Adaptive Traffic Processing for High-Speed Networks

Lukáš Kekely, Viktor Puš, Jan Kořenek
CESNET a.l.e.
Zikova 4, Prague, Czech Republic
Email: kekely,pus,korenek@cesnet.cz

***Abstract*—The raising speeds of computer networks combined with ever-increasing complexity of networking applications create a need for smarter ways of traffic processing acceleration. In this paper, we propose a novel concept of application-driven auto-adaptation mechanism to dynamically divide network traffic processing between CPU and hardware accelerator. Feasibility of proposed approach is demonstrated on the realization of practical flow monitoring application for high-speed networks.**

I. Introduction

The aim of our work is to use auto-adaptation to the actual characteristics of network traffic as part of hardware/software codesign concept enabling creation of effective packet processing systems. Our ultimate goal is to achieve improved performance as compared with CPU-only implementation and/or enhanced flexibility as compared with purely hardware (ASIC or FPGA) implementation.

From the architectural point of view, the packet processing system consists of: (1) a general CPU and (2) a hardware accelerator placed *before* it on the path of incoming packets. From the functional point of view, a packet processing is split into: (a) complex tasks performed by a software program running at CPU and (b) simpler operations offloadable into hardware accelerator. Such arrangement generates interesting problem of effective mapping of processing tasks (a) and (b) onto the computational resources (1) and (2) for different groups of packets, where auto-adaptation can prove beneficial.

Our concept supposes that the decision whether some portion of network traffic *should* be offloaded to the accelerator is non-trivial and therefore it must be done by the software program. Therefore, we introduce an adaptation feedback loop from the CPU to the accelerator and also all new unknown packets must be forwarded to the CPU. Once the software decides that the desired processing of a particular portion of network traffic can be offloaded to the accelerator, the feedback loop is used to adapt the behavior of accelerator accordingly. From that moment on, the CPU is spared the burden of processing that traffic portion, leaving more time for other tasks.

II. System Design

Top-level scheme of the proposed system concept is shown in Fig. 1. Fast data path is shown as solid arrows, while the adaptation feedback loop is represented by dashed lines. The system is composed of two main parts: hardware accelerator and CPU, connected together through a data bus.

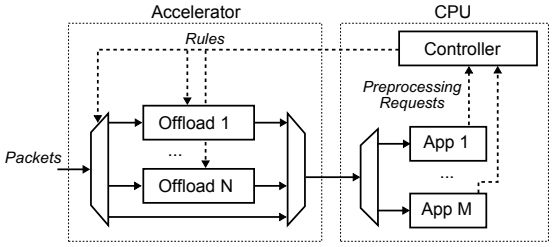

Fig. 1. Top-level view of proposed system concept

The modular architecture of hardware accelerator contains multiple processing offload modules which may be configured to perform various tasks. The distribution of packets among these modules is configurable as well. The software application plugins (App 1 to App M) perform advanced packet processing tasks and control the offloading of portions of traffic. Their decisions are forwarded to the controller, which aggregates them and adapts the accelerator behavior accordingly. The aggregation of offloading requests is done in a way ensuring that information requirements of all applications are satisfied, so no loss of interesting data occurs.

III. Experimental Results

To prove the soundness of proposed approach, we have implemented a complete proof of concept prototype. Using the NFB-100G1 FPGA board [1] plugged into a commodity server, we have created an instance of our concept performing accelerated 100 Gbps network flow monitoring enhanced by software support of application layer (L7) processing. Conceptual scheme of the prototype is shown in Fig. 2.

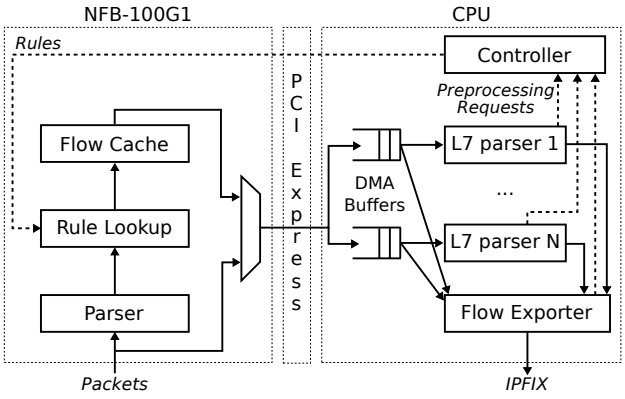

Fig. 2. Implemented 100 Gbps flow monitoring system

The processing of all incoming packets starts with parsing of their headers and extracting metadata from them (Parser). Extracted metadata is then used to assign the packets into corresponding processing groups (network flows) based on a software controlled set of rules (Rule Lookup). Packets can be processed in accelerator's flow cache (i.e. aggregated to the selected type of flow record) or sent to the software unchanged. Flow records from the flow cache are periodically exported to the software. The data from the accelerator is sent over the PCI Express bus to the software using multiple independent channels to support multi-core processing. This data is processed by an arbitrary set of L7 protocol parsers (we used HTTP and DNS) and the flow exporter which analyze the received data and exports the flow records to a flow collector. The software modules also on the fly specify which portions of traffic they want to inspect and which might be offloaded into accelerator.

Since the accelerator's flow cache has limited capacity, it must be filled with care to enable accelerated processing of as many packets as possible. Generally, it is advantageous to select the currently biggest groups of packets (heaviest flows) for offload into the cache. Of course, only if no application plugin expressed its interest in them. The simplest method to recognize the heaviest flows on network is based on a rule that every flow is considered to be among the heaviest after arrival of its first k packets for some selected decision threshold k. We further extend this rule by automatically adapting the value of k in reaction to the changing characteristics of network traffic in time. The adaptation is based on the current load of the accelerator's flow cache. For the best offload ratio, it is advantageous to keep the flow cache nearly full at all times. That way, there is still some space left for new heavy flows, while the amount of offloaded traffic is maximized. So, the value of heavy flow decision threshold is decreased when the flow cache utilization drops below a specified point and increased when the flow cache becomes nearly full.

Graphs in Fig. 3, 4 and 5 show courses of various prototype parameters during a whole day of its deployment in our real backbone network Cesnet2. Packet offloading ability is presented in the first graph. During the whole day, the majority of all received packets (solid line) is processed by accelerator (dashed line), leaving only a small portion for software processing (dotted line). Offloaded portion of packets is always between 70 and 85 % of total traffic and is shown in gray shade bar at the bottom of the grid. With this offload ratio, we are able to achieve the CPU utilization reduction of 2 to 3 times. The second graph shows the utilization of (number of rules in) accelerator's flow cache compared to the total number of active flows in the network. Gray area demarcates a desired flow cache load maintained by the adaptation of heavy flow decision threshold. The adaptation of the threshold value is illustrated in the third graph. During the heaviest network load, the threshold value raises and when the load starts to decline the threshold value follows.

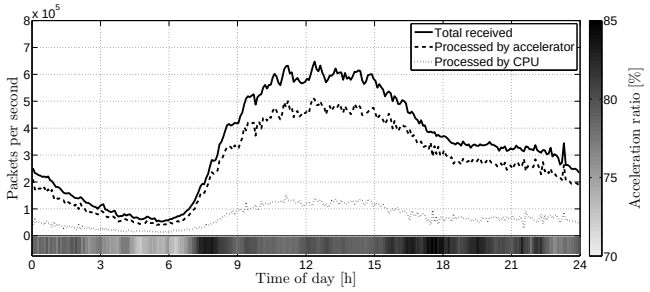

Fig. 3. Prototype deployment: processing of packets

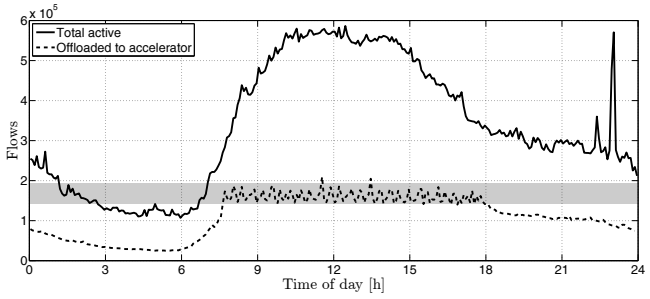

Fig. 4. Prototype deployment: total and offloaded flows

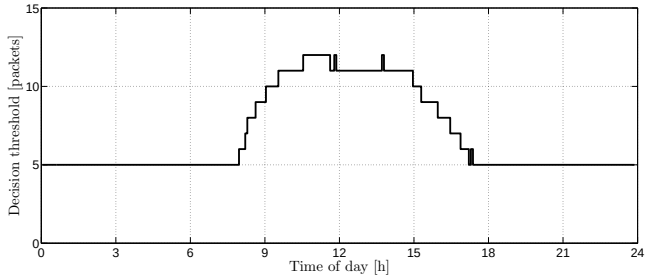

Fig. 5. Prototype deployment: value of heavy flow decision threshold

IV. Conclusion

In this paper we proposed a general concept of hardware/software codesign with auto-adaptation features for high-speed network traffic processing. We showed the feasibility of proposed concept on implementation of its concrete instance – flow monitoring system with application layer parsing support. The implemented system was, thanks to adaptation, capable to accelerate processing of around 80 % of incoming packets, bringing reduction of CPU load by 2 to 3 times. Our future work include creation and evaluation of systems based on proposed concept performing other networking task (e.g. firewall, intrusion prevention, anomaly detection).

Acknowledgment

This research has been partially supported by the "E-infrastructure CESNET" project no. LM2015042 funded by the Ministry of Education, Youth and Sports of the Czech Republic, and the project TH01010229 funded by the Technology Agency of the Czech Republic.

References

[1] Netcope Technologies, "NFB 100G1 FPGA Board," 2016. [Online]. Available: https://www.invea.com/en/products-and-services/fpga-cards/nfb-100g