Network Intrusion Detection Using Hardware Techniques: A Review

Razan Abdulhammed, Miad Faezipour and Khaled M. Elleithy
Computer Science and Engineering Department
University of Bridgeport, Bridgeport, CT
rabdulha@my.bridgeport.edu, mfaezipo@bridgeport.edu, elleithy@bridgeport.edu

***Abstract* — The increasing amount of network throughput and security threat makes intrusion detection a major research problem. In the literature, intrusion detection has been approached by either a hardware or software technique. This paper reviews and compares hardware based techniques that are commonly used in intrusion detection systems with a special emphasis on modern hardware platforms such as FPGA, GPU, many-core processors and ASIC. It also provides a detailed comparison between these hardware solution platforms. Our approach to classify modern hardware-based Intrusion Detection System (IDS) techniques is based on the detection approach. In addition, we provide a comparison between the classified detection approaches based on essential criteria such as definition, update process, detection ability, features of the system, and implementation requirements. Finally, a classification tree of hardware-based NIDS platforms is given.**

***Index Terms* — Intrusion detection system, FPGA, GPU, NFA, DFA, Pattern matching, TCAM, ASIC, Many-Core Processors.**

I. Introduction

The concept of Intrusion Detection System (IDS) was first described in the early eighties [1]. Intrusion detection system is a security measure that helps to identify a set of malicious actions that compromise the integrity, confidentiality, and availability of information resources [2]. From a general prospective, IDSs can be divided based on the detection approach into two categories: anomaly, and signature based detection. The fundamental concept in an anomaly- based detection method is to define the behavior of the targeted network, and then this predefined behavior is compared to the normal actual network behavior. Anomaly based detection is used to detect both known and unknown attacks. In contrast, signature based systems are used to detect known attacks and require prior knowledge of attack signatures. Table I shows a comparison between these approaches.

The IDS gathers, observes and collects data before the analysis phase using different strategies. These include: host-based, network-based, and hybrid-based approaches. The analyzing process takes place either while the sessions are ongoing, yielding a real time IDS, or after the information data has been already collected, yielding an offline IDS. The offline type of IDS is easy to implement compared to the real time one. However; the real time IDS is useful for understanding the attacker's behavior. The performance of an intrusion detection system depends on two metrics: throughput and the total number of patterns that can fit on a device.

A Network Intrusion Detection System (NIDS) requires pattern matching, string matching, multimatch packet classification, and a regular expression to perform its functions. In spite of the great progress that has been made, there are several factors that contribute to increasing the gap between the processing requirements of IDS and their software implementations. Such as the increase in network throughput, the increasing number and sophistication of attacks, and the performance limitations of sequential software execution. Within these factors, developers consider to build IDS by using hardware implementation techniques to secure network infrastructure. The utmost reason for shifting from software to hardware is to enable real-time implementation of IDS. The major performance difference between hardware and software solutions lies in their dissimilar execution paradigms. Essentially, hardware involves implementations that allow execution of many operations in parallel, while the execution of pure software implementation is serial or with limited parallelism [3]. As a matter of fact, the design boundary between hardware and software is illusory at this point [4], [5].

The primary goal of this paper is to investigate the common hardware-based techniques in the literature to build and implement NIDS functions. The organization of the rest of the paper is as follow: hardware based techniques for packet classification is described in the Section II as a base to understand the methods. Hardware based techniques for string and pattern matching are reviewed in the third part, and hardware based techniques for regular expression are discussed in part 4. A sample of the literature of hardware-based Intrusion detection system is given in part 5. A brief conclusion is in the final part of the paper.

II. Hardware-Based Techniques For Packet Classification

A search of the literature revealed a few studies which provide various multi-match packet classification solutions. To start with, Song et al. [6] presented BV-

TABLE I. A COMPARISON BETWEEN IDS DETECTION APPROACHES.

Criteria	Anomaly-Based	Signature-Based
Update	No	Yes
Detection ability	Both Known and unknown attacks	Only Known attacks with very high accuracy
Definition	Use deviation from normal usage pattern to identify intrusions	Use patterns of well-known attacks to identify intrusions
Feature of the system	High False Alarm	Low False Alarm
Implementation requirement	Requires less computation and resources	Requires more computation and resources

TCAM that is a packet classification architecture based on FPGA for network intrusion detection. The BV-TCAM architecture uses a combination of both Ternary Content Addressable Memory (TCAM) and the Bit Vector (BV) algorithm. This architecture is able to eliminate the requirement for the prefix expansion of the port ranges through TCAM and a multi-bit trie algorithm that is implemented in a tree bitmap to be used as the source or destination port lookup. The architecture gives an acceptable performance. However, the BV-TCAM algorithm suffers from the $O(N^2)$ memory requirement.

Likewise, Fang Yu et al. [7] developed a new algorithm to split the rule set into two groups in order to perform separate TCAM lookups among the rules. The new algorithm that is called Set Splitting Algorithm (SSA) was able to meet both requirements of low TCAM memory usage and low-power consumption comparing to the BV-TCAM. The rules set that were used to verify the algorithm were SNORT rules. Moreover, SSA can be combined with SRAM/TCAM hybrid approaches for further energy consumption reduction. However, the worst case memory requirement is still $O(N^2)$, which means that this approach is expensive regarding power and memory consumption when the rules set has many intersections. By the same token, a TCAM-based multi-match packet classification partitioning approach was proposed in [8]. This approach was called a Maximum-Minimum Insertion Partitioning approach (MX-MN-IP). The MX-MN-IP partitions the entire rule set into several disjointed partitions. It helped not only to reduce power consumption but also allows only one portion active when classifying a packet. Furthermore, each partition is further partitioned so that each sub partition can output one matching result in only one clock cycle. One of the drawbacks of this approach is that it may need a large number of small TCAMs and as a result the actual implementation becomes impractical. Moreover, the number of partitions depends on the characteristic of the rules set which will vary the power reduction ratio.

Jiang et al. [9] proposed an SRAM-based architecture which was called field-split parallel bit vector (FSBV). In this architecture, the authors make a key observation in the original BV algorithm where some header fields of a packet are further split into bit-level subfields. To clarify, a W-bit field is split into w subfields in which each subfield takes only 1 bit, and the number of unique values in each subfield will be no more than 2. The result was able to reduce the memory requirement compared to the original BV algorithm in which the memory requirement of FSBV is a linear increase of the number of rules assuming fixed number of W. The splitting process of a field through FSBV architecture was efficiently implemented in an FPGA. The FSBV memory requirement is $O(wN)$ where w is the number of subfields and N is number of bit vectors in each field. Table II shows a comparison of the hardware-based packet classification NIDS techniques.

TABLE II. COMPARISON OF THE PERFORMANCE OF THE STATE OF THE ART HARDWARE SOLUTIONS FOR MULTI-MATCH PACKET CLASSIFICATION IN NIDS.

Authors	Algorithm	Platform	Through-put	Storage	Power	Support SNORT Feature
Song	BV-TCAM	FPGA	10	73.8	17.7	No
Fang	TCAM-SSA	TCAM	20	13	312	No
Faezipour	MX-MN-IP	TCAM	80	13	296	No

III. HARDWARE TECHNIQUES FOR STRING AND PATTERN MATCHING

String and pattern matching-based techniques are commonly used in network based intrusion detection systems in which the attack patterns are modeled, matched and identified based on the packet header, packet content or both. The NIDS has to scan the incoming traffic in real time and on fast link. Much work has been done in this field and the state of the art solutions in the literature for string and pattern matching that have been used in IDS can be generally divided based on the implementation platform into these categories: multi core processor-based, Specific Integrated Circuit-based, Field Programmable Gate Array- Based, TCAM, and pipelined Non-Deterministic Finite Automaton (NFA), and Pipelined Deterministic Finite Automaton (DFA).

The most common method to implement pattern matching in hardware is to use Finite Automata (FA) approaches. The approaches of Finite Automata (FA) are either Deterministic Finite Automata (DFA) or non-deterministic finite automata (NFA). Each of these approaches has its own advantages and drawbacks. The DFA based pattern approach is fast enough but requires more memory. In contrast, the NFA based pattern matching approach comparatively takes less memory but the speed of matching is very slow. Many approaches have been proposed to overcome the drawbacks of both types of the finite automata. A good comparative study of some Finite Automata (FA) based techniques for pattern matching in network intrusion detection system (NIDS) is reported by the study of Rathod et al. [10].

Due to the huge growth of network traffic, multi-pattern and string matching has been a major performance bottleneck in network intrusion detection systems and what follows presents a sample of the literature.

Scarpazza et al. [11], parallelized Aho-Corasick string searching algorithm using IBM Cell/B.E. processor as the hardware platform .The goal was to perform exact string matching against large dictionaries. Since memory

congestion plays a crucial role in determining the performance of the original algorithm, thus, the authors proposed a new approach to fully exploit the DMA-based communication mechanisms of the Cell/B.E. in order to provide an unprecedented level of aggregate performance with irregular access patterns.

Gnort was reported as the first systematic study of using GPU to build IDS in 2008 [12]. In this study, the authors used SNORT open-source NIDS in order to exploit the underutilized computational power of modern graphics cards, specifically the NVIDIA GeForce 8 Series (G80) cards using CUDA. The goal was to offload the costly pattern matching operations from the CPU, and hence increase the overall processing throughput. This study showed that modern graphics cards can be used effectively to speed up intrusion detection systems, as well as other systems that involve pattern matching operations.

Tian Song et al. [13] presented the ACC multiple pattern matching algorithm; the acronym ACC stands for Aho-Corsaic with counters. The presented algorithm is based on a Cached Deterministic Finite Automata (CDFA) that extends a DFA through associating some memory as cache and uses a next-state addressing (NSA) scheme to store and access transition rules in memory. The authors use several optimizations methods such as set-associative policy, fine grain multithreading, and entry combination to overcome the problems associated with this kind of implementation such as the shortened critical path and the memory utilization efficiency. The CDFA model can be used for multiple regular expression matching with an acceptable performance and efficient memory utilization.

Jung et al. [14] implemented a bit-split string-matching architecture on FPGA towards addressing memory efficiency, pin count, and optimizing the number of keywords per rule module. One of the characteristics of the bit-split algorithm is that it allows large hardware state machines to be converted into a form with much higher memory efficiency [15]. This was done through using a new architecture that utilizes a priority encoder to reduce the number of external IO pins. The results showed that the architecture can be effectively optimized for FPGA implementation.

Lunteren [16] presented BFSM, hardware-based scheme for pattern-matching. The presented scheme exploits a hardware-based programmable state machine technology in order to achieve deterministic processing rates that are independent of both input and pattern characteristics. Furthermore, BFSM can exploit hardware parallelism. The implemented algorithm was able to achieve processing speed in the order of 10 Gb/s for FPGA and at least 20 Gb/s for ASIC.

E. Yang et al. [18] proposed pipelined field-merge architecture to enhance the bit-split approach [14] for Large-Scale String Matching (LSSM) that is memory-efficient and yieds high-throughputs. The fundamental concept of the pipelined architecture is to partition the character input (8-bit) into several bit-field inputs of smaller widths (usually 2-bits), followed by a matching process that is done for each bit-field input in a partial state machine (PSM) pipeline. The (PSM) is constructed from the respective bit-field patterns. Then, in every pipeline stage a matching process with the help of an auxiliary table (ATB) is used for the matching results from all the bit-fields.

One of the problems associated with AC-DFA (Aho-Corsaic with Deterministic Finite Automaton) is that it requires a large number of state transitions. This problem inhibits efficient hardware implementation to achieve high performance. Some developer's recent work [18], [19] have shown that the AC-DFA can be reduced to a character trier that contains only the forward transitions through incorporating pipelined processing. However, they have limitations in either handling long patterns or extensions to support multi-character input per clock cycle to achieve high throughput.

Weirong Jiang et al. [20] generalized the previous mentioned problem and provided a better solution by presenting a scalable pipeline architecture. The architecture can be easily extended to support multi-character inputs per clock cycle through mapping a compressed AC-DFA [21] onto multiple pipelines.

Hoang Le et al. [22] proposed the "leaf-attaching" approach to preprocess a given dictionary such that the resulting set of the post-processed patterns can be searched using any tree-search data structure without increasing the number of patterns. Furthermore, the work proposed an architecture based on a pipelined binary search tree for large-scale string matching (LSSM). The new architecture was scalable in such a way that the MASM module can be duplicated to accept multiple characters per cycle. One of the advantages of this work is that the update process of the dictionary involves simply rewriting the content of the memory, which can be done without reconfiguring the FPGA chip. A study carried out by Gharae [23] provided a deep insight on the pattern matching algorithms that are used in intrusions detection systems. Table III provides a comparison of different hardware-based string matching techniques.

TABLE III. COMPARISON OF VARIOUS HARDWARE-BASED STRING MATCHING PERFORMANCE SOLUTIONS

Authors	Approach	Platform	Number of patterns	Pattern Length	Throughput
Jung	Bit-Split	FPGA	1316	No limit	1.76
Vasiliadies	AC-DFA	GeForce 8600GT	4000	<25	2.3
Lunteren	B_FSM	FPGA	8000	No limit	2.2
Scarpazza	AC-DFA	Cell/B.E	8400	<10	2.5
E Yang	Field Merge	FPGA	6944	<64	4.56
Song	CDFA	ACIS	1785	No limit	6.1
Jiang	Depth-Bounded Pipeline	FPGA	9033	No limit	11.4
Hoang Le	leaf-attaching	FPGA	10856	<64	11.8

IV. Hardware-Based Techniques for Regular Expression

Regular expression (regex) matching is an important mechanism used by NIDS to perform deep packet inspection against potential threats. Regex matching is becoming both a bottleneck and a vulnerability of the NIDS because both the increasing bandwidth of network traffic and the huge number of patterns that need to be scanned by the NIDS. A basic regular expression matching engine can be implemented in hardware as a state machine, either a deterministic finite machine or a non-deterministic finite machine [24].

Recently, a considerable literature has grown up around the theme of regular expression matching; a sample of which is provided in the following paragraphs.

Yamagaki et al. [25] presented a high throughput approach for matching regular expressions. The presented approach was capable of converting the regular expressions into a multi-character Non-deterministic Finite Automata (NFA) that is able to be processed as multiple characters per clock cycle. The work was implemented and configured onto an FPGA platform. The result showed that the work can implement the range match operation efficiently.

Clark et al. [26] presented a scalable design methodology for searching network packet payloads for complex regular expressions. The basic idea is to use a multi-character decoder based on the NFA technique that produced high-performance circuits over a wide range of pattern set sizes. The result showed that this methodology was able to offer flexible trade-offs among throughput, character capacity, and data bus rate.

Sourdis et al. [27] described a regular expression pattern matching approach for reconfigurable hardware based on the NFA direction. The proposed approach introduced three new basic building blocks in order to support more efficiently the constraint number of repetition syntaxes. Optimization techniques were employed to maximize the performance and reduce the area cost of the designs. Table IV provides a comparison of a few hardware-based regular expression matching engines.

Table IV. Comparison of the most common hardware-based Regular expression matching engine performance solutions

Authors	Approach	Non-Meta Chars	Number of LUT per state	Multi-Char Per Cycle	Throughput
Clark et al.	FPGA	17,537	3.1	4	73.8
Sourdis et al	FPGA	69,127	0.66	1	2.42
Yamagaki et al	FPGA	40,896	0.94	4	3.63
Bispo et al.	FPGA	19,580	1.28	1	2.9

V. Hardware-Based Intrusion Detection Systems

A hardware-based intrusion detection system is a scalable method as it is able to inspect packets in high speed networks. We have taken a glance at earlier hardware techniques platforms; the general purpose processor such as Central Processing Unit (CPU), Graphical Processing Units (GPU), network processors, multi core processors, Application Specific Integrated Circuit (ASIC), Field Programmable Gate Arrays (FPGA), and Ternary Content Addressable Memory (TCAM). The processors are highly programmable, and as a result of the Von Neumann architecture, the bandwidth between the processor and the external memory is the limit. Therefore, their performance will be less effective for network processing. In contrast, ASICs are able to provide better performances; however, they cannot be reconfigurable. On the other hand, FPGAs which are reconfigurable hardware platforms, offer better performance than ASIC and processors. Most developers use FPGA as a preferred hardware platform for their superior performance and reprogramming applicability. The reported research work in this area has been well summarized by Chen et al. [5]. Ternary Content Addressable Memory (TCAM) [6, 28, 29] is a special ASIC that is used by developers in network search engines and for multi-match packet classification engines. It has been adopted to solve the multimatch classification problem because of their ability to perform fast parallel matching. However, TCAMs are expensive, not scalable with respect to clock rate, have high power consumptions, and circuit area. In the recent years, there has been an increasing amount of literature on packet classification. Taylor [30] provided an excellent survey of the seminal and recent solutions to the problem. The TCAM-based approach is expensive with respect to memory and power consumption when the rules set has many insertions [5]. Furthermore, researchers have attempted to reduce the power consumption in TCAM-based approaches [29]. Graphics Processor Units have been used to step-up computationally intensive tasks in intrusion detection systems. For example, Vasilios et al. reported the first systematic study of using GPU in 2008 [31]. The architecture of these hardware solutions has its own advantages and disadvantages. Table V shows a comparison among hardware platform solutions.

The performance of a hardware implementation will, in general, exceed that of software, and there are still many constraints that must be taken into account when designing an IDS based on hardware detection approaches because these hardware solutions are complex to customize and are usually tied to a definitive application, making it difficult to update the design [32].

Table V. Comparison among hardware platform solutions

Multi-Core Processors	ASIC	FPGA
Can enhance the aggregate throughput dramatically by using a large number of threads to process multiple input stream in parallel	Provide impressively high per-stream throughput	Provide desirable high performance ,flexibility of software and reconfigurable programming
Additional Complexity is introduced in scheduling ,buffering ordering, and load balancing	The applicability is limited by the high implementation cost and low reprogrammability	It takes considerable time to resynthesize the design and reprogram the FPGA device

VI. Classification Tree of the Survey

We summarize our findings in Table VI. Despite our best attempts, these tables do not take in all available literature. We listed the IDS platform in column 2. The audit features column provides a description of what features a system is working on. The analysis time column indicates whether the IDS is a real time or offline based IDS. The audit material specifies the type of the data collecting mechanism.

In [33], the authors developed a different, traffic-aware, modular approach in the design of an FPGA based NIDS. The approach classifies and groups homogeneous traffic and dispatches it to different capable hardware blocks, each supporting a (smaller) rule set tailored to the specific traffic category. The SNORTrules have been split into different subdivisions based on protocol, port, and the direction of the traffic that helps support the dispatcher. The basic implementation of the string matching circuit in this workflow is the basic architecture that is considered in the shift-and-compare architecture presented in [34]. The IDS was implemented on the INVEA COMBO-LXT, an express PCI x8 mother card equipped with the XILINX Virtex5 XC5VLX155T, two QDR RAM memories, and up to 4 GB of DDR2 memory. The authors stated that their approach produced an 80% reduction in the number of Lookup tables (LUTs).

In the same fashion, the authors of [35] implemented a real-time intrusion detection system prototype that uses a high frequency sampling rate enabled field-programmable gate array (FPGA). The FPGA is equipped with a software-driven engine. The software-driven engine is implemented based on an internally evolvable improved Block Based Neural Network (BBNN), which is an intelligent learning algorithm, to perform the attack related pattern recognition. This combination provides the flexibility to learn and detect unknown attacks. The experiment compared the improved system with four major schemes of Support Vector Machine (SVM) and the Naive Bayes algorithm. The authors used flow records that are extracted from a DARPA packet based dataset after converting it into a NetFlow format because the DAPRA data set is in the form of tcpdump format. The tcpdump is a common packet analyzer format that runs under the command line to allow the user to display TCP/IP and other packets being transmitted or received over a network to which the computer is attached. The results indicated that the improved BBNN outperforms other algorithms with respect to the classification and detection performances. The system false alarm rate is successfully reduced to as low as 5.14% with a detection rate of 99.92%. This work uses a large-scale Altera Cyclone III FPGA Starter Board with Universal Synchronous Bus (USB) interface. It is noticeable that SVM-based IDS can classify only discrete features. Thus, preprocessing of those characteristics is required. Similarly, cited work [36] examined a hybrid network intrusion detection system (SFAOENIDS) for Solarflare AOE devices using Inline FPGA with a network adapter to provide hardware support for pattern matching and software support for post processing. The system was implemented using a hardware based description language. The SFAOENIDS was experimentally tested on a real network environment operating at 200 MHz, handled a 10Gbps data rate without dropping packets, while simultaneously minimizing the server CPU load. SFAOENIDS performance was tested and compared against SNORT. Since SFAOENIDS missed the TCP stream reassembly features, these were disabled in SNORT. The CPU-only based SNORT implementation dropped 90% of packets at speeds near 10 Gbps. In contrast, SFAOENIDS dropped zero packets at these speeds.

Equally important, the authors of cited work [37] proposed the anomaly based IDS using FPGA architecture. At first, a feature extractor module (FEM) was developed. By using an FPGA, significant performance was achieved compared to software solutions like SNORT. It was shown in that work that around 22 Gbps throughput can be achieved. Their proposed IDS also used Principal Component Analysis (PCA) to reduce data dimensionality. The results of their analysis produced acceptable accuracy scores on the KDDCup dataset. The FPGA used in their work was the Xilinx Vertex Family.

Different from cited work [37], the authors of [38] considered a support vector machine intrusion detection method that used a Graphical Processing Unit (GPU)–based parallel computing model. The work used the KDDCup1999 dataset to implement the proposed model. The simulation result showed that the implemented system was able to reduce the time consumption in the training procedure, while keeping the performance of IDS as usual. This work was based on the anomaly detection method.

By the same token, cited work [39] studies a layered, simple statistics anomaly based network intrusion detection system. The study used a GPU with parallel implementation to describe the training algorithm. The work was done in two stages: training stage and decision stage. In the training stage, similar operations were performed for all of the packets residing in the training set. These operations included threshold setting, bin statistics computation, packet scoring, and bin counting. In the detection stage, computation functions that included bin counting and packet scoring were performed. The study tested the performance of the parallel implementation of the system using Compute Unified Device Architecture (CUDA) framework with the Nvidia GTX 780 model and by a CPU on a machine with AMD FX-8350, 16GB RAM, and Ubuntu Linux 12.04.2.

TABLE VI: CLASSIFICATION OF NIDS BASED ON HARDWARE SOLUTION PLATFORMS.

Author	Plat form	Analysis time	Audit material	Audit Features
Pontarelli	FPGA	Real time	Network-Based	UDP Packets,TCP Packets
Quang	FPGA	Real time	Network-Based	IP , Port ,Packets, Octets, Start Time, End Time, Flags
Jaic	FPGA	Real time	Network-Based	Packet payload
Xia	GPU	Offline	Network-Based	TCP/IP packet header features
Kim	GPU	Offline	Network-Based	Packet payloads
Jamshed	GPU	Real time	Host-Based	Packet payloads
Jaehyun	MultiCore Processors	Offline	Network-Based	Packet payloads
Das	FPGA	Real time	Network-Based	Packet payload

The training set size ranged from 10k packets to 500k packets. The packets have been captured from a real web server.

In the light of GPU, cited work [40] designed highly scalable software IDS architecture (Kargus) that used a GPU, multiple CPU cores, non-uniform memory access (NUMA) with symmetric architecture per each NUMA domain, and multiqueue 10 Gbps network interface cards (NICs). The design used two techniques to achieve high performance, including batch processing and parallel execution with an intelligent algorithm for load balancing across CPU and GPU. The designated system applied batch processing from packet reception, flow management, all the way to pattern matching. Kargus is functionally compatible with SNORT and its configuration files. The single-process, multi-thread architecture of Kargus consists of a dual-NUMA machine, two hexacore CPUs, and two GPUs.

On the other hand, the authors in [41] designed a highly scalable network intrusion detection system using system-on-chip many-core processors (MCPs). The design used TILE-Gx72 many-core processor and the authors were able to maximize the NIDS performance using different design principles such as computation offloading, lightweight data structure, flow offloading, and shared-nothing architecture. In shared-nothing, each thread is running a separate NIDS engine without any shared data structure between the NIDS and pinned to a tile. In computation offloading, the system employed multicore Programmable Intelligent Packet Engine (mPIPE) packet I/O engine in a TILE-Gx72 processor to offload per-packet. Partitioning was applied to divide the data structures into two groups of frequently used fields and rarely used fields, in which he latter was extracted from the data structure. To apply flow offloading, the authors exploited TRIO module, a hardware module that performs bidirectional PCIe transactions, in a TILEGx72 processor. The TRIO module is used to dynamically offload subsequent flows to host-side CPU or analysis, when the many-core processor faces a high workload. The experimental results showed that the proposed design significantly improves the NIDS performance such that the system was able to process 79 Gbps with 1514B synthetic packets.

VII. CONCLUSION

The application of hardware for intrusion detection systems has become an active area of research with the support of the modern powerful hardware devices such as FPGA, ASIC, Multicore processor and GPU. However, hardware-based techniques might face real performance challenges due to multiple factors that may affect the implementations of these techniques such as the fiercely rising demands to handle higher traffic rates, more intensive analysis, real-time operation, power consumption, memory requirements and the collapse of Moore's law for sequential processing.

REFERENCES

[1] M. Whitman, and H. Mattord, Principles of information security. 4th Ed, Boston: Cengage Learning. 2011

[2] R. Bhatnagar, and U. Shankar, "The proposal of hybrid intrusion detection for defense of sync flood attack in wireless sensor network," Int. J. Comput.Sci and Eng. Surveys, vol. 3, no 2, pp 31-38, Apr.2012.

[3] D. E. Culler, A. Gupta, and J. P. Singh, Parallel Computer Architecture: A Hardware/Software Approach. Morgan Kaufmann Publishers Inc., 1997

[4] P. Hunter, "Hardware-based security: FPGA-based devices," Computer Fraud & Security, vol. 2004, no. 2, pp. 11–12, 2004

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

[6] Song, Haoyu, and John W. Lockwood. "Efficient packet classification for network intrusion detection using FPGA." Proceedings of the 2005 ACM/SIGDA 13th international symposium on Field-programmable gate arrays. ACM, 2005.

[7] Yu, Fang, et al. "Efficient multimatch packet classification for network security applications." Selected Areas in Communications, IEEE Journal on24.10 (2006): 1805-1816.

[8] M. Faezipour, M. Nourani, "Wire-Speed TCAM-Based Architectures for Multimatch Packet Classification," IEEE Trans. Comput., vol.58, no.1, pp.5-17, Jan. 2009

[9] Jiang, Weirong, and Viktor K. Prasanna. "Field-split parallel architecture for high performance multi-match packet classification using FPGAs."Proceedings of the twenty-first annual symposium on Parallelism in algorithms and architectures. ACM, 2009.

[10] Z.K. Baker, and V.K. Prasanna, "Automatic Synthesis of Efficient Intrusion Detection Systems on FPGAs", IEEE Trans.on Dependable and Secure Computing, vol.3, no.4, pp.289-300, Oct.-Dec. 2006

[11] Scarpazza, Daniele Paolo, Oreste Villa, and Fabrizio Petrini. "High-speed string searching against large dictionaries on the Cell/BE processor." Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on. IEEE, 2008.

[12] G. Vasilios, S. Antonatos, M. Polychronakis, E. P.Markatos, and S. Ioannidis, "Gnort: High Performance Network Intrusion Detection Using Graphics Processors" In Proc. of the 11th International Symposium on Recent Advances in Intrusion Detection (RAID'08), 2008

[13] T. Song, W. Zhang, D. Wang and Y. Xue, "A memory efficient multiple pattern matching architecture for network security." INFOCOM 2008. The 27th Conference on Computer Communications. IEEE. IEEE, 2008.

[14] Jung, Hong-Jip, Zachary K. Baker, and Viktor K. Prasanna. "Performance of FPGA implementation of bit-split architecture for intrusion detection systems." Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International. IEEE, 2006.

[15] Van Lunteren, Jan. "High-Performance Pattern-Matching for Intrusion Detection." Infocom. Vol. 6. 2006.

[16] Yang, Yi-Hua E., and Viktor K. Prasanna. "Memory-efficient pipelined architecture for large-scale string matching." Field Programmable Custom Computing Machines, 2009. FCCM'09. 17th IEEE Symposium on. IEEE, 2009

[17] D. Pao, W. Lin, and B. Liu, "Pipelined architecture for multistring matching," Computer Architecture Letters, vol. 7, no. 2, pp. 33–36, Feb. 2008.

[18] Y.-H. E. Yang and V. K. Prasanna, "Memory-efficient pipelined architecture for large-scale string matching," in FCCM 2009. 17th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, April 2009.

[19] Jiang, Weirong, Yi-Hua E. Yang, and Viktor K. Prasanna. "Scalable multi-pipeline architecture for high performance multi-pattern string matching."Parallel & Distributed Processing (IPDPS), 2010 IEEE International Symposium on. IEEE, 2010.

[20] M. Alicherry, M. Muthuprasanna, and V. Kumar, "High speed pattern matching for network IDS/IPS," in ICNP '06: Proceedings of the 2006 IEEE International Conference on Network Protocols. IEEE Computer Society, 2006, pp. 187–196.

[21] Le, Hoang, and Viktor K. Prasanna. "A memory-efficient and modular approach for large-scale string pattern matching." Computers, IEEE Transactions on 62.5 (2013): 844-857.

[22] A. S. K. Pathan, "The State of the Art in Intrusion Prevention and Detection", 1st Ed, New York, CRC Press, 2014

[23] P.M., Rathod, N. Marathe, and A.V Vidhate, "A survey on Finite Automata based pattern matching techniques for network Intrusion Detection System (NIDS)," in IEEE International Conference on Advances in Electronics, Computers and Communications (ICAECC'14), Oct. 2014

[24] Yamagaki, N., Sidhu, R., & Kamiya, S. (2008, September). High-speed regular expression matching engine using multi-character NFA. In Field Programmable Logic and Applications, 2008. FPL 2008. International Conference on (pp. 131-136). IEEE.

[25] Clark, C. R., & Schimmel, D. E. (2004, April). Scalable pattern matching for high speed networks. In Field-Programmable Custom Computing Machines, 2004. FCCM 2004. 12th Annual IEEE Symposium on (pp. 249-257). IEEE.

[26] Mitra, A., Najjar, W., & Bhuyan, L. (2007, December). Compiling pcre to fpga for accelerating SNORT ids. In Proceedings of the 3rd ACM/IEEE Symposium on Architecture for networking and communications systems (pp. 127-136). ACM

[27] Sourdis, I., Bispo, J., Cardoso, J. M., & Vassiliadis, S. (2008). Regular expression matching in reconfigurable hardware. Journal of Signal Processing Systems, Vol. 51,Issue,1, pp 99-121.

[28] N. Onizawa, W.J. Gross, and T. Hanyu, , "A Low-Energy Variation-Tolerant Asynchronous TCAM for Network Intrusion Detection systems," in Asynchronous Circuits and Systems (ASYNC'13), 2013 IEEE 19th International Symposium on, pp.8-15, 19-22 May 2013

[29] M. Faezipour, M. Nourani, "Wire-Speed TCAM-Based Architectures for Multimatch Packet Classification," IEEE Trans. Comput., vol.58, no.1, pp.5-17, Jan. 2009

[30] D. E. Taylor, "Survey and taxonomy of packet classification techniques," ACM Comput. Surv., vol. 37, no. 3, pp. 238–275, 2005.

[31] G. Vasilios, S. Antonatos, M. Polychronakis, E. P.Markatos, and S. Ioannidis, "Gnort: High Performance Network Intrusion Detection Using Graphics Processors,'. In Proce. of the 11th International Symposium on Recent Advances in Intrusion Detection (RAID'08), 2008

[32] G. Diana, D. Marco, M.P Joao, and B. Koen, "Reconfigurable Computing: Architectures, Tools, and Applications,": 10th Int. Symp., ARC 2014, Vilamoura, Portugal, April 14-16, 2014.

[33] S. Pontarelli, , G. Bianchi, , S. Teofili, , "Traffic-Aware Design of a High-Speed FPGA Network Intrusion Detection System", IEEE Trans. Comput., vol.62, no.11, pp.2322-2334, Nov. 2013

[34] T. Quang Anh, F. Jiang, , and Hu. Jiankun, "A Real-Time NetFlow-based Intrusion Detection System with Improved BBNN and High-Frequency Field Programmable Gate Arrays," in IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom'12), pp.201-208, 25-27 June 2012.

[35] K. Jaic, M.C. Smith, and N. Sarma, , "A practical network intrusion detection system for inline FPGAs on 10GbE network adapters," in IEEE 25th International Conference on, Application-specific Systems, Architectures and Processors (ASAP'14), pp.180-181, 18-20 June 2014

[36] A. Das, D. Nguyen, J. Zambreno, G. Memik,and A. Choudhary, "An FPGA-Based Network Intrusion Detection Architecture," IEEE Trans. Inf. Forens. Security, vol.3, no.1, pp.118-132, March 2008

[37] T. Quang Anh, F. Jiang, , and Hu. Jiankun, "A Real-Time NetFlow-based Intrusion Detection System with Improved BBNN and High-Frequency Field Programmable Gate Arrays," in IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom'12), pp.201-208, 25-27 June 2012

[38] Y. X. Xia, Z. C. Shi, Y. Zhang, , and J. Dai," A SVM intrusion detection method based on GPU," In Applied Mechanics and Materials, vol. 610, pp. 606-610, October 2014

[39] S. I. Kim, W. Edmonds, and N. Nwanze," On GPU accelerated tuning for a payload anomaly-based network intrusion detection scheme," in Proceedings of the 9th Annual Cyber and Information Security Research Conference, pp. 1-4. ACM, April 2014

[40] M. A. Jamshed, J. Lee, S.Moon, I. Yun, D. Kim, , S. Lee, and K. Park," Kargus: a highly-scalable software-based intrusion detection system," in Proceedings of the 2012 ACM conference on Computer and communications security, pp. 317-328, ACM, October 2012

[41] N. Jaehyun, M. Jamshed, C. Byungkwon, H. Dongsu, P. KyoungSoo, "Scaling the performance of network intrusion detection with many-core processors," in Architectures for Networking and Communications System (ANCS'15), 2015 ACM/IEEE Symposium on , pp.191-192, 7-8 May 2015.