

# FlowMon Probe

**Pavel Čeleda, Milan Kováčik, Tomáš Koníř, Vojtěch Krmíček, Petr Špringl, Martin Žádník**

21.12.2006

## 1 Abstract

The objective of this technical report is to describe hardware and software activities during the development of FlowMon probe. FlowMon probe is second phase of IP flows monitoring project originally inspired by NetFlow. Its first phase was a proof of concept which showed feasibility to implement IP packet monitoring in board with FPGA and higher functionality such as exporting flow-records in host PC. Today the FlowMon probe is ready to use but still some improvements are in progress. In near future we would like to support various mix of features. For example adaptive sampling, large flow-cache, variable flow-record, web front-end.

**Keywords:** FlowMon, NetFlow, FPGA, Liberouter

## 2 Introduction

The project started two years ago and its goal was to create an autonomous network probe which gathers important data about network traffic. Our inspiration was the NetFlow implemented by Cisco. It provides aggregated information about IP flows which could be used to manage current networks, plan new topologies or detect DoS attacks, also for billing and accounting. Standalone dedicated device for such type of monitoring has several benefits. For example it is not necessary to change routers that do not support NetFlow. Also the probe has high performance and flexibility in means of various enhancements to protect itself against malicious traffic. These countermeasures (proposed in [EV03]) are missing even in the state-of-the-art routers.

Today the phase one [CEL05], [ZAL05] is stable and distribution packages are available for download at the Liberouter<sup>1</sup> web site. The phase one is successfully used to monitor networks by our GEANT2 partners such as SURFnet, SWITCH, ISTF (monitoring whole Bulgarian NREN), etc. Phase one is running on older

---

<sup>1</sup><http://www.liberouter.org/clients/>

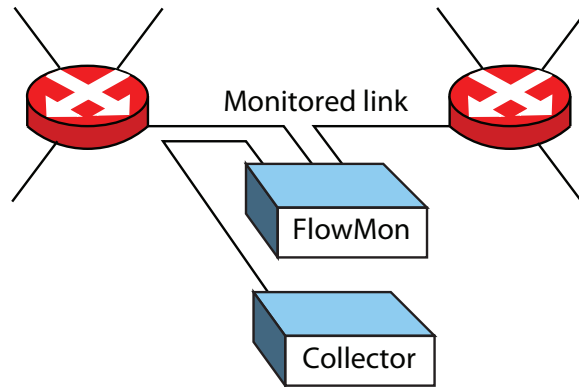
type of COMBO cards, COMBO6 mother card and COMBO-4MTX add-on card. These cards are now being replaced by much more powerful version of COMBO6X mother card and COMBO-4SFPRO add-on card which are used as a new platform for FlowMon probe. Therefore we consider the phase one to be finished but we will still continue with its support in terms of bug fixes or minor improvements. The rest of the paper is focused on the phase 2 - FlowMon probe.



**Figure 1:** COMBO6X - PCI-X 64/66MHz hardware accelerated card

The FlowMon probe works as a T-splitter and when inserted in a line it is fully transparent for all network traffic. IP flows are monitored by FPGA (*Field Programmable Array*) chip and stored in on-board external memories. Expired flow-records are forwarded to upper software layers. User space application sends them to collector.

The report is organised as follows. Section 3 describes an abstract view of the FlowMon probe. Next two sections then give the details about firmware and software. Following sections enumerate FlowMon probe capabilities and results of testing FlowMon probe with various NetFlow collectors. Finally, further improvements and plans are discussed.



**Figure 2:** Probe inserted in line and sending data to collector.

### 3 FlowMon probe structure

The whole system of the FlowMon probe consists of two parts - hardware and software. The hardware part is intended for processing of incoming packets on high speed (wire speed) while the software together with ordinary network card are utilized as a transmitter of flow-records. Time critical monitoring has to be implemented on the adapter card, on the other side complex implementation of NetFlow protocol is done in software. This distribution of tasks allows fast upgrades every time when new export format is deployed.

The system could be divided into several layers with specific tasks (see Figure 3).

PC	Flow Exporter
	Filtering and Anonymization
	Driver
FPGA	Export to SW
	Monitoring
	Packet Header Parsing
Phytors	Physical Layer

**Figure 3:** FlowMon probe layers

## Description of FlowMon probe layers

- Physical layer - process the L1 layer.
- Packet Header Parsing - extracts important data from packets on L2 and L3 layer.
- Metering - gather information about IP addresses, port numbers, protocols, number of bytes and packets, start and end timestamps and other header fields.
- Driver - provides interface to load firmware, set parameters of metering process and transport flow-records from the card to the PC.
- Anonymization - modifies IP addresses to keep users privacy.
- Filter - defines which flow record is send to given collector.
- Flow Exporter - encapsulates flow records in NetFlow v5 or NetFlow v9 packet and sends it to collector.

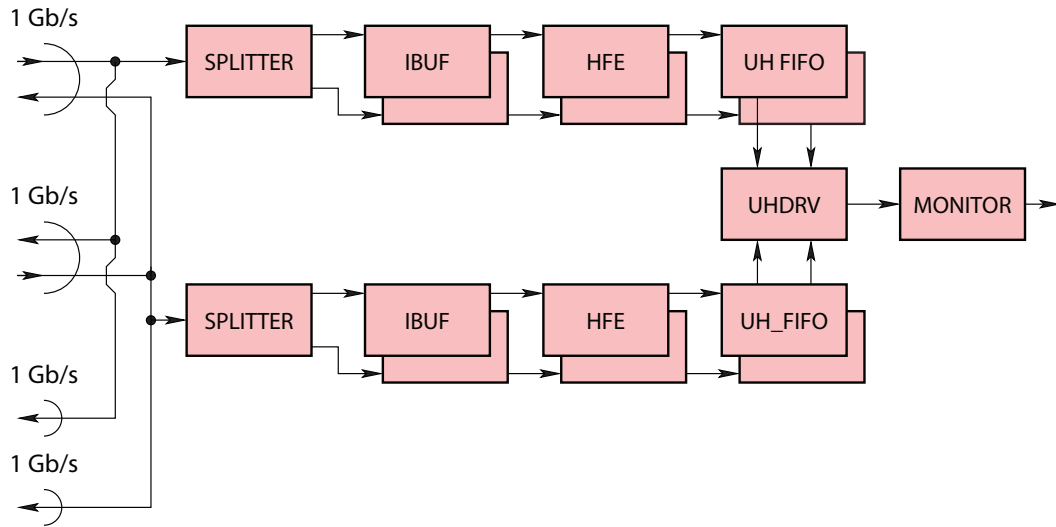
## 4 FlowMon probe firmware

The firmware is composed of several units which are chained in processing pipeline. Phase one showed us bottlenecks of the previous design. Therefore some parts of the packet processing pipeline is instantiated multiple times. It allows to increase the throughput to be sufficient to process all packets no matter of what size at full wire speed. At first the packet header parsing pipeline is described and then the metering part.

Incoming packets from network interface are entering the *Input Buffer (IBUF)* block. For each incoming packet the CRC (*Cyclic Redundancy Check*) checksum and packet length (up to 8192 bytes) is checked and only valid packets pass through. Valid packets are subject of optional sampling. Sampling rate can be set from 1:1 (take every packet) to 1:65535. Periodic, random and byte sampling are implemented. Timestamp (resolution 640 ns) is assigned to sampled packet and packet is sent for further analysis.

Input buffer was redesigned to be able to handle short packets at full gigabit speed. Older version was not able to handle burst of packets and therefore a lot of them were dropped randomly or high sampling rate had to be set.

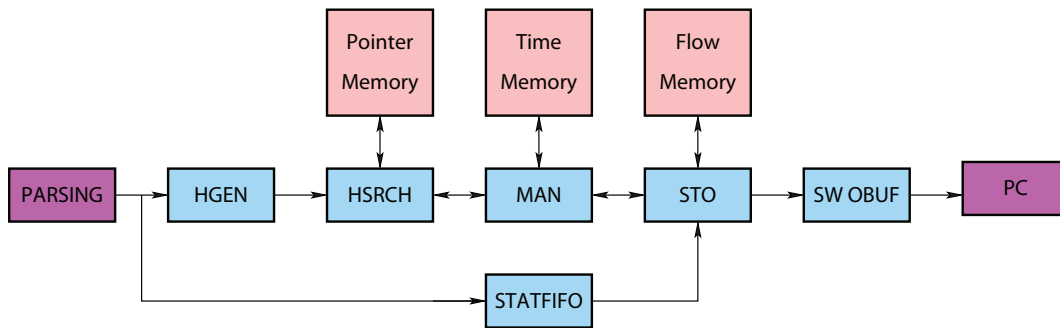
Packets with assigned timestamp are processed by *Header Field Extractor (HFE/GENA)*. The HFE2 is a processor based on RISC (*Reduced Instruction Set Computer*) architecture controlled by specific instruction set intended for stream



**Figure 4:** Packet header parsing pipeline

analysis. The processor core is called GENA (*GENeric NANoprocessor*) and with its peripherals and program for FlowMon it is able to handle one million packets per second. Two HFE2 are instantiated per one interface to process full gigabit with no packet loss. The extracted structure is called *Unified Header (UH)* and is stored in *Unified Header FIFO* temporarily.

*UH driver (UHDRV)* manage reading of unified headers out of the FIFOs. The policy of management is round-robin. UHDRV transfers UH to the metering part.



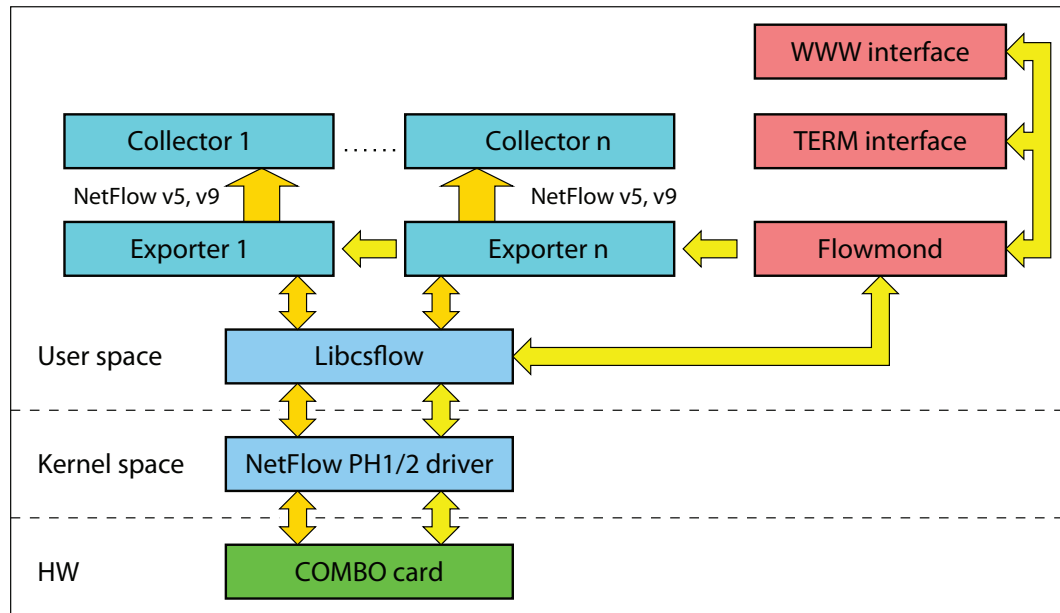
**Figure 5:** Metering pipeline

At the input of metering part data are duplicated into two paths (control and data path). Data path consists of *Statistic FIFO (STATFIFO)* which stores the UH until is processed by *Storage (STO)*. The control path begins with *Hash Generator (HGEN)* which computes 64-bit CRC from the fields which designates the flow. Part of the CRC is used as an address in *Hash Search (HSRCH)*. Hash Search role

is to look-up if the record is already in the memory a give a pointer to the flow memory. Using the pointer *Manager* checks a time-record of the flow. Finally the flow-record is created or updated in *Storage (STO)*. If the flow expires its record is deleted and transfered to *Software Output Buffer (SW OBUF)* where it waits till it is read by driver.

## 5 FlowMon probe software

The software for FlowMon probe consists of several parts - kernel driver, user space libraries, configuration programs and flow exporters.



**Figure 6:** FlowMon probe software layers

The FlowMon kernel driver for COMBO6 and COMBO6X cards supports latest 2.4 and 2.6 Linux kernels. The configuration of the driver sources is standalone, but the configured Linux kernel source tree must be present to proceed a successful compilation. The configuration script is available for the user to make the driver configuration easy.

The libcsflow library was designed as a middle layer between exporter applications and kernel driver. It hides specific system calls and provides the standard interface in C language. The future improvements will be to move more common code like the NetFlow record parsing to this library from exporter application.

The configuration programs are used for loading and initializing the FlowMon probe firmware into COMBO cards after kernel drivers are loaded. Main con-

figuration programs are `csxtool(1)` for firmware loading and `flowmonctl(1)` tool for initialization routines. The routines are initialization of hardware units UHDRV and HGEN, loading program to HFE processor and settings timeouts, sampling rate and starting up the whole probe.

The flow exporter is a basic data exporting tool in NetFlow v5 and v9 format. It is written in C to support the COMBO6 and COMBO6X cards with NetFlow extension. It communicates with a collector via the IPv4 or IPv6 protocol using UDP protocol.

A set of shell scripts can be used for configuring and starting up the FlowMon probe. These scripts call programs like `csxtool(1)` and `flowmonctl(1)` and check any unexpected situations. User only set several constants which say how to configure the probe. The script loads right drivers. Detects which COMBO cards are installed in computer and according that it will load right firmware into FPGAs. Then calls `flowmonctl(1)` and initialize firmware and starts it up. Active and inactive timeouts are set according parameters of the script. Flow exporter is activated and exporting data to specified collector.

## 6 Flow export program

Because of some problems with our first version of data exporting program we proposed and implemented new `flowmon_nf5,9(1)` program. FlowMon is a modular NetFlow v5 and v9 data exporting tool. It is written in pure C language to support COMBO6 and COMBO6X mother cards with NetFlow extension. It uses IPv4 or IPv6 for communication with independent NetFlow collector.

### Structure and functions

FlowMon is designed as modular system and contains four basic parts. The first supervising module maintains all other modules and controls the flow processing. The next two modules are designed for reading flows and made some additional work (filtering and anonymization). Last part is responsible for creating and sending NetFlow datagrams. The supervising module parses the command line options, calls init functions for all modules and handle errors. The basic algorithm is really simple. With all modules enabled, FlowMon works in infinite loop (until terminated):

```
init_main_part
init_hardware
init_addons      /* filtering, anonymization */
init_exporter   /* v5 or v9, depends on executable name */

loop_begin
```

```

        if <read_flow_from_hw>
            if <test_filter>
                do_anonymization
                add_to_export
            fi
        fi
    loop_end

```

The flow reading module is now designed to support only COMBO cards, but may be extended in future to support other flow sources. The module contains two parts. One for testing hardware availability and hardware initialization. Second for continuous reading of the flows from hardware and transmitting them to supervising module. The basic algorithm is described below.

```

begin init_hw
    test_hw_present
    init_hw
end

begin read_hw
    get_flow
    if_not_error
        fill_independent_flow_struct
        return_flow
    else
        return_error
    fi
end

```

The exporting module is splitted into two independent modules for NetFlow v5 and v9. The both modules do the similar work. Collects flow until packet size is reached and than send packet to collector. NetFlow v5 module only fills packet and sends it to collector. NetFlow v9 module is watching if timeout or packet counter is reached and than sends the v9 templates to collector. The templates are fully configurable using simple configuration file, described in user documentation.

The basic algorithm of NetFlow v9 exporting module:

```

begin init_v9
    create_collector_connection
    init_templates
    add_templates_to_send

```

```

end

begin process_v9
    find_best_template_to_use
    fill_template_data
    add_data_to_send
    if <packet_size_reached>
        send_packet
        if <template_refresh_interval_reached>
            add_templates_to_send
        fi
    fi
end

```

The last part are add-on modules. These modules are designed to do some middle work, like flow mangling, filtering etc. Following modules are available:

### **Filter module**

The FlowMon filter is a basic interval matching implementation related to network addresses. It works for both IPv4 and IPv6 (actually, each IP protocol family has own filter). Current filter storage is a list. This fact implies poor performance for large filters. But the filtering framework allows to implement storage more effectively - addresses are being converted into "integers" in host byte order (for IPv6 this means a 16 byte unsigned "long long long int") and a couple of macros dedicated to compare those integers are provided too. This makes it possible to replace storage back ends independently of the stored data.

### **Anonymization module**

Anonymization module is designed to hide original data source information, like IP addresses and port numbers. Currently supported are two modes basic and AES (*Advanced Encryption Standard*). Basic module only masks IP address to defined address range (default is 192.168.0.0/16) and works only for IPv4 flows. AES anonymization module applies AES block cipher to selected fields (IP address, port numbers) to do total anonymization. Cipher may be initialized using zero string or current system time to provide better security level. Fields which will be anonymized are configurable. Supported fields are source and destination addresses and ports. This module is IP version independent and is able to to anonymize IPv4 as well as IPv6 flows.

## 7 FlowMon probe capabilities

The phase one helped us to localize the bottlenecks of the original design. Many problems were discovered during tests on real network. Such as old input buffers were unable to handle packets with short preamble, or packets which arrived in intensive burst. Further the old HFE had poor performance to process larger amount of packets per second. Therefore the new HFE/GENA was created and instantiated two times per one interface.

HW parameters of FlowMon probe:

- monitoring of two gigabit ports at full speed
- precise timestamp, active and inactive timeouts
- standard sampling, sample and hold
- repeater and splitter ports

SW parameters of FlowMon probe:

- export in NetFlow v5, v9
- anonymization and per collector filtering

## 8 Testing with NetFlow collectors

During development of FlowMon probe, we have used different collectors to collect and store exported NetFlow data from FlowMon probe. The main effort was focused on NfSen collector<sup>2</sup>, but we tested and used FTAS collector [FTAS] and Ntop<sup>3</sup> as well.

Important set of features, which should be supported by collector:

- ability to process NetFlow version 5 and NetFlow version 9 datagrams,
- process NetFlow data at high speeds,
- well arranged graphical output of measured data,
- possibility to process historical NetFlow data.

### NfSen collector

NfSen collector is a graphical web based front end for the NFDUMP<sup>4</sup> - tool for

---

<sup>2</sup><http://nfsen.sourceforge.net>

<sup>3</sup><http://www.ntop.org>

<sup>4</sup><http://nfdump.sourceforge.net>

collecting and storing NetFlow data. The main advantages of this collector are well arranged output of the most important network traffic characteristic, storing data in RRD (*Round Robin Database*), possibility of detailed processing of NetFlow data during specified time period, working with user defined profiles and powerful filtering of NetFlow data based on various characteristics.

We are using this collector for more then a half year, so after longer period of testing we can say that NfSen together with FlowMon probe works without any problems. This is also our preferred collector, when we deploy FlowMon probe to external testers.

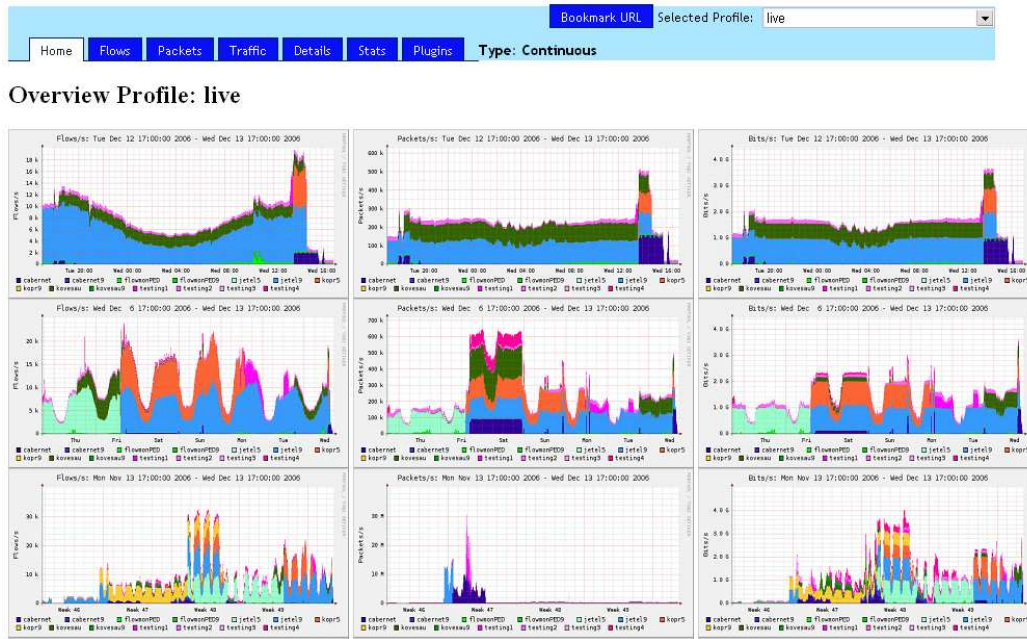


Figure 7: NfSen preview page

There are some things, which need to be mentioned. At the side of collector, actually a support of sampled NetFlow data and correct displaying of them is not implemented.

### FTAS collector

FTAS (*Flow-Based Traffic Analysis System*) is an experimental traffic analysis system which aims to give both short-term and long-term view at network traffic. It is database based solution and for displaying of gained data users need to run queries on a database. It has a powerful filtering tool and all processed queries could be visualized by a several types of graphs.

We have used this solution at the first phase of developing and testing FlowMon probe, but in these days FTAS doesn't have capabilities for displaying aggregated



**Figure 8:** NfSen detail page

longer term statistics (e.g. average traffic during some period), so we decided to use NfSen.

## Ntop

Ntop is an advanced tool for analysis of network traffic, based on Linux libpcap library. Via web based front end it displays a current state of network and with special extending module for NetFlow it is enabled to process NetFlow data.

During our tests with this collector we have found out that the big advantage of this solution is almost real time displaying of NetFlow data. On the other hand, problem was with processing of NetFlow version 9 data, which Ntop NetFlow extension module doesn't support correctly.

## 9 Future improvements

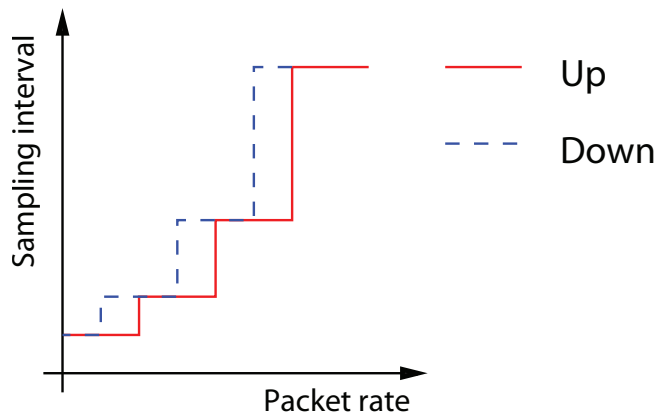
When comparing phase one and FlowMon, most improvements were focused to increase throughput of firmware design or to make software transparent and modular. As these goals were already achieved we would like to extend capabilities in other directions.

First of all adaptive sampling at the network interface will be added. Such type of sampling guarantees optimal resource utilization through various traffic mix. There are two parameters (packets per second and memory utilization)

according to which the adaptation takes place. The adaptation function is implemented as a look-up table. Each row contains lower and upper boundary of parameters and the sampling rate. The unit works as follows:

- The table is initialized by software.
- Row-pointer is set to zero, which means that row zero is currently addressed.
- Input parameters are compared with lower and upper boundaries.
- If parameters exceeds lower/upper boundary given in the row then the row-pointer is decremented/incremented and thus new sampling rate is selected. Go to previous point.

Note that it is up to user how to initialize the table. But it is recommended that he keeps certain hysteresis when setting the boundaries. So the row-pointer remain steady for longer time after change.

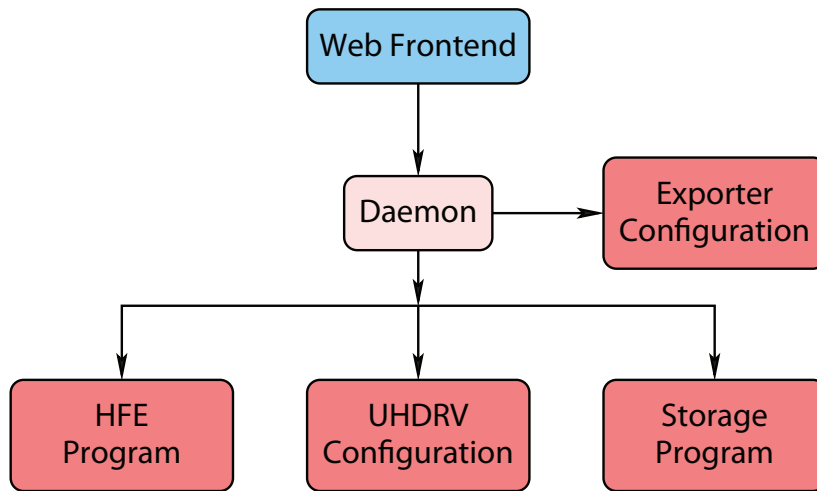


**Figure 9:** Hysteresis of boundaries

Next extension will address the issue of huge amount of active flows. Today the FlowMon probe is able to keep 64000 simultaneous flows which is insufficient for large scale networks. Therefore it is planned to change SSRAM (4MB) memories for SDRAM (512MB). Unfortunately SDRAM memories have longer delay between data request and data acquisition. The delay can be shortened using two level system of cache and SDRAM. The success of such an approach depends on a deep understanding of the dynamical characteristics of the incoming data. In the case of network traffic the most important property is temporal correlation (locality). To this end, our simulation and verification group performed measurements on traffic samples taken from several points in the CESNET backbone and Brno campus network in order to do realistic estimates for the temporal

locality of the typical network traffic. The parameter of interest is the flow gap, i.e., the number of foreign packets between two packets of the same flow. You can find more about their work in [KRA06]. The results of their work will be utilized in a new memory-efficient design of the FlowMon probe.

The improvements will also focus on flexibility of flow record. To this end, the firmware flow record is static which was enough for protocol NetFlow v5, but is too constraining for new protocols like NetFlow v9 or newly standardized IPFIX. This is maybe the most tricky extension as it requires to implement whole framework to coordinate firmware and software parts.



**Figure 10:** Framework structure for variable flow record.

## 10 Conclusion

The FlowMon probe is tested on the access link connecting the Brno Academic Computer Network to the CESNET2 backbone and CESNET backbone in Prague. The results collected by NfSen and FTAS are very promising.

Supported platforms are COMBO6X with COMBO-4SFPRO and COMBO6X with COMBO-2XFP. The FlowMon firmware now works at 100 MHz, which results in the theoretical maximum throughput of 3.2 Gb/s. This improvement was achieved mainly by duplicating the input processing part and using new input buffers and new HFE2 (GENA). These goals were stated in our technical report last year and were successfully fulfilled. So today we are able to monitor two gigabit interfaces at wire speed no matter of packet size.

The flow cache currently has room for 64 thousand flows but this capacity will soon be increased to 256 or 512 thousand flows. The almost ten-fold increase in

flow cache capacity should allow to monitor all flows with higher precision and to absorb parts of DoS attacks.

Our plans concerning the firmware are to enhance the design in the direction of network traffic adaptivity, memory efficiency and IPFIX compliance by means of supporting variable record format.

## 11 References

### References

- [CEL05] Čeleda P., Kováčik M., Krejčí R., Kysela J., Špringl P.: *Software for Net-Flow Monitoring Adapter*. Technical report 33/2005<sup>5</sup>, Praha: CESNET, 2005.
- [EV03] Estan C. and Varghese G. New directions in traffic measurement and accounting. In: *Proc. ACM Transactions on Computer Systems (TOCS)*, ACM, 2003, p. 270-313.
- [FTAS] Košňar T. *Flow-Based Traffic Analysis System - Architecture Overview*, Technical report 15/2004<sup>6</sup>, Praha: CESNET, 2004.
- [KRA06] Kramářeková M., Jakubík D., Žádník M. and Šafránek D.: *Modelling, simulation and analysis of packet localization in time*. Technical report 32/2006<sup>7</sup>, Praha: CESNET, 2006.
- [RFC3917] Quittek J. et al. Requirements for IP Flow Information Export. RFC 3917<sup>8</sup>, IETF, 2004.
- [ZAL05] Žádník M., Lhotka L.: *Hardware-Accelerated NetFlow Probe*, Cesnet technical report 32/2005<sup>9</sup>, Praha: CESNET, 2005.

---

<sup>5</sup><http://www.cesnet.cz/doc/techzpravy/2005/netflow>

<sup>6</sup><http://www.cesnet.cz/doc/techzpravy/2004/ftas-arch/>

<sup>7</sup><http://www.cesnet.cz/doc/techzpravy/2006/netmix>

<sup>8</sup><http://www.ietf.org/rfc/rfc3917.txt>

<sup>9</sup><http://www.cesnet.cz/doc/techzpravy/2005/nflowhw>