ISSN: 2277-9655
(I2OR), Publication Impact Factor: 3.785

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

FIREWALL ENGINE BASED ON GRAPHICS PROCESSING UNIT

Akash Belkhede *, Sanket Sonawane, Chaitanya Khirsagar, Prof. D. D. Sapkal
Department of Computer Engineering Pune Vidhyarthi Griha's College of Engg. & Tech. Pune, India

DOI: 10.5281/zenodo.47575

ABSTRACT

A firewall is very important object in network security. The Working of firewall policy rules has become time consuming and very much complex. The main issue is the slow filtering action during heavy load of internet traffic. To reduce the time consumption there is a need to Use parallel computing for optimized firewall engine, which runs on GPU's. We mainly focus on the creation of parallel algorithms for firewall system by using parallel computing which reduces the time consumption and concurrently it can allow for strong threat and virus detection of incoming packets by using GPU enabled CUDA platform for parallel computing. In this paper we learn different firewall policy rules and parallel computation done by GPU for threat detection, packet filtering which runs on NVIDIA's GPU card and it is based on CUDA platform. For our project we are creating test packets and for virus scanning we are going to use virus database.

KEYWORDS: Firewall, CUDA, Parallel Computing, GPU, NVIDIA, Threat Detection, Clam-AV.

INTRODUCTION

Firewall is a network security design system that controls and access the incoming and outgoing network traffic based on firewall policy set of rules. A firewall is a system designed to secure unauthorized access to or from a private network. The firewalls in now a days are very slow due to the great traffic in the network. Modem firewalls work on access-lists. The increase in virus signatures database also makes virus-scanning for incoming traffic i.e. very slow with serial processors. Also the increase in network bandwidth requires fast threat detection, intrusion detection, network address translation so that bandwidth of the traffic crossing internet remains constant along with assuring good quality of service.The basic computers CPUs are also insufficient due to the heavy load of packets and compute intensive packet processing. One of the way, to address this problem of slow filtering is to process the packets in parallel using number of processors.

The Graphics Processing Units manufactured by NVIDIA contain thousands of cores which can process data in parallel. NVIDIA has provided a framework called CUDA which gives access to parallel processing power of the GPUs for non-graphic application processing. We are going to take advantage of the same. We are going to use these all multiple cores of the graphics card for the purpose of the processing the packets in parallel. So that multiple packets could be processed in the same amount of the time as that which is required by the single packet by using CPU. More number of the packets and increase in the network bandwidth forces to exploit the high throughput processing feature of the GPU. Modern GPUs are very efficient at high throughput and processing their highly parallel structure makes it more effective than CPUs for algorithms where processing of large blocks of data is done in parallel.

While offloading the CPU intensive part to GPU's also has some bottlenecks such as the bandwidth of the PCI slot to which the GPU is connected. We try to overcome this issue using asynchronous streaming process of CUDA which tries to hide the latency to some extent. In today's scenario most of the packets are encrypted using HTTPS algorithms. And it's not known which level of encryption is done to the payload of the packet. So there are no standard signatures which can be matched with those packets. Hence virus scanning cannot be performed on those packets. We have no solution for this problem, in our work we did virus scanning with only those packets whose encryption level and type is known and we have available signatures for it.

Virus scanning of incoming packets involves comparison of packets with signature strings which is very much CPU intensive. On the other side packet filtering rules also need to match with each incoming packet, it also becomes very much CPU intensive when there is high load and big size of packet filter rules list. Threat detection can be measured by checking the drop rate of packets, advanced threat detection involves creating data base about information of packets whose drop rate crosses the threshold value. Now during heavy load or congestion when the firewall has to perform intrusion detection, packet filtering, threat detection, network address translation and scanning the incoming packets.

In this framework, GPU used instead of CPU (central processing unit). CPU is not capable to handle large data sets and it requires lots of time to perform large operations. There are lots of benefits of GPU like Faster Processing, Lower capital cost, reduced power consumption; The GPUs extended the performance beyond what is possible with CPUs alone at any cost.

Our Experiments are done on simulated packet or traffic systems on Windows by using Visual Studio CUDA C/C++. The virus signatures have been downloaded from various packet traffic websites and Clam-AV for virus-detection.

RELATED WORK

The recent work includes packet filtering of firewalls, virus scanning of incoming traffic. Previously all of the algorithms have been made serial, which also prevents strong threat detection for incoming traffic. There have been proposed many algorithms to minimize the time effect such as decision-tree algorithm, dynamic-rule ordering, multi-field alphabetic tree, Huffman tree filtering etc. Most of the algorithms used to optimize the virus-scanning time which is serial execution and it use less number of cores of general CPU's. CUDA platform on GPU used to speed up the serialized optimization techniques.

Firewall policies and rules

Each firewall model has several sets of rules associated with it: Virus signature for virus scanning, Threat detection and Network Address translation (NAT), signatures of IDS

Analyzing virus signature for virus scanning

Mostly the virus-signatures are of three types.

• Basic Signatures: These types of signatures are all hexadecimal strings. These are basic types of signatures from the full content of any files.

• MD5 Signatures: The files which infected by virus are matched by MD5 signatures present in their content are MD5 checksum of a target file.

• Regular Expression Signatures: This is the next version of basic signatures. This in addition to basic signatures it includes several kinds of extend viruses. Generally the antivirus matches this type of signatures with the whole content of the file.

Signatures for Intrusion Detection Systems

A network IDS signature is a pattern that we want to look for in traffic. Below are some of the examples of IDS signatures

• Connection attempted from a reserved IP address can be filtered by checking the source address field in an IP header.

• Packet having an illegal TCP flag combination can be filtered by comparing the flags set in a TCP header for known good or bad flag combinations.

• Email containing a particular virus can be filtered by comparing the subject of each email to the subject associated with the virus-laden email, or it can look for a fake emails with particular same name.

• Filtering DNS buffer overflow attempt contained in the payload or actual information of a query, by resolving the DNS fields and checking the length of each of them, the signature can identify an attempt to perform a buffer overflow using the DNS field. A different method which can be used for exploiting shell code sequences in the payload.

• Denial of service attack on a POP3 server is caused by issuing the same command thousands of times. On signature

for this attack would be used to keep track of how many times the command is issued and to alert to system when that number exceeds a certain threshold.

• File access attack on an FTP server is caused by issuing file and directory commands to it without logging in first time. A state-tracking signature could be developed in which FTP traffic for a successful login and would alert if certain commands were issued before the user had authenticated properly.

Threat detection and Network Address translation

Threat detections measures the rate at which the packets are dropped may be for various reasons. And in advanced threat detection mode, the firewall creates a database about the packets which are dropped. So that the firewall can alert the user about the threat. In network address translation usually, the incoming packets are forwarded to another address inside a network to ensure high end security. But at the same time it also includes matching the arriving packets address with the inside network address so that it can be rightly translated

Parallelization Technique On CUDA Enabled Platform

In order to make the best use of the architecture of the GPU and CUDA platform, we propose a scheme for firewall engine based on parallel algorithms experimented on CUDA-based platform. A whole task is assigned to CPU and inside GPU; the no of blocks and threads per block is decided before kernel starts as it depends on virus signature size, IDS signature size and total number of rule-set for packet filtering, intrusion detection and network address translation. We propose an algorithm to generate no of blocks and threads which give us maximum optimization with current Tesla graphics card. In CUDA platform, the proper combination of threads and blocks gives us much better result than other.

PROPOSED WORK

Overall System Architecture

Figure 1 depicts the overall proposed system architecture based on Firewall Engine, framework to be implemented by using Visual Studio CUDA C/C++ and GPU.

Figure 1:

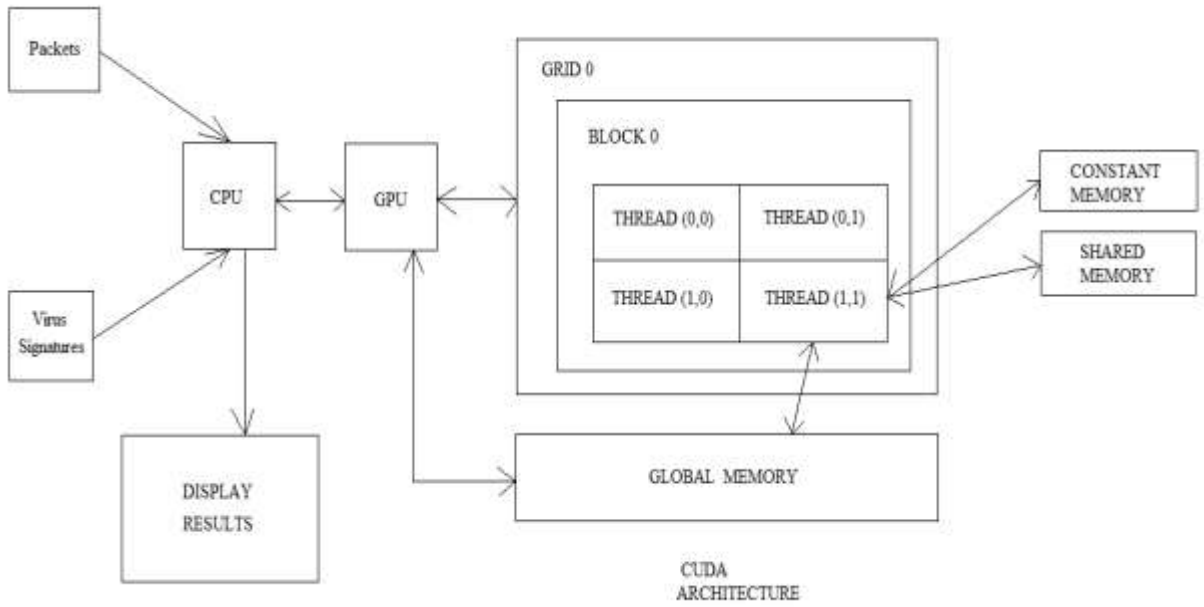

Figure 1.Sysyem Architecture

The overall system design consists of the following modules:

- CUDA thread block
- GPU
- CPU
- Incoming packets & Virus Signature

The input to the system is incoming packets & virus signatures which is captured by Wireshark packet analyzer. We will be focusing on parallelization of packet filtering and threat detection on GPU with CUDA architecture. Proposed implementation will consist following components:

- Maximum packets for packet filtering and threat detection databases captured by Wireshark packet analyzer
- Availability of both CPU and GPU processor
- NVIDIA Graphics card integrated on CUDA Platform.
- Design our own virus signatures

We searched for different packet creation methods and software. Wireshark creates the packets gives the description packets such as no of the packet, source and destination address for the packet, protocol, packet information as well as actual packet content in ASCII hexadecimal format. Wireshark is free and open source software available on the internet.
Wireshark is a packet analyzer which uses to capture packets in your computer. Wireshark is a free network protocol analyzer that is compatible with linux, windows as well as MAC

Graphics Processing Unit

A Graphics Processing Unit is a single-chip processor primarily used to manage and boost the performance of video and graphics. GPU can be used in many applications like computer gaming and graphics, image processing, designing automobiles, Graphical user interface, Construction, movie animation, medical field.

Figure 2:

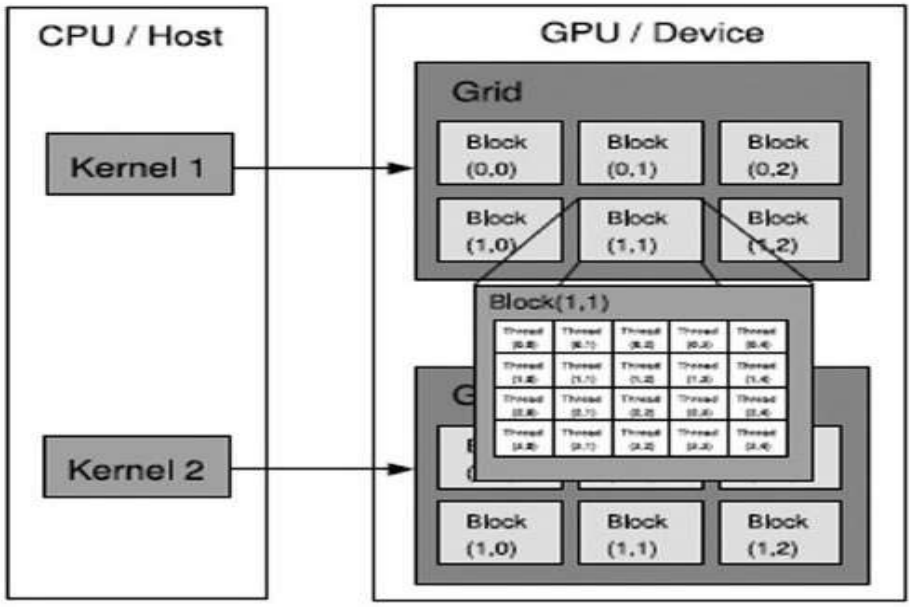

Figure 2 Architecture of Device Memory in GPU

For better performance CUDA platform will be used for GPU programming. CUDA is well suited for large datasets and highly parallel algorithms. CUDA is NVIDIA's parallel computing architecture in our GPUs. NVIDIA based CUDA architecture provides a complete toolkit for programming that includes the compiler, debugger, profiler, libraries and other information. The CUDA architecture supports C and Fortran like programming language, and APIs for GPU Computing, such as OpenCL and Direct Compute. The CUDA platform is accessible through CUDA-accelerated libraries, compiler directives, application programming interfaces, and CUDA platform is extensions to industry-standard programming languages. This project focuses on CUDA C programming. CUDA is also a scalable programming model that enables programs to transparently scale their parallelism to GPUs with varying numbers of cores, while maintaining a shallow learning curve for programmers familiar with the C programming language.

A CUDA program consists of a mixture of the following two parts:

- The host codes runs on CPU.
- The device code runs on GPU.

PROPOSED ALGORITHM

In this section, we present the detailed CUDA implementation of parallel packet-filtering along with threat detection algorithm, parallel computing-Firewall especially, we present how our proposed scheme is applied to this algorithm. Our algorithms are been divided into two
Major parts based on their working.

Serial Implementation

- Convert .pcap file to psml which can be treated as XML file.
- Now parse this XML file using tinyxml2 parser. TinyXML is use as parser.
- Store all the data which is to be needed like source_ip, dest_ip of a packets in a structure database.
- Write functions for reserved_ip, illegal _tcp_flag etc .by using the firewall policy rules.
- Run the serialized code and record the serial execution time.

Parallel Implementation

- Make use of structure database of serialized implementation for accessing data.
- Write CUDA library functions for the parallel computing based on NVIDIA based Graphics Processing Unit.
- CUDA library functions such as cudaMalloc( ) functions use for allocating memory to accessing data.
- cudaMemcpy( ) use to copy all data from CPU to GPU.
- Run the parallel code and record the parallel execution time.

Using both implementation methods we can show that speed up parallel computing over serialized implementation.

MATHEMATICAL MODEL

SYSTEM = {INPUT, OUTPUT, FUNCTIONS, FAILURE, SUCCESS}
INPUT = {PAC_1, PAC_2, PAC_3,PAC_4,……PAC_N}
OUTPUT= {PURE PACKETS, INFECTED PACKETS, Detection for threat in system}
INFECTED PACKETS = {$IPAC_1$, $IPAC_2$ $IPAC_3$, ……..IPACn }
PURE PACKETS = {$PPAC_1$, $PPAC_2$, $PPAC_3$,……..PPACn}
FUNCTIONS = {Filtered_packets(), Threat_packets(), reserved_IP(), illegal_ TCP_flag(), DNS_length(), DOS(), FTP_check(), threat_check_flag() }
SUCCESS = {Packets are filtered (infected packets are detected)}
FAILURE = {Infected packets are not detected as infected}

CONCLUSION

In this paper we have concluded that parallelization of packet filtering and threat detection on GPU with CUDA architecture can be able to give us Speedup over serial implementation of firewall on GPU. To exploit the new parallel platform for firewalls, we proposed optimized CUDA-based parallel techniques i.e. parallel firewall algorithms. As our size of packet set increases, our speedup also increases. From the above discussion, we believe that the GPU with CUDA parallel computing architecture will provide compelling benefits in the fields of security or firewalls applications. We will get better performance in terms of execution time by optimizing our parallel algorithm based on release of new CUDA graphics cards.

ACKNOWLEDGEMENTS

We express our sincere thanks to Prof. D. D .Sapkal PVG's COET, Pune for their valuable guidance in the overall execution of this project work. We would also like to thanks Prof. Dr. G. V. Garje and Prof. A. M. Bhadgale.

REFERENCES

[1] A. V. Aho and M. J. Corasick, Efficient string matching: An aid to bibliographic search, Communications of the ACM, 18(6):333-340, 1975.
[2] R. S. Boyer and 1. S. Moore. A fast string searching algorithm, Communications of the ACM, 20(10), 1977.
[3] T. Kojm, Clam-AV, In http://www.clamav.net. 2004.
[4] Source fire, Introduction to Clam-AV. http://www.sourcefire.com/products/clamav. 2007.
[5] O. Erdogan and P. Cao, Hash-A V: fast virus signature scanning by cache-resident filters, Proc. Of the International Conference on Systems and Networks Communications (ICSNC), 2007.
[6] F. Skulason, The evolution of polymorphic viruses, In http://vx.netlux.org/lib/staticlvdatlpolyevol. htm, 2004.
[7] B. Commentz-Walter. A string matching algorithm fast on the average. Proceedings of ICALP, pages 118–132, July 1979.
[8] C. Cowan, S. Arnold, S. Beattie, C.Wright, and J. Viega. Defcon capture the flag: Defending vulnerable code from intense attack. In DARPA DISCEX III Conference, Washington DC, April 2003.
[9] W. Eatherton, Z. Dittia, and G. Varghese. Tree bitmap: Hardware/ software ip lookups with incremental updates. Unpublished, 2002.
[10] NVIDIA (2008) CUDA programming guide 2. 1. http://www.nvidia.com/objectlcuda_develop.html.
[11] M. Fisk and G. Varghese, An analysis of fast string matching applied to content-based forwarding and intrusion detection, Technical Report CS2001-0670, University of California – San Diego, 2002.
[12] Nigel Jacob and Carla Brodley: Offloading IDS computation to GPU, Proceedings of the 22nd Annual Computer Security Applications Conference (ACSAC'06)
[13] Symantec, Network Intrusion Detection Signatures. http://www.symantec.com/connectlarticles/network-intrusiondetection-signatures-part-one.