collecting NetFlow data for operation or security monitoring purposes.

5 Conclusions

This deliverable describes the hardware-accelerated FlowMon probe, and tests performed using it by CESNET and four JRA2 partners who expressed interest in early tests of the probe.

The testing environments included NREN backbones with high traffic volumes as well as smaller sites. In all cases the test results were largely positive, and all testers found the probe valuable for their own monitoring purposes and needs. The main benefits over existing methods of collecting NetFlow data from routers are:

- The burden of NetFlow data export can be offloaded from the routers, thus decreasing their CPU load.
- The probe (final version) is able to generate *unsampled* NetFlow data from 1 Gb/s links, which is not the case even with many relatively high-end routers.
- The probe fully supports IPv6 in hardware. Many routers don't, or do it in software.
- The probe is able to provide certain information elements with finer granularity than routers (for example TCP flags).
- FlowMon firmware and (more importantly) software is open source, which makes the device much more flexible and allows for an easier assessment of its reliability and security.

Bug reports and other feedback from the testers helped the developers considerably in preparing new firmware and software releases. In general, this interaction was quite smooth and proved that such a cooperation of partners in projects like GN2 can lead to valuable research and development results that can be turned into competitive products.

One important piece that is still missing is the support for 10GE interfaces. Recurring problems with availability and parameters of high-end electronic components (for example phyters) are now hopefully resolved, and first tests of the FlowMon probe with 10GE interfaces are planned for April 2007.

Still in year 3 of the GN2 project, eight GE probes will be distributed to GN2 partners as a part of the JRA2 Security Toolset. The probes will be purchased, including support and maintenance, from a start-up company that CESNET helped to establish. The tests described on the previous pages suggest that the Toolset (combining the FlowMon probe with NfSen) will enable the NRENs and their CSIRT teams to perform effective, flow-based security monitoring on a continuous basis.

6 References

[Cel06] Čeleda, P. et al. *FlowMon Probe.* Technical report 31/2006, CESNET, Praha, 2006. URL: http://www.cesnet.cz/doc/techzpravy/2006/flowmon-probe/

[EV02] Estan, C. and Varghese, G. New directions in traffic measurement and accounting. In: *Proceedings of the 2001 ACM SIGCOMM Internet Measurement Workshop*, San Francisco, California, 2001, pp. 75-80.

[Kos04] Košňar, T. Flow-Based Traffic Analysis System – Architecture Overview. Technical Report 15/2004, CESNET, Praha, 2004. URL: http://www.cesnet.cz/doc/techzpravy/2004/ftas-arch/

[RFC3954] Claise, B. (Ed.). *Cisco Systems NetFlow Services Export Version 9*. RFC 3954, IETF, October 2004. URL: http://www.ietf.org/rfc/rfc3954.txt

[ZPK05] Žádník, M., Pečenka, T. and Kořenek, J. FlowMon probe intended for high-speed networks. In: Rissa, T., Wilton, S. and Leong, P. (Eds.), *Proceedings of the 15th International Conference on Field-Programmable Logic and Applications (FPL05), Tampere, Finland.* IEEE CS, 2005, pp. 695-698.

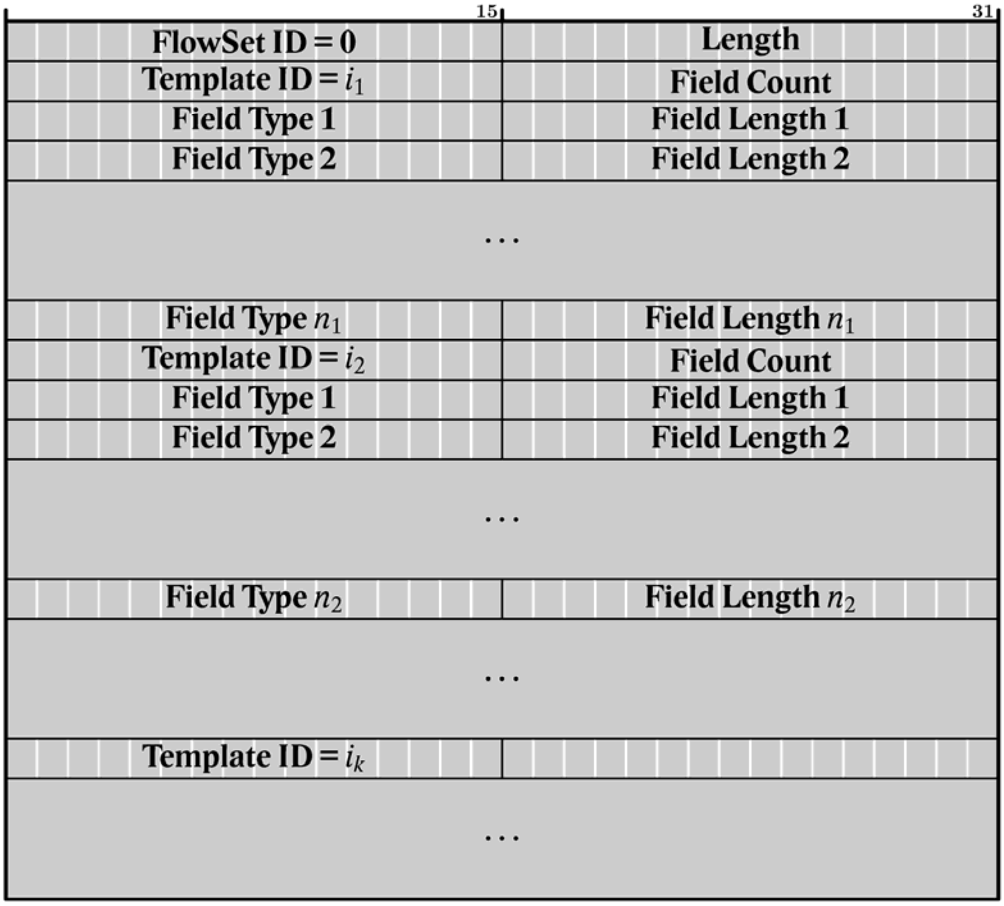
7 Acronyms

CRC	Cyclic Redundancy Check
FIFO	First-In-First-Out
FPGA	Field-Programmable Gate Array
FTAS	Flow-based Traffic Analysis System
GCC	GNU C Compiler
GE	Gigabit Ethernet
GNU	GNU is Not Unix
HFE	Header Field Extractor
HSRCH	Hash Search Unit
IBUF	Input Buffer
ICMP	Internet Control Message Protocol
JRA	Joint Research Activity
MAN	Manager Unit
PCI	Peripheral Component Interconnect
RAM	Random Access Memory
SCTRL	Storage Control Unit
TCP	Transmission Control Protocol
TOS	Type of Service
UDP	User Datagram Protocol
UH	Unified Header
WI	Work Item
10GE	Ten-Gigabit Ethernet

Appendix A NetFlow Version 9 Templates

This appendix shows the structure of all Netflow v9 templates that are used by the FlowMon NetFlow v9 exporter program. All eight templates in use are periodically sent inside a single Template FlowSet [RFC3954]. The period can be configured via a command-line parameter to the exporter program.

The general structure of the Template FlowSet used by the NetFlow v9 exporter program is outlined in the table below:



15	31
FlowSet ID = 0	Length
Template ID = i_1	Field Count
Field Type 1	Field Length 1
Field Type 2	Field Length 2
...	
Field Type n_1	Field Length n_1
Template ID = i_2	Field Count
Field Type 1	Field Length 1
Field Type 2	Field Length 2
...	
Field Type n_2	Field Length n_2
...	
Template ID = i_k	
...	

The following sections contain tables describing the layout of all templates (the blocks that start with “Template ID” in the figure above). Details about individual fields can be found in [RFC3954].

IPv4/TCP

- Template ID = 256
- Field count = 12

Field Type (Decimal Code)	Field Length
FIRST_SWITCHED (22)	4
LAST_SWITCHED (21)	4
IN_BYTES (1)	8
IN_PKTS (2)	4
SAMPLING_INTERVAL (34)	4
SAMPLING_ALGORITHM (35)	1
PROTOCOL (4)	1
IPV4_SRC_ADDR (8)	4
IPV4_DST_ADDR (12)	4
L4_SRC_PORT (7)	2
L4_DST_PORT(11)	2
TCP_FLAGS (6)	1

IPv4/UDP

- Template ID = 257
- Field count = 11

Field Type (Decimal Code)	Field Length
FIRST_SWITCHED (22)	4
LAST_SWITCHED (21)	4
IN_BYTES (1)	8
IN_PKTS (2)	4
SAMPLING_INTERVAL (34)	4
SAMPLING_ALGORITHM (35)	1
PROTOCOL (4)	1
IPV4_SRC_ADDR (8)	4
IPV4_DST_ADDR (12)	4
L4_SRC_PORT (7)	2
L4_DST_PORT(11)	2

v4/ICMP

- Template ID = 258
- Field count = 10

Field Type (Decimal Code)	Field Length
FIRST_SWITCHED (22)	4
LAST_SWITCHED (21)	4
IN_BYTES (1)	8
IN_PKTS (2)	4
SAMPLING_INTERVAL (34)	4
SAMPLING_ALGORITHM (35)	1
PROTOCOL (4)	1
IPV4_SRC_ADDR (8)	4
IPV4_DST_ADDR (12)	4
ICMP_TYPE (32)	2

IPv4/OTHER

- Template ID = 259
- Field count = 7

Field Type (Decimal Code)	Field Length
OUT_PKTS (24)	4
FIRST_SWITCHED (22)	4
OUT_BYTES (23)	5
LAST_SWITCHED (21)	4
PROT (4)	1
IP_SRC_ADDR (8)	4
IP_DST_ADDR (12)	4

IPv6/TCP

- Template ID = 260
- Field count = 13

Field Type (Decimal Code)	Field Length
FIRST_SWITCHED (22)	4
LAST_SWITCHED (21)	4
IN_BYTES (1)	8
IN_PKTS (2)	4
SAMPLING_INTERVAL (34)	4
SAMPLING_ALGORITHM (35)	1
IP_PROTOCOL_VERSION (60)	1
PROTOCOL (4)	1
IPV6_SRC_ADDR (27)	16
IPV6_DST_ADDR (28)	16
L4_SRC_PORT (7)	2
L4_DST_PORT(11)	2
TCP_FLAGS (6)	1

IPv6/UDP

- Template ID = 261
- Field count = 12

Field Type (Decimal Code)	Field Length
FIRST_SWITCHED (22)	4
LAST_SWITCHED (21)	4
IN_BYTES (1)	8
IN_PKTS (2)	4
SAMPLING_INTERVAL (34)	4
SAMPLING_ALGORITHM (35)	1
IP_PROTOCOL_VERSION (60)	1
PROTOCOL (4)	1
IPV6_SRC_ADDR (27)	16
IPV6_DST_ADDR (28)	16
L4_SRC_PORT (7)	2
L4_DST_PORT(11)	2

IPv6/ICMP

- Template ID = 262
- Field count = 11

Field Type (Decimal Code)	Field Length
FIRST_SWITCHED (22)	4
LAST_SWITCHED (21)	4
IN_BYTES (1)	8
IN_PKTS (2)	4
SAMPLING_INTERVAL (34)	4
SAMPLING_ALGORITHM (35)	1
IP_PROTOCOL_VERSION (60)	1
PROTOCOL (4)	1
IPV6_SRC_ADDR (27)	16
IPV6_DST_ADDR (28)	16
ICMP_TYPE (32)	2

IPv6/OTHER

- Template ID = 263
- Field count = 7

Field Type (Decimal Code)	Field Length
OUT_PKTS (24)	4
FIRST_SWITCHED (22)	4
OUT_BYTES (23)	5
LAST_SWITCHED (21)	4
PROT (4)	1
IPV6_DST_ADDR (28)	16
IPV6_SRC_ADDR (27)	16

Appendix B Terminology

The following terms are used throughout the deliverable:

active timeout Time period after which data about an ongoing flow are exported. This way, information about long-term flows is not lost by exporter failure.

collector Computer running a program that receives flow records from one or more exporters. It may either simply store the records in a database or provide additional functions, such as further processing or viewing the data.

exporter Device sending packets that contain records about observed flows to one or more collectors. It could be an IP router or a specialised probe.

inactive timeout If no packets for a given flow are observed for this time period, the flow is terminated and the corresponding flow record exported. Note that termination of flows can sometimes be based on protocol-related events such as presence of FIN or RST flags in TCP segments.

IP traffic flow Subset of IP packets passing an observation point during a certain time period that share a specified set of common properties (key fields) taken from IP and transport headers, and/or additional context such as input interface, source autonomous system.

sampling Process in which a subset of the incoming packets is selected for further processing. Only these selected packets are taken into account in the flow records. The simplest case is *statistical sampling*: Each incoming packet is selected with probability p_s. It is often described in terms of sampling rate R, which is the reciprocal value of the sampling probability, i.e., $R = 1/p_{s.}$