Cent. Eur. J. Comp. Sci. • 4(4) • 2014 • 222-230
DOI: 10.2478/s13537-014-0213-6

VERSITA

Central European Journal of **Computer Science**

A Survey of parallel intrusion detection on graphical processors

Review Article

Liberios Vokorokos*, Michal Ennert†, Marek Čajkovský‡, Ján Radušovský§

Dept. of Computers and Informatics, Technical University in Košice, Letná 9 Košice, 042 00, Slovakia

Received 27 February 2014; accepted 26 June 2014

Abstract: Intrusion detection is enormously developing field of informatics. This paper provides a survey of actual trends in intrusion detection in academic research. It presents a review about the evolution of intrusion detection systems with usage of general purpose computing on graphics processing units (GPGPU). There are many detection techniques but only some of them bring advantages of parallel computing implementation to graphical processors (GPU). The most common technique transformed into GPU is the technique of pattern matching. There is a number of intrusion detection tools using GPU tested in real network traffic.

Keywords: intrusion detection • GPGPU • parallel computing • high performance computing • survey

© *Versita sp. z o.o.*

1. Introduction

Within the cost of information processing and Internet accessibility, organizations are becoming increasingly vulnerable to potential cyber threats such as network intrusions. There exists a need to provide secure and safe transactions through the use of firewalls, Intrusion Detection Systems (IDSs), encryption, authentication, and other hardware and software solutions. Intrusions can be defined as any set of actions that attempt to compromise the integrity, confidentiality, or availability of a computer system resource. Attackers can access the system from Internet or from inside as authorized users attempting to gain and misuse non-authorized privileges. There are more ways of computer system's defense. The most widely used is access control. This defense type allows an access to individual objects in the system but doesn't allow the user to control his activity with these objects. Intrusion detection has a significant role in the overall computer architecture. There can be defined as process of monitoring computer networks and systems for violations of security policy (the set of laws, rules, and practices that define the system boundaries and detail exactly what operations are allowed) [1].

* *E-mail: Liberios.Vokorokos@tuke.sk (Corresponding author)*
† *E-mail: Michal.Ennert@tuke.sk*
‡ *E-mail: Marek.Cajkovsky@tuke.sk*
§ *E-mail: Jan.Radusovsky@tuke.sk*

Intrusion detection system (IDS) plays a role of the last defense line in overall system security. Although it doesn't replace direct access control devices it is successful for monitoring and violation detection of network traffic attempts or anomaly activities. Thereby it provides important information for countermeasures in time. Computer networks without this addition are not secured enough against attacks in the organizations that can occur in everyday network traffic. IDSs usually work in a dynamically changing environment.

2. Background of IDS

Intrusion detection systems are based on two fundamental approaches: the detection of anomalous behavior as it deviates from normal behavior, and misuse detection by monitoring those "signatures" of those known malicious attacks and system vulnerabilities. IDS produces alerts, messages or reports for administrators. Its primary orientation is to identify and to record information about possible events and to report these attempts simultaneously [2]. There are three steps in the process of intrusion detection:

- Monitoring and analyzing traffic;
- identifying abnormal activities;
- assessing severity and raising alarm.

The potential goals of IDS usage include detection and prevention attacks, detection of policy violations, enforcement of use and connection policies and collection of evidence. There are several types of IDS and the choice of which one to use depends on the overall risks to specific network. One of the classifications of IDSs is established by the resource they monitor. According to this classification, IDSs are divided into two categories:

- **host-based (HIDS).** HIDS resides on a particular host and looks for indications of attacks on that host.
- **network-based (NIDS).** NIDS resides on a separate system that watches network traffic, looking for indications of attacks that traverse that portion of the network. The current trend in intrusion detection is to combine both host based and network based information to develop hybrid systems [3].

3. Detection techniques

There are two main categories of intrusion detection techniques [4]:

- **Anomaly intrusion detection** - IDS catches anomalies found in the data folder that are different from the normal system's behavior. The advantage of anomaly detection is the ability to detect novel attacks against software, unknown attacks, and deviations from normal usage of programs.
- **Misuse intrusion detection** - IDS collects previous invaders' attacks, characteristics and figures and saves them in the database. Therefore it is able to reveal attacks with the same characteristic or pattern in the future.

There is hybrid intrusion detection which combines anomaly and misuse approach. Target monitoring system is other technique used in IDS which don't search for anomalies or misuses. It looks for the modification of specified files or system status. Stealth probes IDS attempts to detect any attackers that choose to carry out their mission over prolonged periods of time. Intrusion detection techniques are categorized based on many different ways like statistics, neural network approach, data mining, genetic algorithm and computer immunology approach [5].
In anomaly intrusion detection, there are some method approaches which some of them can be self-learning. Statistical method monitors the user/network behavior by measuring certain variables statistics over time. Then, one of data mining algorithms is used to compute activity patterns from system audit data and extracts predictive features from the patterns. New patterns can be automatically discovered from a large amount of data there. Distance method tries to overcome limitations of statistical outlier detection approach when the data are difficult to estimate in the multidimensional distributions. Signature-based method matches available signatures in its database with collected data from activities

for identifying intrusions. Rule-based method uses a set of condition rules to detect intrusive patterns. Neural network method uses different combinations of the supervised learning multi-level perceptron, the unsupervised learning and reduction raw audit data of self-organizing maps or another topology of Elman network together for anomaly detection. Based on several models with different measures, fuzzy logic makes the final decision of whether current behavior is abnormal or not. Artificial neural network learns the behavior of training data collected from audit logs or user activities in a certain period. Computer immunology anomaly detection technique behaves as immune system, which protects animals from dangerous foreign pathogens or parasites or viruses [4].
Misuse intrusion detection includes methods which distinguish malicious intrusion from normal use like genetic algorithm. Expert system based method is a combined software and hardware capable of competently executing a specific task performed by specialized computer systems capable of simulating a human specialist's knowledge. State transition based method tries to identify intrusion by using a finite state machine. Comparison of selected intrusion detection techniques is presented on Figure 1. Two approaches, genetic algorithms and decision trees, are used to automatically generate rules for classifying network connections [4].

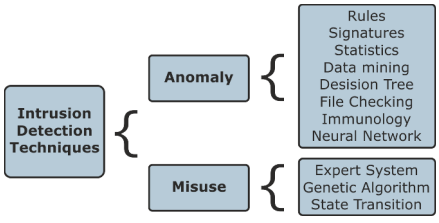

Figure 1. Intrusion detection techniques.

4. IDS performance issues

Intrusion detection systems must operate at multi-Gigabit speeds, while performing highly-complex per-packet and per-flow data processing. Thus, they (mainly signature-matching) can have two types of performance limitations [6]:

- CPU-bound limitations;
- I/O-bound limitations.

This is caused by the overhead of reading packets from the network interface card and high computing load in detection engine. This classification scheme is borne-out by runtime profiling results for well-known network-base intrusion detection system Snort (see chapter 7.1).

4.1. CPU-Bound Limitations

In IDS, load has a direct impact on performance. Detection mechanism network of IDS systems uses the most system features and in some cases it can occupy up to 75% of modern IDS systems' performance [7]. The main operation of detection engine is a comparison of incoming network packets to a set of signatures of known attacks. The comparison of packets to signatures is a string-matching operation which is CPU-intensive. Research into improving the runtime of the packet comparison falls into three categories: improvements in the string-matching algorithm portion of IDS, utilizing packet-header characteristics to optimize comparison and using hardware to improve IDS performance. The first two approaches are complementary and have been incorporated into most software-based IDSs, while the third approach has been widely deployed in router technology [6].

4.2. I/O-Bound limitations

Reading packets from the network interface card is a first major operation of IDS. Packet reading can become a bottleneck when the number of packets overwhelms the IDS host's internal packet buffers [6].

According to the [1], there are other limitations of IDS:

- **Encryption:** The ultimate threat to the very existence of the signature based NIDS systems is the increasing use of data encryption. Once the packet payloads are encrypted, the existing signatures will become completely useless in identifying the anomalous and harmful traffic.
- **New and sophisticated attacks:** NIDS which are signature based are unable to detect new attacks whose signatures are not yet devised. Anomaly based NIDS can detect such attacks but due to the limitations of the current anomaly detection algorithm.
- **Human intervention:** Almost all NIDS systems require a constant human supervision, which slows down the detection and the associated actions.
- **Evasion of signatures:** Some researchers have argued that it is not difficult for an attacker to evade a signature. Additionally there has been an increase in polymorphic worms which can automatically change their propagation characteristics thereby effectively changing their signatures. Such worms also pose a critical threat to the current NIDS.

5. Advantages of parallel computing implementation on GPU

It is necessary to constantly search more advanced and more sophisticated tools and methods that will be more effective to reveal the security violation and will be able to heal a violated system. Nowadays, we focus on the violation detection in real time (or with minimal latency) with maximum performance/throughput because invaders themselves can have a performance of distributed attacks or other types of attacks. The most important factors for effective violation detection are quick reaction time of IDS while analyzing networks and many processing packets at the same time. There are some factors [8] that complicate and restrict the activity of IDS when analyzing network traffic:

- The number of network elements/nodes is increasing. Most analysis algorithms are highly dependent on the number of elements/nodes on the network;
- Network traffic speed and amount is heavily increasing;
- Analysis algorithms are getting more complex;
- Computing analysis systems are reaching two computational limits, known as memory wall and instruction-level parallelism wall due to system architecture limitations.

Actual method how to remove these unwelcome facts is an effort to divide and distribute tasks to more computer stations. It brings only the time solution because of bigger scalability and usage of modern distributed architectures. But it doesn't reduce the impact to the memory wall. Moreover, this solution requires distributed software with communication and synchronization layer. Finally, price demand for the hardware is enormous. The use of GPU as a mathematical coprocessor executing complicated and computationally demanding detection algorithms can effectively increase the efficiency of network traffic analysis. Such attempt can eliminate complications coming from the previous factors even more [8].
NIDS using GPU can have the following theoretical advantages [8]:

- Offers a solution to the problem;
- Better performance, memory bandwidth and parallelism capabilities;
- High Scalability;

[1] *Sailesh K., Survey of Current Network Intrusion Detection Techniques, 2007, website: http://www.cse.wustl.edu/ jain/cse571-07/ftp/ids.*

- Costs could be lower;
- System could still be a distributed system.

These facts are the main reason of appropriate use of distributed parallel architectures for intrusion detection with usage of GPU.

6. Related parallel computing implementations

The intrusion detection with usage of GPU introduced many procedures of implementation and parallelization recently. One of the most often method of implementation is transformation of IDS detection mechanism to GPU, mostly at pattern-matching technique which calculation algorithm is executed on GPU.
In the article [9], there is the famous Wu-Manber (WM) algorithm [10] transformed into GPU-based multiple-pattern matching algorithm for NIDS. The proposed algorithm exploits the computational power and high memory bandwidth of GPUs and makes use of the task parallelism of GPUs to process the content of a packet in parallel. Experimental results show that the purposed algorithm offers twice performance of the modified WM algorithm used in famous open-source network IDS Snort. But these experiments were executed on personal computer. The efficiency of intrusion detection execution could be higher by using high-performance GPU (e.g. NVidia Tesla).
The article [11] present an efficient GPU-based pattern matching algorithm by leveraging the computational power of GPUs to accelerate the pattern matching operations and to increase the over-all processing throughput with a maximum of 2.4 Gbit/s. It takes advantage of DMA execution of GPUs to impose competition between the operations handled by the CPU and the GPU. The proposed method balances the workload among the thread, and it performs the pattern matching based on hierarchical hash table architecture on GPU. The experiment results demonstrate the proposed method achievement with 10x speedup over the CPU implementation of Aho-Corasick algorithm [12] for various UDP packet sizes.
In the article [13] local outlier factor algorithm is modified and transformed into GPU as a very powerful anomaly detection method available in machine learning and classification. The principle of modification was in CUDA-based GPU implementation of the k-nearest neighbor algorithm to accelerate LOF classification. There was achieved more than a 100X speedup over a multi-threaded dual-core CPU implementation of LOF method with small impact of increasing arguments' values on computation time.
In [14], system parallelizes network traffic processing and analysis at three levels, using multi-queue NICs, multiple CPUs, and multiple GPUs. This work introduces a new multi-parallel architecture based on Snort for high-performance processing and stateful analysis of network traffic. The proposed design avoids locking, optimizes data transfers between the different processing units, and speeds up data processing by mapping different operations to the processing units where they are best suited. The big advantage of design is in its support for pattern matching using several GPUs at a data-parallel level. The experimental results reach processing speeds of up to 5.2 Gbit/s when analyzing traffic in a real network, whereas the pattern matching engine alone reaches speeds of up to 70 Gbit/s.
In [15], there is an interface for NIDS Snort that allows porting of the pattern-matching algorithm to run on a GPU with achieving up to four times speedup over the existing Snort implementation. Modification rests in porting the string-matching algorithm used in Snort, Aho-Corasick, to run on a GPU. The algorithm depends on a set of DFAs (deterministic finite automaton to perform simultaneous pattern matching) for the string comparison. These DFAs need to be transferred to the GPU memory for the string comparison.
The article [8] describes a new developed framework for programming network traffic analysis on GPUs based on CUDA technology. It fulfills the conditions like open source; easy extensible; scalable; modular; easy to use and it is well documented. The framework is able to perform many types of analysis: real-time analysis, batched analysis and forensics analysis. However, it has some specific limitations based on the technology used. But the developers continue to improve and reduce these issues.

7. Intrusion detection tools using GPU

There are numerous tools for detecting intrusions using IDS principles nowadays. These can be specific with the use of own methods and actions. The most commonly used is the open-source network IDS solution Snort (and its software

derivations) which is able to detect and alert in case of an attack. Moreover, it has great presumption for the use of lifting security by implementing GPGPU technologies in addition with distributed systems.

7.1. Snort

Snort[2] is a signature-based Intrusion Detection Prevention software package that performs real-time network traffic analysis and logs the output. It represents an open-source IDS solution that is able to detect and alert the attacks. During the designing of Snort modular architecture and a language based on rules were used. The fact that it is an open-source solution brings many advantages:

- Its evolution is faster than in embedded systems;
- The community of security experts constantly publishes reviews, creates tests and designs for improving the tool source code;
- Security technicians and specialists create new Snort rules for new threads in very short time periods.

Snort architecture consists of a number of components. *Sniffer* generates packets from the network traffic and sends them into the packet decoder, which uses the Libcap Library. This module is represented by a network interface which accumulates packets. *Packet Decoder* decodes data from the packets on the second layer of the ISO/OSI model and sends them to the processor. *Preprocessor* decides whether the detection mechanism will process the gathered data because the detection mechanism has greater hardware requirements for the system itself. *Detection engine* is the core of the whole Snort tool which compares gathered data with the database of stored Snort data based on its rules. *Output* defines the format and the action how the data on the output will be represented.
The majority of networks systems for intrusion detection uses for its operation models of finite-state automats and regular expressions to analyze packets/patterns, compare and evaluate. Snort operates on the rules basis. E.g. there are rules for ports scanning detection, DoS attacks etc. Snort uses and easy but flexible and strong language for rules creation. By comparing of each caught packet within a sequence it is possible to create a filter for not appropriate content.

7.2. Pixel-snort

This is one of the first Snort extensions with GPGPU technology. Pixel-Snort was created by scientists from Tufts University in Medfort in 2006. This solution tries to solve the problems of IDS performance with off-load packet processing in detection engine into GPU. It focuses on lessening the CPU-load of the IDS by offloading the string matching portion in the detection engine of the open source IDS Snort to GPU. Experiment ports parts of Snort to GPU using the Cg programming language. For CPU-intensive workloads, the results demonstrated GPU can successfully be used for off-load computation from CPU. The proposed approach decreases CPU load beyond 50% compared to Snort [6].

7.3. Gnort

Network system for intrusion detection Gnort is one of the first tools that used effective calculation performance of GPGPU technologies. Gnort is a project developed by scientists of Computer Science in Heraklion on the Crete Island since 2009. It is based on the open-source system Snort that uses calculation power of modern graphic adapters. As it was mentioned, detection mechanism of NIDS systems uses the most of system facilities. This load mainly causes processing of each byte of caught packet within an algorithm searching strings and following analysis and comparison with the rues and data in IDS database. Therefore in Gnort this task was sold and parallelized to the GPU calculations. Whole process is based on the architecture (Figure 2) which main function power is three functions of caught packets processing:

[2] *Snort - Network Intrusion Detection System, 2013, website: http://www.snort.org.*

- **Data transfer from CPU to GPU** - According to the programming model of GPU architectures the packets are sent to the graphic hardware in sections. Gnort organizes content signatures to the groups according to the number of source ports and packet destination for each rule. It uses separate buffer in a way that it copies the packet to the memory after group assignment. When the memory is full or 100 milliseconds expire the content of the memory is automatically sent to the GPU for further calculation.

- **Patterns comparison in GPU** - As soon as packets are brought in the GPU the comparing algorithm that compares all bytes of packets in the flow is activated and the status is evaluated. This ensures deterministic finite-state automat saved as 2-dimensional array. This matrix contains all possible statuses that can occur and based on this matrix the automat evaluates the packets.

- **Re-transmission of data from GPU to CPU** - Every time when GPU processes all flow of given bytes it informs CPU about it. Status packets reports are written into the GPU memory as a matrix line for each packet. After fulfillment of the matrix whole evaluated content is sent backwards to CPU [16].

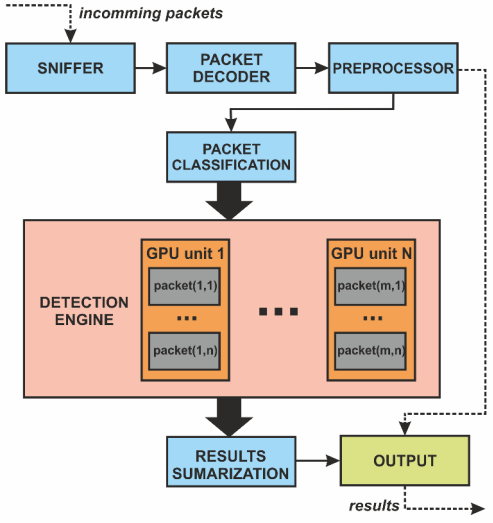

Figure 2. IDS architecture using GPU.

By implementing this architecture it is possible for GNORT to decrease the load of all system and thereby to increase throughput of all NIDS. First results of the prototype approached the throughput up to 2-3 Gbit/s in real time. The results of implementation of GPGPU technologies to IDS in this case confirm that modern graphic cards can be effectively used to speed up the system intrusion detection calculations based on comparison of obtained data with the data saved in the IDS database [16].

7.4. Suricata

Network system of intrusion detection called Suricata[3] is a project of non-profit organization OISF (Open Information Security Foundation) in which the company NVidia participates, too. Its target is to implement IDS of new generation that represents faster and more efficient solutions. Suricata offers some advantages:

- **Support of more fibers (multithreading)** - Divides the load in more processes;

- **Compatibility of rules** - Almost all rules are compatible with NIDS Snort;

[3] *Suricata, 2013, website: http://suricata-ids.org.*

- **Automatic detection of protocol** – Suricata is able to automatically detect protocols what simplifies the configuration of system;
- **Statistical detection of anomalies;**
- **GPU support and experimental support of CUDA** – allows to parallelize performance on GPU.

Suricata takes all rules set of Snort tool in the same way as NIDS. Its source code is written clear and modularly and offers a lot of scalability. Theoretically it is able to reach the throughput up to 10 Gbit/s. It is an interesting and modern direction of NIDS development with deployment of GPGPU technologies [17].

8. Conclusion

The evolution of GPGPU technologies is moving forward every day from both hardware and software point of view and it provides new possibilities of use. Nowadays, there are some functional prototypes in which these technologies were successfully implemented and tested in practice. Figure 3 provides general overview of famous implementations of GPGPU technologies in intrusion detection. Blue cells show real solutions/applications as intrusion detection systems using GPU. Grey cell describes proposed framework based on CUDA metalanguage. Green rows represent very interesting experiments or proposals as a good inspiration for our future work. Intrusion detection in the field of Computer Science faces many limits that move forward. New more sophisticated detection techniques arise that detect potential invaders effectively. This requires high-computing performance. As a result of this article, parallel computing and GPGPU Technologies are the right choice how to accelerate computation performance of IDS today. Moreover, if high computational capacity is needed, intrusion detection systems could be also extended by a group of computers using each of them GPUs to distribute computing process.

Work	Year	Based on	Improving Method	Experimental Results
NIDS Pixel-Snort	2006	Snort	Off-load packet processing in detection engine to GPU	Decreases CPU load by 50%
Multiple-pattern Matching Algorithm	2008	Wu-Manber algorithm	Transformed into GPU as multi-pattern matching algorithm	Twice computational performance
NIDS Gnort	2009	Snort	Off-load packet processing in detection engine to GPU	2,3 Gbit/s network traffic bandwidth
Framework for network traffic analysis	2010	CUDA	New framework for programming network traffic analysis	---
Accelerating the LOF algorithm	2010	Local outlier factor algorithm	GPU implementation of the k-nearest neighbor algorithm to accelerate LOF classification	100x speedup than multicore CPU
Multi-Parallel IDS Architecture	2011	Snort	Multi-parallel architecture	5,2 Gbit/s: network traffic 70 Gbit/s: pattern matching
NIDS Suricata	2011	Snort	Experimental CPU and CUDA technology support	10 Gbit/s network traffic bandwidth
Efficient Packet Pattern Matching	2012	Hierarchical hash table architecture on GPU	Balances the work load among the thread	10x than Aho-Corasick algorithm
Parallelizing NIDS	2012	Snort	Aho-Corasick algorithm on GPU	4x speedup than CPU implementation

Figure 3. Comparison of IDS using GPU.

Acknowledgment

This work was supported by the Slovak Research and Development Agency under the contract No. APVV-0008-10, APVV-0073-07 and KEGA 008TUKE-4/2013 Microlearning environment for education of information security specialists.

References

[1] R. Bace, Intrusion Detection, 2nd Edition (Macmillan Technical Publishing, Indiana, 2000)
[2] L. Vokorokos, A. Baláž, M. Chovanec, In: Towards Intelligent Engineering and Information Technology, Volume 243, Studies in Computational Intelligence (Springer, Berlin, Heidelberg, 2009) 389–403
[3] P. Fanfara, A. Pekár, Usage of Hybrid Honeypots an Intrusion Detection System Mechanism, SCYR 2012: Proceedings from conference : 12th Scientific Conference of Young Researchers, 2012
[4] S. Sonawane, Sh. Pardeshi, G. Pradad, A Survey on Intrusion Detection Techniques (Department of Information Technology, Technocrats Institute of Technology, Bhopal, India, April 2012)
[5] V. Marinova-Boncheva, A Short Survey of Intrusion Detection Systems (Institute of Information Technologies, Sofia, 2007)
[6] N. Jacob, C. Brodley, Offloading IDS Computation to the GPU (Computer Science Department, Tufts University, Medford, 2006)
[7] J.B.D. Cabrera, W. Lee, R.K. Mehra, On the Statistical Distribution of Processing Times in Network Intrusion Detection, CDC., 43rd IEEE Conference on Decision and Control vol. 1 (IEEE, 2004) 75–80
[8] M.S. Clos, A Framework for Network Traffic Analysis Using GPUs (Universitat Politecnica de Catalunya, Barcelona, 2010)
[9] Huang N., Hung H., Lai S., Chu Y., Tsai W., A GPU-based Multiple-pattern Marching Algorithm for Network Intrusion Detection Systems, 22nd International Conference on Advanced Information Networking and Applications, Workshops, 2008.
[10] S. Wu, U. Manber, A Fast Algorithm for Multipattern Searching, Technical Report TR-94-17 (Department of Computer Science, University of Arizona, 1994)
[11] Ch. Hung, Ch. Lin, H. Wang, Ch. Chang, Efficient Packet Pattern Matching for Gigabit Network Intrusion Detection using GPUs, 14th International Conference on High Performance Computing and Communications (IEEE, Liverpool, 2012)
[12] A.V. Aho, M.J. Corasick, Efficient string matching: An aid to bibliographic search, Communications of the ACM 20, 761–772, 1977
[13] M. Alshawabkeh, B. Jang, D. Kaeli, Accelerating the Local Outlier Factor Algorithm on a GPU for Intrusion Detection Systems (Dept. of Electrical and Computer Engineering, Northeastern University, Boston, MA, 2010)
[14] G. Vasiliadis, M. Polychronakis, S. Ioannidis, MIDeA: A Multi-Parallel Intrusion Detection Architecture (FORTH-ICS, Greece, 2011)
[15] A.P.M. Sathik, Parrallelizing a Network Intrusion Detection System using a GPU, (B. Tech. University of Kerala, India, 2012)
[16] G. Vasiliadis, S. Antonatos, M. Polychronakis, E.P. Markatos, S. Ioannidis, Gnort: High Performance Network Intrusion Detection Using Graphics Processors, Proceedings: The 11th international symposium on Recent Advances in Intrusion Detection, Sept. 15-17 (Springer, Berlin, Heidelberg, Cambridge, MA, USA, 2008)
[17] L. Vokorokos, M. Ennert, M. Čajkovský, A. Turínska, A Distributed Nerwork Intrusion Detection System Architecture Based on Computer Stations using GPGPU, INES 2013 : IEEE 17th International Conference on Intelligent Engineering Systems, Costa Rica, Budapest, 2013