Next Generation FPGA-Based Platform for Network Security

Alexander P. Antonov, Alexey S. Filippov, Olga V. Mamoutova
Peter the Great St. Petersburg Polytechnic University
St. Petersburg, Russia
{antonov, filippov}@eda-lab.ftk.spbstu.ru, mamoutova@kspt.icc.spbstu.ru

***Abstract*—The driving forces for this project are demands for a hardware, reconfigurable, high performance, low cost and low power solution that could be used in a widest range of Internet of Things (IoT) devices. The solution should be able to secure computer networks against emerging threats and vulnerabilities, sustaining privacy and trust. This paper presents an approach for developing a novel hardware platform for Ethernet-based network firewall security services for IP networks. The article highlights functional and structural levels of the proposed hardware architecture, performance estimations and a trade-off between performance and hardware cost. Some implementation details, including HDL used, testing approaches and design tools are provided as well.**

I. Introduction

The innovative field of telecommunications brings new trends and technologies with ever increasing speed: cloud computing, virtualization, Internet of Things (IoT) and Internet of Everything, modern wireless solutions, etc. [1]. With explosively expanding network connectivity the traditional question of cyber security (the ability to properly secure computer networks against emerging threats and vulnerabilities, sustaining privacy and trust) becomes even more important, both for personal and enterprise uses.

Firewalls address this problem and provide an effective first-level barrier of the network security to support an established (presumably well-crafted) security policy. A firewall is a utility that connects two networks and filters network traffic between the two networks according to the provided security policy. In general, its aim is twofold: firstly, the firewall protects data and computing infrastructures of the protected network from external malicious attacks, and secondly – restricts unwanted accesses from the protected network to the external one (Fig. 1).

However, there are two main problems [2]. The first one is that the number of different types of malicious network attacks constantly increases and requires the growth of applied security policies complexity. The second one is that with the introduction of IPv6 and the speed of network connections increasing up to 1G and 10G the volume and throughput of network traffic increases as well. This puts a huge computation load on a firewall and makes it a bottleneck of a network infrastructure.

The common criteria of a good firewall protection are:

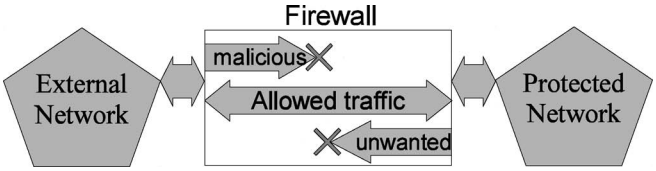

Fig. 1. Concept of firewall

- how high is the speed of applying the rules;
- how easy is the maintenance of the rules database;
- how secure the firewall itself is.

Software firewalls can no longer satisfy the demands of critical network applications. Lately a popularity of the field programmable gate arrays (FPGA) technology enabled development of hardware firewall solutions. Advantages of a hardware firewall are high performance, low latency, low cost and low power for "green communications". Those advantages result from the ability of a single compact device to process a large number of concurrently analyzed fields and rules, and a large number of concurrently analyzed network channels. A disadvantage of such a hardware firewall is a lack of scalability – it can be used for perimeter firewall models only, where either one, or a limited set of firewalls see all the traffic coming to and from the protected network. The increase of performance and simplification of maintenance make up this deficiency of scalability though.

This paper presents a new FPGA-based platform for network security devices, designed for firewall and intrusion detection functions. The goal of this work is to have a firewall device, which will be easy to setup and will have secure configuration and reconfiguration procedures. Although the implementation of packet filtering and other algorithms constitutes an important part of current research in the field, this paper focuses mainly on the architecture of the platform, which aim is to provide a possibility to implement the widest range of algorithms.

Another aim of the platform is to support the modern trend of system-wide collaboration for a network security task [3], when individual applications get a global view of dynamic security situations through the multi-point collaboration over the network structure.

The rest of the paper is organized as follows. Section II gives an overview of basic firewall terms and investigates the

current hardware implementations. Section III highlights concept, top-level architecture of the proposed firewall platform and functional diagram of the core unit of the platform. Section IV contains more detailed descriptions of internal architecture of core unit, some performance estimations, possible trade-offs between performance-hardware cost, and gives some basic information regarding implementation details (hardware description language used, tools). Summary is presented in Section V.

II. State of the Art

A. Firewalls – basic terms and definitions

A *computer network firewall* is a component or set of components that restrict access between a protected network and the Internet, or between other sets of networks [4]. This means that a firewall serves as a *rule checker* for traffic coming in and out of the network under protection. A set of defined rules (SoDR) guarantees that the traffic conforms to the desired *security policy*. Different types of firewalls vary in functionality, application and supported network architecture. This section covers the necessary basic principles of firewall implementations.

Simple firewall can be done as a *packet-level filter*, which selectively accepts or denies packets according to information contained in the header (simple packet filter) or in the body of the packet (*deep packet filter*). Such filtering is usually done at network and transport layers. For example, a firewall can block all connections from a certain untrusted network based on the IP source address field of the packet. Another example of a simple packet filtering is an inherent function of VPN implementations aimed to determine which packets will get sent through the VPN. A virus or spam protection, which searches for certain patterns in the packets' data, is an example of a deep packet filter.

Another elaborate dimension of a firewall complexity is a *stateful firewall* that employs state machines to track protocol connections. In this case a firewall tracks and manages a sequence of packets that belong to one connection, like in a TCP protocol.

Usual auxiliary functions of a firewall are network address translation (NAT), event logging, intrusion detection/prevention and quarantining.

Many firewalls are created for protection of a single host. Usually such a personal firewall is a kernel-level software on a user's workstation. It monitors network activity and blocks suspicious processes. An alternative to workstation firewall could be an appliance firewall – a dedicated device external to the host that is to be protected. Opposed to a *localized* instrument, a firewall can be *distributed*, for example over several hosts of a network, and employ security policies by several individual network endpoints.

A firewall can act as a "bump in the wire" in case of a *bridging firewall*, which works transparently to the rest of the network elements. Or a firewall can operate with a dedicated IP address and replace a traditional router in case of a *proxy firewall*, which on demand creates and manages client's connections to the remote network destination.

In most cases a firewall runs on the same hardware as user-level applications. To keep up with high-speed networks the network protection tasks typically require multiple dedicated multi-core processors, which still have a limited performance and are power-hungry. During peak network utilizations and/or for critical applications low latencies and high throughput of the firewall may become indispensable. A hardware acceleration of firewall functions can address this need.

The growth of FPGA technology enabled a hardware implementation of firewalls and intrusion detection systems [3]. Such hardware accelerations provide a flexible implementation of a highly desirable parallelism for a packet data processing with extensive sets of rules. It has been shown that hardware packet-filtering engines provide capacity and speedup, which are impossible for fully-software implementations.

B. Hardware firewall solutions

At the moment a lot of research and design efforts both from academia and industry go towards hardware implementation of various firewall functions. This section covers questions of some recent research efforts in the field with the focus on architectures of the solutions. The common research goals are to make hardware designs as compact and efficient as possible and to have a firewall solution, which will be simple to support and update.

A typical scheme for implementation and prototyping is to have an FPGA with a firewall engine as a co-processor in embedded solution. Thus a management task lies on a software part of the system. An example is a hardware based firewall and a rate-limiting engine by Park et al. [5], where filtering is implemented on FPGA and a rule management is software-operated in embedded CPU. Another example is a processor-based embedded system with real-time operating system by Ajami and Dinh [6], which is designed to achieve highly customized and on-the-fly configuration change in the firewall. Thinh et al. [7] have created a fully-hardware architecture for the virus signature matching engine, where a system update requires only an update of on- and off-chip memory contents.

Another approach is to provide an ability to change a configuration of FPGA for the necessary firewall updates. Dramicanin et al. [8] explore an interactive reconfigurable firewall concept. Jedhe et al. [9] describe an embedded system on FPGA-based platform, running Linux with the packet classification in hardware, where the firewall rule update involves only memory re-initialization in software without any hardware change.

The next approach, which targets firewall maintenance, is to automatically generate the hardware components of the design in response to changing security policies. The motivation behind this approach is the reliability of a firewall solution. In order to simplify the updates during the support stage of a firewall lifecycle, the process of manual design of the hardware is fully eliminated. For example, Kayssi et al. [10] present a software tool, which translates a set of rules into hardware VHDL blocks for packet filtering.

Another topic is an effective mapping of a certain algorithm onto FPGA resources. For example, Cho and Mangione-Smith [11] achieve a significant shrink of the size of FPGA deep packet filters, which perform complex data search and analysis, by sharing the common sub-logic and using a memory-based architecture. Prasanna, Jiang et al. have done an extensive work [12]–[16] on packet classification engine for FPGA, which provides dramatically reduced memory requirement. Jedhe et al. [9] achieve a scalable and high performance architecture by using a reconfigurable hardware implementation of Extended Ternary Content Addressable Memory. Sato et al. [17] present an implementation of a wave-pipelined firewall to achieve high clock frequencies.

The driving force for presented project is the necessity to have hardware reconfigurable, high performance, low cost and low power consuming solution that could be used in a widest range of IoT devices.

III. Firewall platform

A. Concept of firewall platform

Top-level architecture (Fig. 2) of the proposed firewall platform is based on System-on-Programmable-Chip (SOPC) devices provided by Altera Inc. or Xilinx Inc.

Key features of the platform are hardware re-configurability and extensibility. These features are achieved by using SOPC devices, expansion modules and "soldering-by-demand" components.

Three basic configurations of the platform are defined.

The smallest configuration (SC) comprises Field Programmable Gate Arrays (FPGA), Ethernet Network 1 & 2, and SD card. SD card keeps SoDR and some statistical data from SoDR engine. Personal Computer (PC) with special-purpose Graphic User Interface (GUI) is used for programming SoDR and for statistical data analysis and visualization.

The typical configuration (TC) comprises Field Programmable Gate Arrays (FPGA), Ethernet Network 1 & 2, SD card, DDR3 memory (up to 1GB), Ethernet (ETH) and/or USB connectors for remote control. SD card keeps operating system (OS) for embedded processor, SoDR and some statistical data. In this configuration, an external PC with special-purpose GUI should be connected to ETH or to USB for SoDR programming and for statistical data analysis and visualization.

The expanded configuration (EC) could include up to 16 additional Ethernet connectors for protected and external networks respectively and additional DDR3 memory (up to 1 GB).

SC and TC configurations are intended for "single-point" network connection (Fig. 3). In "single-point" network connection there are one connector for the protected network and another one for the external network. SC could be used for the smallest embedded applications, including mobile applications such as robotics, which need high security level and have fixed, or nearly fixed, network environment. TC well suits embedded devices with additional requirements for simplicity of rebuilding and reprogramming SoDR, i.e. for variable network environment.

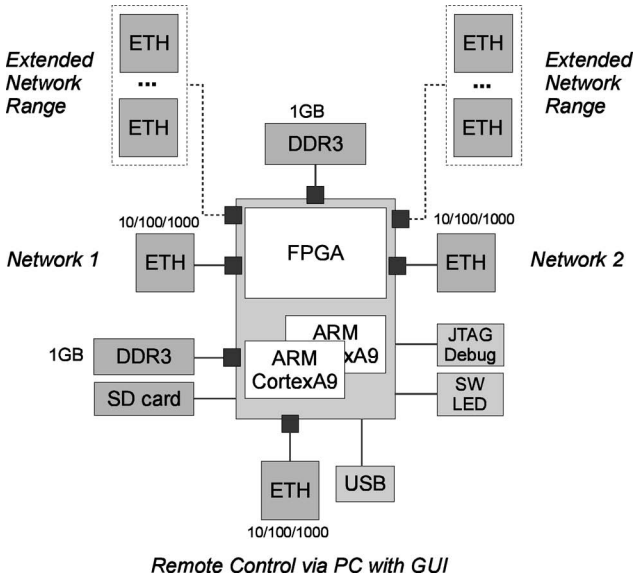

Fig. 2. Top-level architecture of firewall platform

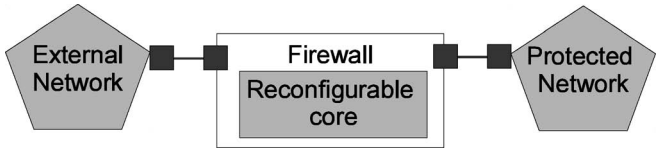

Fig. 3. "Single-point" network connection

EC is intended for "Multi-point" network connection (Fig. 4). In "Multi-point" network connection there are some connectors for external and protected networks. It could have one or several reconfigurable core(s) implemented in FPGA. This configuration well suits complex network security solutions with high-speed processing, small form-factor and low cost demands.

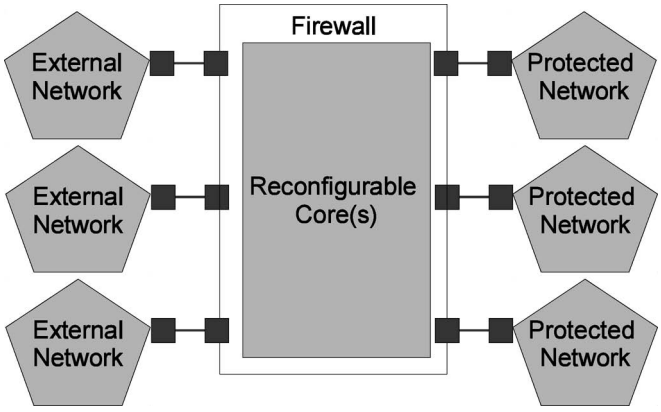

Fig. 4. "Multi-point" network connection

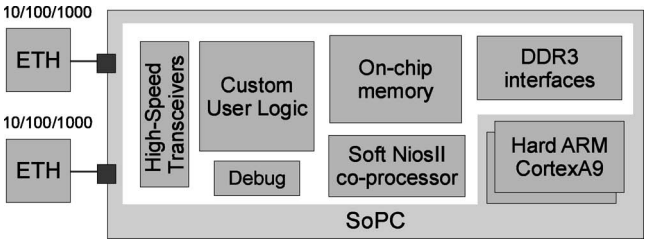

Fig. 5. Reconfigurable core of firewall platform

The reconfigurable core for the firewall platform (Fig. 5) comprises at least logic array (Custom/User Logic), Debug unit and High-Speed Transceivers. Generally, it could include DDR3 interface(s), soft and/or hard processors and on-Chip memory as well.

FPGA based solution could be unsuitable for the applications with high volume manufacturing demand. For such application, a chip scale solution (Application Specific Integration Circuits – ASIC) should be used. In this scope, the proposed platform provides extremely good possibilities for ASIC prototyping.

The platform has a small form factor (approximately 2 by 3 inches). It could be reconfigured from SC to TC by soldering additional devices and from TC to EC by connecting expansion modules.

B. Architecture of reconfigurable core

The architecture of the reconfigurable core from the functional point of view (Fig. 6) includes:

- Ethernet Interface Unit – responsible for connection with Protected and External Networks;
- Delay Lines – for Ethernet packet buffering;
- & – gate AND;
- PPar – Ethernet packet Parsers. Unit is responsible for reading values of fields and parameters extraction;
- SoDR – Set of Defined Rules;
- RCE – Rules Checker Engine. Unit provides a decisions on filtering;
- Control Unit – Service functions: rules database (SoDR loading), data collection (getting statistical data from RCE/PPar), and maintenance interface over remote Ethernet connection or USB security connection.

IV. Implementation Details

A. RCE architecture

The most time critical component, from the overall performance point of view, is RCE. That is because it should compare "properties" – values/rangers of values of the current Ethernet packet fields with hundreds of rules.

A rule is defined as a set of fields, which need to be compared with parameters, and values of incoming Ethernet packet. The fields could have different lengths (from 8 bits up to 48 bits). The number of the fields defines a complexity of the single rule (Table I).

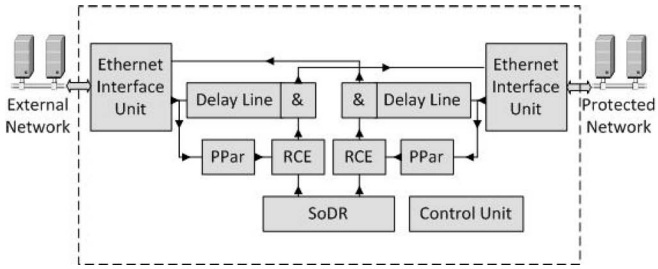

Fig. 6. Functional structure of reconfigurable core

TABLE I. Fields of Rules

Field	Length (bits)
Dstmac	48
Srcmac	48
Ipproto	16
Ethproto	8
srcip4	32
dstip4	32
Srcport	16
Dstport	16
Icmp/type&code	16

Each of the fields could be compared with a value of incoming Ethernet packet in accordance with any of the following approaches (Decision Type):

- AV – any value (up to 4 different values);
- EN – enumeration (a set of consecutive values);
- RA – range of values.

To achieve a maximum throughput two approaches for building RCE should be applied: concurrency and pipelining.

Having nine fields in each rule and up to 1k rules in total makes it necessary to build RCE, which consists of Field Decision Units (FDU) integrated in Decision Lines (DL), up to 1k DLs, and Decision Make (DM) unit generating the final decision (Fig. 7).

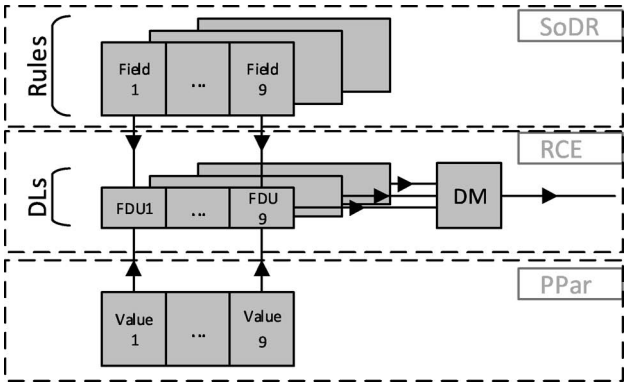

Fig. 7. Top-level structure of RCE

RCE consists of up to 1k DLs working concurrently with pipeline stage between DLs and DM. Each DL is nine concurrently working FDUs (Fig. 8). The architecture of FDU

is a pipeline of the following units: AV, EN, RA (working concurrently) and D – decision unit. The algorithm of the unit D pays attention to AV, EN, RA results and Decision Type value.

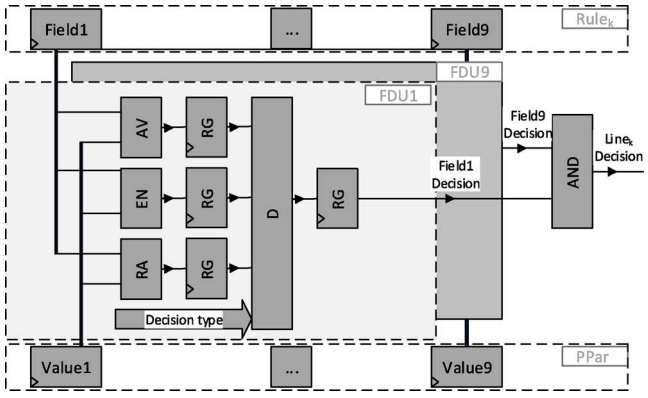

Fig. 8. Structures of DL and FDU

B. Performance-area trade-off

The architecture presented above, which is a fully concurrent implementation of RCE, provides the highest performance (measured in number of PPar per second processed). Performed evaluations show that:

- One PPar is processed per one clock cycle.
- Delay of PPar processing is six clock cycles.
- Maximum frequency (minimum clock cycle) is about 300 MHz (is about 3 us). It is highly dependent on the target device family and the device's speed grade. The performance is practically identical for devices from Altera and Xilinx having the same parameters.

The drawback of the fully concurrent implementation of RCE is the cost of hardware. It could be measured in Logic Elements (LE) – key functional units of any FPGA. LE comprises one Look-up-table with four-inputs and one flip-flop. The number of LEs could be easily translated to money, since this number is directly proportional to the cost of the device.

From the other hand, for "single-point" network connection, even if we would have Gigabit Ethernet connections, the maximum frequency of PPar arriving is less than 1 MHz.

It means that we can dramatically reduce the hardware cost of RCE implementation without the performance degradation. The obvious approach is to use a parallel-sequential pipelined architecture.

Taking into account that there is an overhead for sequential implementation (when one DL is used recursively) of RCE, we used the following estimation: if there are 256 Rules and PPar frequency is equal or below 1 MHz, then one DL will be sufficient for RCE implementation.

Estimations of PPar frequency for given DL and Rules numbers are provided in Table II. Where "x" means that PPar frequency is insufficient, i.e. below 1 MHz.

This table could be used as an entry point for finding trade-off between hardware cost and RCE performance.

Having a positive slack between desired PPar frequency and achievable performance (PPar frequency highlighted in Table II) it is possible to find trade-off between performance, hardware area and power consumption. Taking in mind that power consumption is directly proportional to the frequency it is possible to reduce power consumption of RCE by reducing working frequency. Table III provides estimations of RCE's working frequency for the following conditions: PPar frequency equals 1 MHz, given DL and Rules numbers. Here "x" means unsupported conditions.

TABLE II. PPAR FREQUENCY

PPar Frequency (MHz)		Rules number			
		128	**256**	**512**	**1024**
DLs number	**1**	2	1	x	x
	2	4	2	1	x
	4	8	4	2	1
	8	16	8	4	2
	16	32	16	8	4
	32	64	32	16	8
	64	128	64	32	16
	128	256	128	64	32

TABLE III. WORKING FREQUENCY

Working Frequency (MHz)		Rules number			
		128	**256**	**512**	**1024**
DLs number	**1**	128	256	x	x
	2	64	128	256	x
	4	32	64	128	256
	8	16	32	64	128
	16	8	16	32	64
	32	4	8	16	32
	64	2	4	8	16
	128	1	2	4	8

This table could be used as an entry point for finding the lowest acceptable working frequency for the given conditions.

C. Implementation details

The pointed fully concurrent-pipelined architecture and a set of parallel-sequential pipelined architectures are implemented in System Verilog Hardware Description Language (SV HDL). SVHDL is selected because it provides extended (comparing with Verilog and VHDL) possibilities for the structure description and manipulating. SVHDL is used for the logic (unit under development) and testbenches descriptions. Basic building units are developed as reconfigurable units that help to simplify further performance-area trade-off researches. To accelerate experiments with different architectures some self-checking testbenches have been developed. Such tests save a lot of engineering efforts by simplifying and accelerating a procedure of checking modified versions of the architectures under investigation.

Regarding EDA, embedded synthesis tools of QuartusII (ver.14.0) – Altera's development tool, and Vivado (ver. 2014.2) – Xilinx's development tool have been used. Both QuartusII and Vivado include Place and Route and Timing Estimation tools as well.

As a simulation tool ModelSim Altera Started edition and simulation tool embedded in Vivado have been used.

VII. Conclusion

A lot of research and design efforts are being done towards hardware implementation of various firewall functions. The common goal is to develop a hardware solution which is as compact and efficient (in terms of performance, hardware cost, power consumption) as possible.

Key features of the proposed firewall platform are hardware re-configurability and extensibility. These features have been achieved by using SOPC devices, expansion modules and "soldering-by-demand" components. The platform can be used for manufacturing a wide range of embedded solutions with high security demands and for prototyping purposes in ASIC development flow.

The proposed architectures and approaches can be used for building very high performance solutions and cost optimized systems, with low power consumptions as well. Reconfigurable modules and self-checking testbenches developed on System Verilog allow to reduce engineering efforts during further architecture investigations and optimization.

The further efforts towards building a hardware implemented firewall include PPar unit architecture development, analysis and optimization. A proof of concept implementation of the firewall with presented architecture has been successfully tested on a simple set of rules using standard development board and under synthetic network load. Thorough field-test experiments are planned for the future work.

Acknowledgment

This work is supported by scientific-educational center "Embedded Microelectronic System" of Peter the Great St. Petersburg Polytechnic University.

References

[1] E. M. Neira, "Top 10 Communications Technology Trends in 2015," *IEEE ComSoc Technology News*, 2015.

[2] C. Douligeris and D. Serpanos, "Designing Firewalls: A Survey," in *Network Security: Current Status and Future Directions*, Wiley-IEEE Press, 2007, pp. 33-49.

[3] H. Chen, Y. Chen, and D. H. Summerville, "A Survey on the Application of FPGAs for Network Infrastructure Security," *IEEE Communications Surveys Tutorials*, vol. 13, no. 4, pp. 541-561, Fourth 2011.

[4] E. D. Zwicky, S. Cooper, and D. B. Chapman, *Building Internet Firewalls*, 2nd ed. Beijing-Cambridge: O'Reilly, 2000.

[5] S.-K. Park, J.-T. Oh, and J.-S. Jang, "High-speed attack mitigation engine by packet filtering and rate-limiting using FPGA," in *Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference*, 2006, vol. 1, p. 6 pp.-685.

[6] R. Ajami and A. Dinh, "Design a hardware network firewall on FPGA," in *2011 24th Canadian Conference on Electrical and Computer Engineering (CCECE)*, 2011, pp. 000674-000678.

[7] N. T. Thinh, T. T. Hieu, H. Ishii, and S. Tomiyama, "Memory-efficient signature matching for ClamAV on FPGA," in *2014 IEEE Fifth International Conference on Communications and Electronics (ICCE)*, 2014, pp. 358-363.

[8] D. M. Dramicanin, V. Pejovic, and Z. Petrovic, "On Design and Exploitation Strategies of Reconfigurable Firewalls," in *8th International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Services, 2007. TELSKS 2007*, 2007, pp. 597-600.

[9] G. S. Jedhe, A. Ramamoorthy, and K. Varghese, "A Scalable High Throughput Firewall in FPGA," in *16th International Symposium on Field-Programmable Custom Computing Machines, 2008. FCCM '08*, 2008, pp. 43-52.

[10] A. Kayssi, L. Harik, R. Ferzli, and M. Fawaz, "FPGA-based Internet protocol firewall chip," in *The 7th IEEE International Conference on Electronics, Circuits and Systems, 2000. ICECS 2000*, 2000, vol. 1, pp. 316-319 vol.1.

[11] Y. H. Cho and W. H. Mangione-Smith, "Deep packet filter with dedicated logic and read only memories," in *12th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, 2004. FCCM 2004*, 2004, pp. 125-134.

[12] W. Jiang and V. K. Prasanna, "A FPGA-based Parallel Architecture for Scalable High-Speed Packet Classification," in *20th IEEE International Conference on Application-specific Systems, Architectures and Processors, 2009. ASAP 2009*, 2009, pp. 24-31.

[13] Y. Qi, J. Fong, W. Jiang, B. Xu, J. Li, and V. Prasanna, "Multi-dimensional packet classification on FPGA: 100 Gbps and beyond," in *2010 International Conference on Field-Programmable Technology (FPT)*, 2010, pp. 241-248.

[14] T. Ganegedara and V. K. Prasanna, "StrideBV: Single chip 400G+ packet classification," in *2012 IEEE 13th International Conference on High Performance Switching and Routing (HPSR)*, 2012, pp. 1-6.

[15] J. Fong, X. Wang, Y. Qi, J. Li, and W. Jiang, "ParaSplit: A Scalable Architecture on FPGA for Terabit Packet Classification," in *2012 IEEE 20th Annual Symposium on High-Performance Interconnects (HOTI)*, 2012, pp. 1-8.

[16] A. Sanny, T. Ganegedara, and V. K. Prasanna, "A Comparison of Ruleset Feature Independent Packet Classification Engines on FPGA," in *Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW), 2013 IEEE 27th International*, 2013, pp. 124-133.

[17] T. Sato, S. Imaruoka, and M. Fukase, "Verifying firewall circuits by wave-pipelined operations," in *TENCON 2009 - 2009 IEEE Region 10 Conference*, 2009, pp. 1-6.