Speeding up data processing on firewall using FPGA

DOI: https://doi.org/10.54654/isj.v1i21.1035

Tran Ngoc Son, Nguyen Van Manh, Dinh Viet Phuong

***Abstract*— This article focuses on one of the methods to improve the data processing performance of firewalls by taking advantage of the power of FPGA (Field-Programmable Gate Array) chips in analyzing packets at the network layer instead of CPU. FPGA chips provide a flexible and powerful platform for deploying network firewall data processing functions, ranging from IP address filtering, access policy inspection, data classification to virus checking. Additionally, the research team conducted theoretical experimental research on the chip Xilinx XC7A200T, one of the popular and high-performance chip models. The results of this research are the premise for researching and designing a high-speed firewall model using FPGA.**

***Tóm tắt*— Bài viết này tập trung vào một trong những phương pháp cải thiện hiệu suất xử lý dữ liệu của tường lửa bằng cách tận dụng sức mạnh của chip FPGA (Field-Programmable Gate Array) trong việc phân tích gói tin ở lớp mạng thay cho CPU. Chip FPGA cung cấp một nền tảng linh hoạt và mạnh mẽ cho việc triển khai các chức năng xử lý dữ liệu của tường lửa mạng, từ việc lọc địa chỉ IP, kiểm tra chính sách truy cập, phân loại dữ liệu đến kiểm tra virus. Bên cạnh đó nhóm tác giả đã tiến hành nghiên cứu thực nghiệm lý thuyết trên chip Xilinx XC7A200T, một trong những dòng chip phổ biến và hiệu năng mạnh mẽ. Kết quả của nghiên cứu này là tài liệu tiền đề cho việc nghiên cứu thiết kế mô hình tường lửa tốc độ cao ứng dụng FPGA.**

Keywords – FPGA; data processing, firewall based on FPGA, speeding up processing using FPGA.

Từ khóa – FPGA; xử lý dữ liệu; tường lửa ứng dụng FPGA, tăng tốc độ xử lý sử dụng FPGA.

This manuscript is received on May 21, 2024. It is commented on May 24, 2024 and is accepted on June 7, 2024 by the first reviewer. It is commented on June 19, 2024 and is accepted on June 26, 2024 by the second reviewer.

I. INTRODUCTION

In the era of the digital revolution, the internet has become an integral part of daily life and the operations of businesses, organizations, and individuals worldwide. However, with the continuous advancement of technology, cybersecurity threats have become increasingly complex and diverse. In this context, firewalls are considered one of the most important tools for protecting network systems from external threats and intrusions [1]. Firewalls can be hardware devices, software systems, or a combination of both, and their general purpose is to prevent unauthorized or unwanted access to the system they protect. However, firewalls using traditional hardware and software solutions may face limitations in performance when dealing with complex data processing tasks and high demands. Previous firewall solutions often suffer from inflexibility in adapting to rapidly changing network environments and new types of attacks or may encounter issues with latency when processing large data volumes, leading to decreased efficiency and increased data loss potential.

Legacy firewalls often rely on traditional hardware and software solutions, utilizing specialized software running on conventional hardware such as the CPU (Central Processing Unit) [2]. These firewalls analyze network traffic based on predefined rules and policies to determine whether to allow or block specific packets. Techniques such as packet filtering, stateful inspection, and deep packet inspection are often used to enforce security policies [3]. These types of firewalls have the ability to provide basic protection for network systems, including blocking unwanted connections and preventing some simple types of attacks. They

are easy to deploy and manage, making them suitable for small and medium-sized network environments [4]. However, it can be observed that using older technology may present some drawbacks that need to be addressed:

- Performance limitations: Legacy firewalls may encounter performance limitations when faced with complex data processing tasks and high demands.

- Lack of flexibility: Traditional solutions lack flexibility in adapting to rapidly changing network environments and new types of attacks.

- High latency: Legacy firewalls may introduce latency in data processing, affecting performance and user experience.

- Limited scalability: Traditional solutions may struggle with scalability and upgrades to meet the expanding demands of network systems [5].

One of the advanced methods used to optimize and enhance the performance of network firewalls is the implementation of FPGA (Field-Programmable Gate Array) chips. FPGA is a type of electronic component that can be reprogrammed, allowing users to create flexible and efficient digital systems. This flexibility is crucial for the development and deployment of highly customizable network firewalls, enabling rapid response and high performance. [6]. Integrating FPGA into network firewall design brings significant benefits. FPGA provides fast data processing capabilities, enabling network firewalls to meet the demands of complex and diverse data processing tasks. Additionally, the flexibility of FPGA provides users with the ability to customize functions according to the specific needs of the network system, ranging from IP address filtering, access policy inspection, data classification to virus checking [7]. By applying FPGA technology to network firewall design, developers can address challenges related to performance and flexibility, thereby enhancing the responsiveness and security of the network system against external attacks and threats.

In Vietnam, most of the scientific research on FPGA applications for network packet processing focuses on developing IPSec solutions. There is no specific research in the direction of replacing CPU features to address the need for performance and data processing speed of the firewall. This article provides a specific and feasible design and implementation model for a high-speed firewall solution using FPGA.

II. USING FPGA IN NETWORK FIREWALL DESIGN

One of the primary applications of FPGA in network firewall design is packet processing. As data traverses the network, FPGAs can be programmed to perform packet processing functions such as IP address filtering, access policy checks, data classification, and virus scanning. Thanks to the flexibility and high performance of FPGAs, processing modules can be deployed to handle thousands of packets per second without degrading the system's operational efficiency [8].

FPGAs can also be utilized for network bandwidth management. By prioritizing packets based on criteria such as application, service type, or IP address, FPGA helps improve network performance and ensures that critical applications receive prioritized bandwidth. This not only optimizes user experience but also enhances the system's responsiveness to emergency situations. [9].

Additionally, FPGAs provide a flexible platform for deploying cryptographic algorithms and security protocols such as AES (Advanced Encryption Standard), SHA (Secure Hash Algorithm), and SSL (Secure Sockets Layer). Utilizing FPGA to perform security functions enhances the flexibility and performance of the firewall system, effectively protecting the network against potential threats.[10].

Furthermore, FPGAs can be used to deploy Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS), enabling rapid and effective monitoring and response to threatening network behaviors. Deployment on FPGA not only enhances detection capabilities but also minimizes response time, ensuring the safety of the network system [11].

FPGAs provide a flexible platform for integrating additional services such as data

stream analysis, resource management, and network resource management. These applications can be deployed on FPGA to provide scalability and customization features for the firewall system, thereby enhancing its adaptability and efficiency [12].

III. RESEARCH AND EXPERIMENTAL DEPLOYMENT

A. Deployment Model

Based on the idea of deploying a simulated device model featuring the functionalities of a firewall, utilizing FPGA for network layer packet analyzing, processing, and executing disconnection commands upon detecting signs of insecurity, the authors have studied a model comprising two main subsystems: the FPGA processing board and the Embedded Personal Computer (EPC) control unit. The FPGA board is responsible for replacing the Inside and Outside 2-way communication NIC network card using the SFP/SFP+ module to support and analyze each packet passing through the chip while receiving commands and processing data from the EPC according to the receiving instruction set. (specifically, this case is to block the connection of network devices with signs of suspected insecurity detected by IDS) and respond to EPC with processed events. The EPC runs software with a control interface allowing administration, configuration, rule setting, status display, connection to monitoring systems, and receiving commands from the monitoring system via an API.

1. Design of FPGA Processing Board

FPGA Central Processing Block: The main task is to receive and execute control rules from the EPC via the Ethernet Module. The rules received from the EPC are written into external RAM. The FPGA will then use the control rules in combination with the external RAM data to execute processing of digital data input/output from the data stream on the SFP/SFP+ Optical Port Block and the CAT5/CAT6 Ethernet Port Block [13]. To do this, the research team used a specific feature of the hardware description language on the FPGA - Mealy machine (finite-state machine), where the packets of the input and output streams are analyzed and checked and compared with the data. network device from the set of rules stored in RAM, if they match, the device will be immediately isolated from the LAN area using the packet dropping mechanism (as shown in Figure 1).

In essence, by using an FPGA instead of a NIC card, the chip FPGA acts as a span-port on

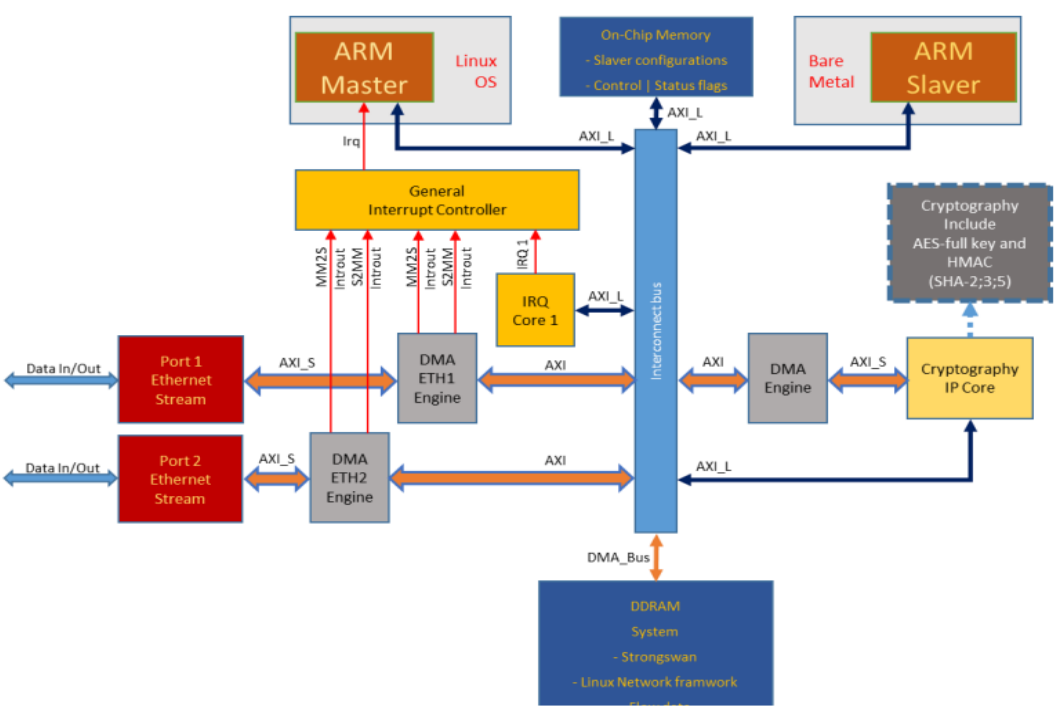

Figure 1. Block diagram of a FPGA Board

the switch. The FPGA board supports Ethernet packet communication and will be able to read and save packets passing through it with a structure similar to ".pcap" package but at much higher speed (in bytes/nanoseconds) of software applications on CPU depending on chip type.

External RAM Block: Controlled by the FPGA execution block through address registers and synchronous Clock. The external RAM block's main task is to store real-time databases from the device management and control software, serving as a data buffer for the FPGA processing block.

Ethernet Port Data Exchange Block: This block is primarily responsible for connecting to the MTQS network system at the units through the CAT5/CAT6 Ethernet standard, streaming data on the MTQS network at the unit through this block before entering the FPGA. Data to the Ethernet port in the form of Visai at the RJ45 port is converted into digital signals through the PHY, and FPGA controls this block through PHY control (as shown in Figure 2).

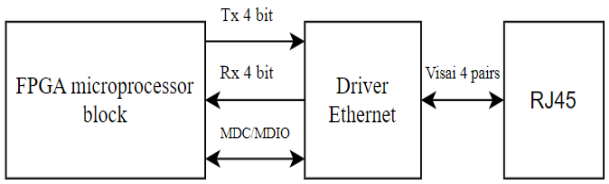

Figure 2. Ethernet Port Data Exchange Block

The Ethernet interface block utilizes the PHY chip RTL8211EG-VB-CG and Ethernet Connectors RJ45 JXD1-2016NL. The PHY communicates with the FPGA via the RGMII (Reduced Gigabit Media Independent Interface) standard. They provide all the necessary physical layer functions to transmit and receive Ethernet packets through CAT.5 UTP cables. The I/O voltage is fixed at 3.3V. The clock input of the PHY is provided by a 25MHz oscillator.

Optical Port Data Exchange Block: This block's main task is to connect to the MTQS network at the unit through the SFP/SFP+ optical standard, and data streams will pass through this block before entering the FPGA. Figure 3 shows the block diagram of Optical Port Data Exchange Module, this block includes: the Optical Port, the Optical Port Driver, and the Optical Transmission Protection Circuit. The data transmission process from the chip will process the signal data consisting of 3 components: transmitted data (GMII_TX_DATA), transmission enable signal (GMII_TX_EN), and error indication signal (GMII_TX_ERR). At the GMII Block, the data will undergo preprocessing and checking to ensure that the transmitted packet has the correct frame structure and complies with protocol standards. Then, the data will be sent to the internal data transmission block for encoding (TBI block). Here, the data will be enscoded as 8-bit to 10-bit for transmission (TXP/TXN). Additionally, the transmission core also includes a Clock block to synchronize signals and a general control block to manage the core components and configure registers in the SFP Module. The processing process is similar but in the reverse direction for chip input data.

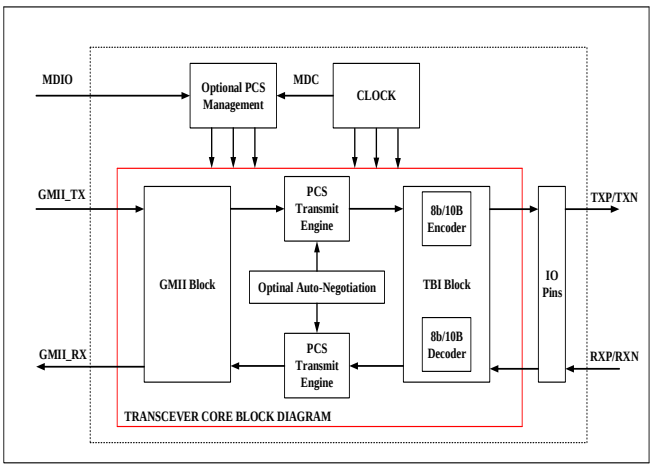

Figure 3. Block Diagram of Optical Port Data Exchange Module

The optical port interface block uses the SFP connector TXS0104EPWR. The optical interface block can implement high-speed data links as well as low-speed links such as optical/ethernet. SFP+ connectors are used to deploy two high-speed link streams that can operate up to 2.5Gbps each (in this case, 1Gbps is used). The clock input for the optical interface is provided by the high-speed clock block.

Power Supply Block: The power supply block uses dedicated power ICs to generate the necessary output voltages for the device such as 1.8V, 1.35V, 1.2V, 1V. This power IC integrates synchronous step-down converters and inductors to simplify the design, reduce external

components, and save board space (as shown in Figure 4).

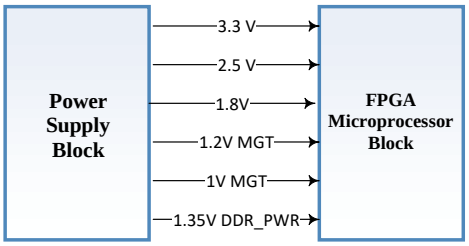

Figure 4. Power Supply Block

High-Speed Clock Block:

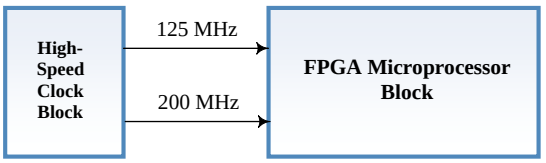

Figure 5. High-Speed Clock Block

The high-speed clock block utilizes the CDCM61002 clock synthesizer from Texas Instruments to provide the reference clock input for the optical port interface block. The output clock of the clock block can be adjusted to match the communication speed of the optical port. The maximum output clock is 625 MHz. In this case, we use output clock values of 125 MHz and 200 MHz (as shown in Figure 5).

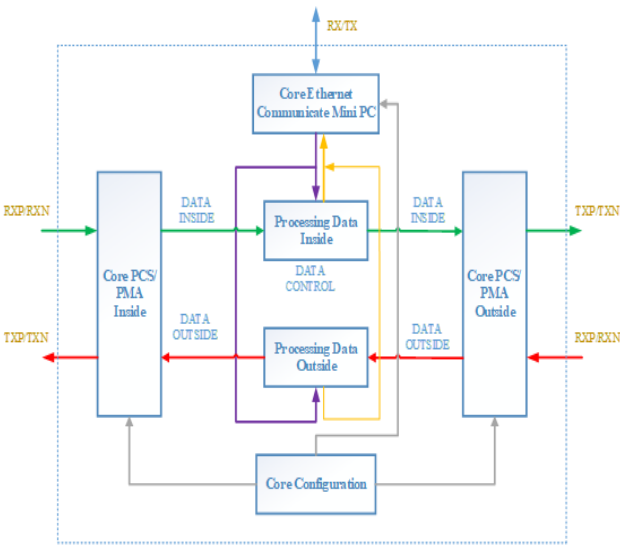

Figure 6. Block diagram of Principle Firmware Integration on FPGA Chip

Figure 6 shows the block diagram of Principle Firmware Integration on FPGA Chip. For each data frame after passing through the SFP/SFP+ port, analysis and processing steps will be performed as shown in Figure 7 below.

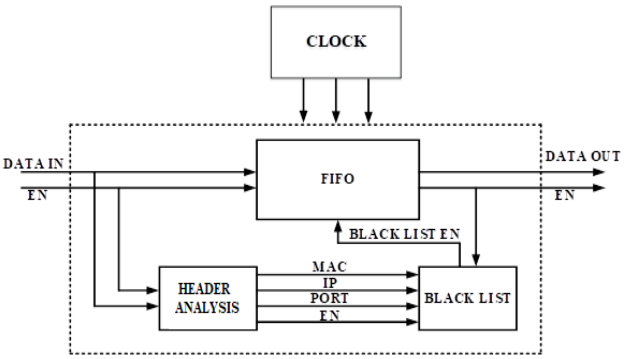

Figure 7. Input data processing

The data of a transmitted Frame is backed up into 2 lines, one to the core FIFO (First In First Out) to wait for the remaining checking process, the other to the core to separate the Header into small components (MAC address, IP, PORT, flags...). After separation is complete, those components are passed through the core Black-list to check. If the components match the "black" list, there will be a signal to delete the data of that transmitted Frame in the FIFO. On the contrary, there will be a signal allowing the Frame to pass until all data is transmitted in the FIFO, which will respond to interrupt the allowing-signal. So the testing process is performed continuously and simultaneously thanks to the Data Flow capability of the FPGA Board.

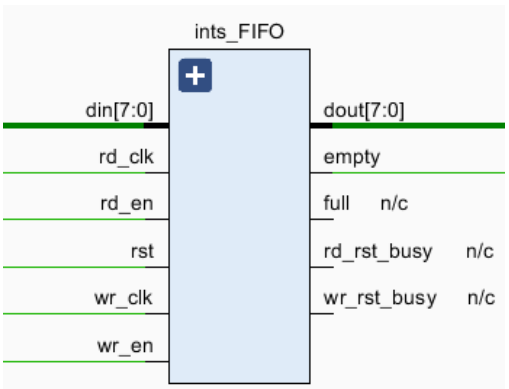

Figure 8. Pin-out core FIFO in Vivado.

To design a FIFO core, the research team considered that the memory components are located in a circle with two pointers write and read, data will be read and written in byte order. In addition, we use two additional error control flags: full (the write pointer has written data in one circle and meets the read pointer in the second circle) and empty (the read pointer state coincides with the write pointer when both pointers are in the same cycle. There is a read

signal for data that has not been written, resulting in data being lost) (as shown in Figure 8).

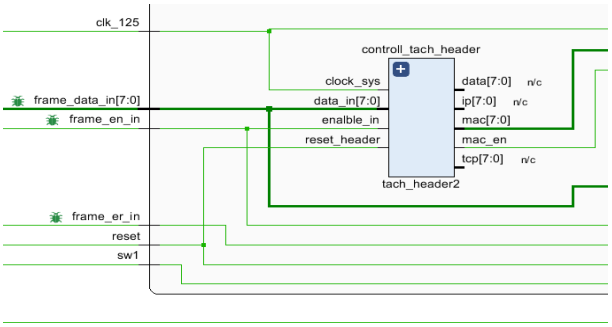

Figure 9. Pin-out core Header-analysis in Vivado

Figure 9 shows the pin-out of core Header-analysis in *Vivado software*. For this core, the research team takes full advantage of the FPGA's stream processing capabilities and the structure of each transmitted Frame to create separate subprograms that run in parallel at the same time. Specifically, for each transmitted Frame there are four main parts: preamble, header, data and checksum. Research team use the transmission enable-signal and the characteristics of the Preamble-part (six consecutive bytes "0x55") to identify the input of the transmission Frame. Then we separate each group and create processes that run concurrently, use counter variables to determine the data to be retrieved (MAC, IP, PORT, flag...) and then extract and package the output.

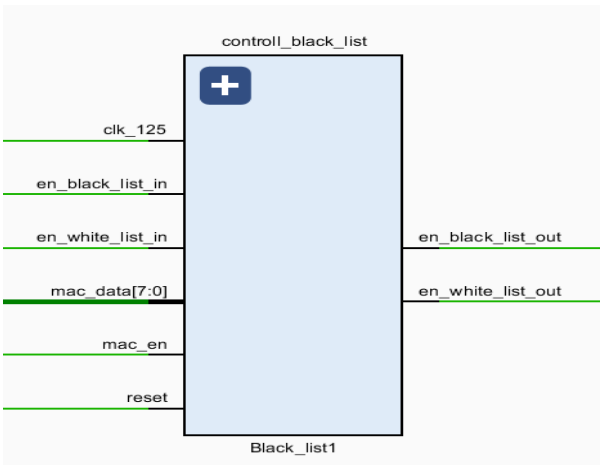

Figure 10. Pin-out core Black-list in Vivado.

In the core black-list (as shown in figure 10), the team will build a "black" list that needs to be prevented. To do that, the research team updates and saves this black-list to the external RAM of the FPGA board. With the FPGA's Data flow capability, we divide the black-list into small groups, each running on a different process. At the same time, the data extracted from the core Header-analysis will be checked in every process, that help the testing process many times faster (as shown in Figure 11). As long as the input data matches an element in the "black" list, the blocking-flag will immediately be transmitted to the FIFO core for processing and the packet will be discarded. Otherwise, the enable-flag will be transmitted to the FIFO core, however data will pass through.

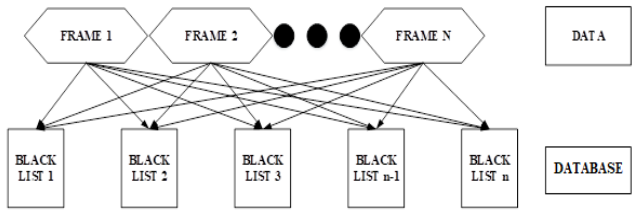

Figure 11. Method of splitting test-data and

The Xilinx XC7A200T chip was selected based on criteria of performance and cost. The chip is mounted on the designed board, and the research team integrated firmware using VHDL language, comprising the following components: Optical port interface module compliant with SFP/SFP+ standard; ethernet port interface module compliant with RJ45 standard connected to the control computer; digital processing module separating packet components; data processing module for Inside and Outside regions; rule execution module sent from the control computer. Figure 12 shows the Testing Device Model after realizing the idea.

Figure 12. Completed Testing Device Model

2. Software design

For the integrated control software on the embedded computer, based on the initial requirements, the research team prioritized choosing the Raspberry Pi 4 embedded computer

integrated with a program using Python language with the following functions:

- Collect, analyze, and standardize data from the FPGA board;

- Receive data and interact with the IDS monitoring system;

- Send block/unblock packet commands based on received rules to the FPGA board.

The deployment model features the capability to receive data from the IDS monitoring system. The IDS system will send a list of MAC addresses, IPs, or service ports along with command codes via an API. Upon receiving this data, the embedded computer will store it in a database while synchronously exchanging data back to the execution circuit to enforce rules based on the newly received data. The algorithm flowchart for receiving data from the monitoring system shown in figure 13.

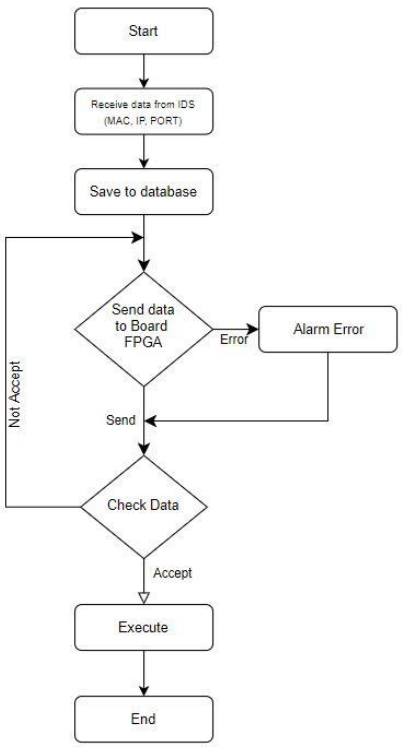

Figure 13. Algorithm Flowchart for Receiving Data from the Monitoring System

B. Experimental Deployment and Results

1. Implementation Plan

The research team collaborated with the network infrastructure management team to install devices in the unit's network infrastructure, worked with the network monitoring system development team to create and send control commands via API. We deployed a test model in an Inline configuration, after the firewall and before the Switch-Core (as shown in figure 14).

The testing device was connected with 3 network interfaces: 2 data flow connections labeled as Outside and Inside, and 1 specific Control connection:

- Outside Connection (SFP+ port): Connected to the output of the network firewall in the computer network at the deployment unit.

- Inside Connection (SFP+ port): Connected to the input of the Switch-Core network at the deployment unit.

- Control Connection (RJ45 port): Connected from the embedded control computer to the LAN network for communication with the IDS.

In case a computer or network device is detected with signs of insecurity, the IDS sends a disconnect command to the controlling embedded computer. The computer or network device will then be isolated from the network of the unit.

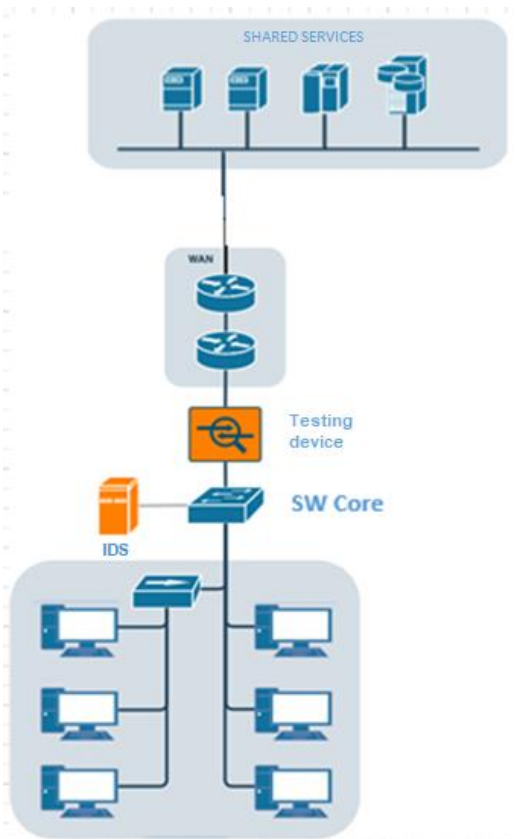

Figure 14. Test Device Deployment Diagram

2. Results Achieved

After configuring the FPGA chip to operate at a clock frequency of "125MHz", the research team conducted tests on several features including packet filtering and processing at the network layer; isolating user computers when violating rules; and blocking unwanted network intrusion activities. The image below depicts the process of frame data processing for transmission and reception in the *Vivado software*. The results of analyzing the MAC address of the packet on the FPGA are consistent with *Wireshark software* (as shown in figure 15 and figure 16).

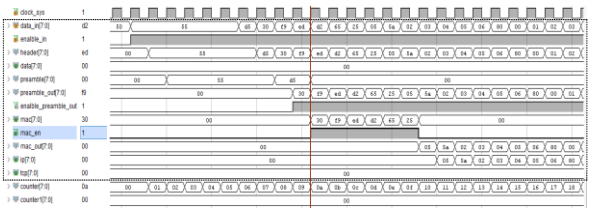

Figure 15. The process of analyzing frame data in Vivado.

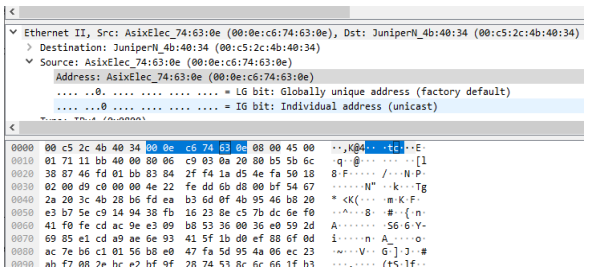

Figure 16. Frame data in Wireshark

The results displayed on the *Vivado software* processing screen show a data processing speed of "1 byte" in "8 nanoseconds," with an additional packet delay of "50 nanoseconds" compared to before the device was installed. The FPGA chip resources are utilized at "40 %". Figure 17 and figure 18 show the computer state before and after executing the command to disconnect from shared services.

```
C:\Users\ryaqu>ping fms.bqp

Pinging fms.bqp [86.2.120.162] with 32 bytes of data:
Reply from 86.2.120.162: bytes=32 time=5ms TTL=120
Reply from 86.2.120.162: bytes=32 time=11ms TTL=120
Reply from 86.2.120.162: bytes=32 time=9ms TTL=120

Ping statistics for 86.2.120.162:
    Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 5ms, Maximum = 11ms, Average = 8ms
Control-C
^C
C:\Users\ryaqu>
```

Figure 17. The computer before being blocked

```
Ping statistics for 86.2.120.162:
    Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 5ms, Maximum = 11ms, Average = 8ms
Control-C
^C
C:\Users\ryaqu>ping fms.bqp
Ping request could not find host fms.bqp. Please check the name and try again.

C:\Users\ryaqu>ping bt186.bqp
Ping request could not find host bt186.bqp. Please check the name and try again.

C:\Users\ryaqu>
```

Figure 18. The result after being blocked by testing device

The computers in the LAN infrastructure almost immediately lose connection to shared services after the blocking command is executed, which is sent from the IDS. It's evident that the processing speed has significantly improved during the firewall functionality simulation.

The research team used device TeraVM TVM5000 to measure some important performance parameters on the firewall - concurrent connection, established connection and throughput of the testing device when used in RAM-based rule set configuration mode. The achieved results show that the performance is much better than that of old generation firewalls that only use CPU. Figure 19 and figure 20 below are the results after measuring.

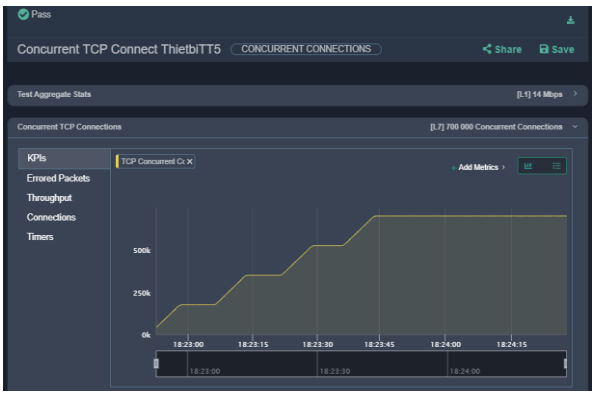

Figure 19. Concurrent connection measured on the testing device

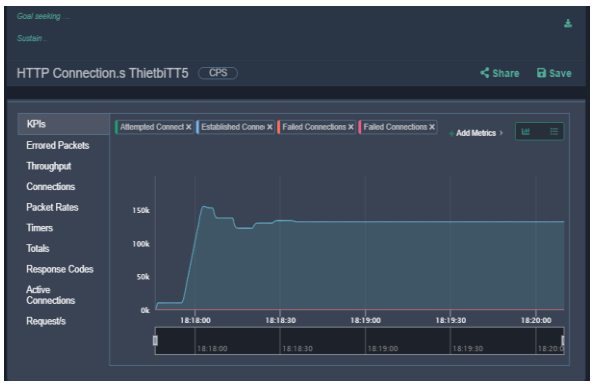

Figure 20. Established connection measured on the testing device

It is easy to see the superior processing performance of FPGA compared to using CPU or ASIC on older generation firewalls through the table below. Table comparing the performance parameters of the test device with old technology firewalls (ASA - uses CPU, Fortigate - uses ASIC) operating in transparent mode, in terms of functionality, it is completely similar to testing equipment.

TABLE I. COMPARISON OF PARAMETERS

	Concurent connection (connection)	Established connection (connection/s)	Throughput (Mbps)	Delay (µs)
Cisco ASA 5505	25000	4000	150	100-300
Cisco ASA 5510	50000	5000	300	150-500
Fortinet Fortigate 60D	100000	20000	1000	3 - 4
Testing device	700000	130000	1000	0,05 - 0,4

IV. CONCLUSION

In the increasingly complex landscape of network environments today, enhancing the data processing speed of firewalls poses a challenge for providers and system administrators alike. The article discussed one of the methods leveraging FPGA capabilities in firewall designs to increase processing speed, thereby minimizing network bandwidth consumption. The successful experimental deployment model on the circuit board demonstrates promising results. Looking ahead, this model could be further utilized for experimenting with integrated intelligent algorithms, behavior analysis based on data returned from packets (similar to IDS rules), maximizing the potential of FPGA chips.

The application of FPGA provides a flexible and powerful mechanism for optimizing data processing on network firewalls. By leveraging the reconfigurable nature of FPGA, we can optimize designs to meet specific performance and security requirements. This helps enhance the firewall's responsiveness to diverse and complex network threats. While deploying FPGA may require technical expertise and initial resource investment, the long-term benefits in terms of performance and flexibility are significant. For organizations seeking to enhance network security capabilities and optimize firewall system performance, using FPGA is a worthwhile consideration.

The article is a product of the research project funded by the Ministry of National Defense: "Research on Developing Intrusion Prevention Firewall Devices for Military Computer Networks".

ACKNOWLEDGMENT

Sincere thanks to the monitoring system development team in the military computer network and the infrastructure management team at Center 586 - Command 86 for their enthusiastic support during the research and deployment process, as well as the process of creating this paper.

REFERENCES

[1] Enis Karaarslan, Mohammed Babiker (2021). Digital Twin Security Threats and Countermeasures: An Introduction. IEEE.

[2] Y. Permpoontanalarp; C. Rujimethabhas (2002). A graph theoretic model for hardware-based firewalls. IEEE.

[3] Suchart Khummanee; Atipong Khumseela; Somnuk Puangpronpitag. (2013). Towards a new design of firewall: Anomaly elimination and fast verifying of firewall rules. IEEE.

[4] Fatih Ertam, Mustafa Kaya (2018). Classification of firewall log files with multiclass support vector machine. IEEE.

[5] Davison Zvabva, Pavol Zavarsky, Sergey Butakov, John Luswata (2018). Evaluation of Industrial Firewall Performance Issues in Automation and Control Networks. IEEE.

[6] Li, X., Zhang, Y., & Wang, J. (2017). "A New Method of Firewall Optimization Based on Machine Learning." In Proceedings of the 2017 IEEE 2nd Advanced Informatzon Technology, Electronic and Automation Control Conference (IAEAC) (pp. 1454-1458). IEEE.

[7] Khan, M. R., Zulkernine, M., & Al-Nashif, Y. (2020). "Network Security Monitoring and Attack Detection: Methods and Techniques." CRC Press. IEEE

[8] Andriotis, P., Korolov, A., & Mavroeidakos, T. (2018). "Firewall Optimization with Genetic Algorithm in Software-Defined Networks." In Proceedings of the 2018 International

Conference on Computer and Information Sciences (ICCIS) (pp. 1-5). IEEE.

[9] Shouyi Yin, Shibin Tang, Xinhan Lin, Peng Ouyang, Fengbin Tu (2019). A High Throughput Acceleration for Hybrid Neural Networks With Efficient Resource Management on FPGA. IEEE.

[10] Mohamed Khalil-Hani; Vishnu P. Nambiar; M. N. Marsono (2010). Hardware Acceleration of OpenSSL Cryptographic Functions for High-Performance Internet Security). IEEE.

[11] Tomoaki Sato, Sorawat Chivapreecha, Phichet Moungnoul, Kohji Higuchi (2017). An FPGA Architecture for ASIC-FPGA Co-design to Streamline Processing of IDSs. IEEE

[12] Raouf Ajami, Anh Dinh (2011). Design a hardware network firewall on FPGA. IEEE

[13] Phan, K.V., Tran, T.V. and La, P.H. 2020. A solution for packet security 1 Gbps on layer 2 with technology FPGA. Journal of Science and Technology on Information security. 8, 2 (Apr. 2020), 19-24. DOI: https://doi.org/10.54654/isj.v8i2.29.

ABOUT THE AUTHOR

Tran Ngoc Son

Workplace: Center 586, Command 86

Email: sontn.mta@gmail.com

Education: He received his BSc in Infonmation Techhnology from Le Quy Don Technical University in 2007; MSc in Computer Science from Le Quy Don Technical University in 2012.

Recent research direction: Research information security solutions and applications for IT and IoT network infrastructure.

Tên tác giả: **Trần Ngọc Sơn**

Cơ quan công tác: Trung tâm 586, Bộ Tư Lệnh 86

Email: sontn.mta@gmail.com

Quá trình đào tạo: Nhận bằng kỹ sư Đại học ngành công nghệ thông tin tại Đại học Kỹ thuật Lê Quý Đôn năm 2007; Thạc sỹ ngành khoa học máy tính tại Đại học Kỹ thuật Lê Quý Đôn năn 2012.

Hướng nghiên cứu hiện nay: Nghiên cứu giải pháp, ứng dụng an toàn thông tin cho hạ tầng mạng IT, IoT.

Nguyen Van Manh

Workplace: Center 586, Command 86

Email: manhng.it@gmail.com

Education: He received his BSc in Infonmation Techhnology from Le Quy Don Technical University in 2010; MSc in Information Security from Academy of Cryptography Techniques in 2023.

Recent research direction: Research information security solutions and applications for IT and IoT network infrastructure.

Tên tác giả: **Nguyễn Văn Mạnh**

Cơ quan công tác: Trung tâm 586 – Bộ Tư Lệnh 86

Email: manhng.it@gmail.com

Quá trình đào tạo: Nhận bằng kỹ sư Đại học ngành công nghệ thông tin tại Đại học Kỹ thuật Lê Quý Đôn năm 2010; Thạc sỹ ngành an toàn thông tin tại Học viện Kỹ thuật mật mã năm 2023.

Hướng nghiên cứu hiện nay: Nghiên cứu giải pháp, ứng dụng an toàn thông tin cho hạ tầng mạng IT, IoT.

Dinh Viet Phuong

Workplace: Center 586, Command 86

Email: dinhphuong49@gmail.com

Education: He received his BSc in 2018; MSc in Information Security from Belarusian State University of Informatics and Radioelectronics in 2020.

Recent research direction: Research information security solutions and applications for IT and IoT network infrastructure.

Tên tác giả: **Đinh Việt Phương**

Cơ quan công tác: Trung tâm 586, Bộ Tư Lệnh 86

Email: dinhphuong49@gmail.com

Quá trình đào tạo: Nhận bằng cử nhân ngành bảo mật thông tin tại Đại học tin học và Vô tuyến điện tử Belarus năm 2018; Thạc sỹ ngành bảo mật thông tin tại Đại học tin học và Vô tuyến điện tử Belarus năm 2020.

Hướng nghiên cứu hiện nay: Nghiên cứu giải pháp, ứng dụng an toàn thông tin cho hạ tầng mạng IT, IoT.