Design and Implementation of Network Intrusion Detection System Using FPGA

Rashmi M Kambalimath[1] Gayatri Patil[2] Mr.Mahesh.B.Neelagar[3]
[1,2]M.Tech. Student [3]Assistant Professor
[1,2,3]Department of VLSI Design & ES
[1,2,3]Centre of PG studies, VTU, Belagavi, India

Abstract— Today's network security is more important factor, which mainly depends on Network Intrusion Detection System (NIDS). NIDS system detects the erroneous data in the network. The FPGA is the most relevant appealing technology which has ability to punctually update the supported rules and detects the new emerging attacks in network. The issue is how to scale FPGA based NIDS implementation to ever faster network links. The important factor is to balance traffic over multiple hardware blocks. Each hardware block implements the whole rule set. The NIDS system can be implemented in software based technology as well. The hardware blocks of NIDS system uses CAM which results in achieving high speed up to 277.635MHz. The hardware implementation is carried out using Verilog HDL language on SPARTAN-6 FPGA kit.

Key words: FPGA, Network Intrusion Detection System

I. INTRODUCTION

The NIDS has an important role in the security of the system. Because of internet threats the security and the protection has more demand in today's technology. NIDS has the information about the network traffic and packet classification that is based on header information that is based on pattern or content matching and further inspects the data which is present in the payload with respect to regular expression matching or content matching based on the rules. This detects the occurrence of attacks or variations.

In this project, we can make changes up to date in the supporting rule set for the new coming hacking techniques. For this we consider the very relevant technology i.e. FPGA based hardware implementation of NIDS. It can also used for the faster networks. In this case the important problem is that how to reduces the resources for the faster networks. The main intention is to maintain the multiple users by using hardware blocks. Each block contains all the rule sets. Here the problem is that as there is increase in the traffic it also increases the usage of resources. To overcome this problem we develop the different "traffic aware modular approach" for the NIDS which is based on the FPGA. Instead of purely dividing the traffic, we can divide and group it to the homogeneous traffic and then sent it to the suitable hardware blocks. Those blocks support the whole rule set that fitted to the traffic category. From the real world traffic statics of the heaviest traffic used by the operator's in the communication network, the use of homogeneous classification can save resource up to 80%.If we spread the network widely then we have to increase the security and protection against the attacks and threats. These are ever increasing with the network connectivity. In this situation NIDS is more important for security. It examines the traffic in detail while crossing through the network.

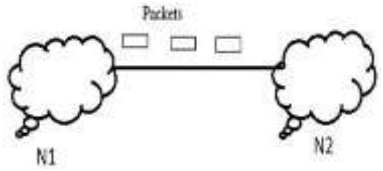


Fig. 1: Network System

Each packet should be checked and classified based on the header information, content which is to be transmitted through network. Each packet should be inspected by pattern matching and further evaluate pay load(data) with respect to data by using standard matching rules for identifying the incidence of variations in the data. Software based NIDS has more scope for software implementation and it cannot tolerate the high speed traffic rates. It also cannot sustain the large network; it is used only for small network and large traffic system. To overcome this we need to go for hardware implementation. The FPGA technique is very appealing technology. Because hardware implementation can give the frequent updates of rule set, we have to secure the network from the continuous incidence of different kinds of network intrusion threats and attacks. It is easy to dynamically reprogram when the SME rules are changed. Recent FPGA technology is capable of provide a very elevated speed dealing out and sustain interfaces for elevated speed. Now FPGAs are available at 100 Gb/sec and we can get up to 400 Gb/sec speed. By comparing the scaling device frequency and traffic collection is not meeting the desired value. But, the traffic collection capacity is not match with device frequency of the comparable scaling. The frequencies are in the order that still logic resources operate.

Device frequency is of 500MHz, which is reachable only by last production of FPGA device. It can be used to process 8- bit characters at 4 Gb/sec. By last generation FPGA device achieve frequency of 500MHz. The device can be used to process 8- bit characters at speed of4 Gb/sec.

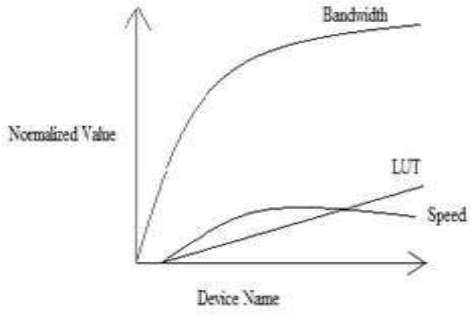


Fig. 2: Evaluation of the FPGA characteristics

From the above graph as the device higher version, then speed will be increases with the number of LUT's. The

bandwidth follows the Gilder's law and Moore's law for logic resources. As Gilder's law states that today, more data to be sent using distinct cable in 1sec than months of duration of information can sent over the whole Internet in 1997. We need the parallelization for the higher speed operation. It is recent trend in the Microprocessor. By using FPGA technology, we can analyze the NIDS traffic more easily and it is compulsory method to tolerate the better network throughput. For this we have find better ways to parallelize a NIDS structural design. The clear way is to collect the traffic from corner to corner from the multiple hardware components; this is developed to check the packets using the equivalent set of rules.

By using FPGA technology, we can analyze the NIDS traffic more easily and to sustain the increased network throughput it is to be mandatory approach. We have to find out the improved behaviour to parallelize NIDS architecture. This is the obvious way is to collect the traffic across the multiple hardware modules, this traffic is devised and packets are inspect by the same set of rules.

In this project we proposed traffic awareness method; this idea is to be carried out by processing the packets through Dispatcher by the following procedure.

1) Uses the basic information (port, protocol etc) using that we can classify that into different categories.
2) They are sending to the "string matching engines", in that hardware component in responsibility of taking rule set.

The above concept is very simple, but this is not so easy to implement in real world.

There are some reasons for the difficulty.

These are namely,

- Traffic classification rules must be extremely simple. It contains only header information and it also restricts the other rules to be applied.
- The traffic division in non uniform in this section the dimensioning of SME is not predictable and it is not in normal link speed. Its speed depends on the actual per category traffic load.
- The traffic rules which are to be implemented must be disjoint as much as possible from each other for dedicated hardware modules.

This technique will minimize the logic resources.

A. "SNORT Rules Analysis and relevant classification policies":

By analyzing the entire rule set we divided those rule sets into as subsets and those must be disjoint to each other of Snort rules. The packets are recognized by suitable combination of packet headers fields. For each subset the rules are changed are by each protocol. In each sets there are subsets which are extremely distinct. It is dedicated to analysis that threats still for http protocol against the WEB clients. This above analysis gives the classification methods that are devised in the dispatching of the traffic, that traffic is distributed through the hardware modules. Here each hardware module supporting the more number of subsets. In the software based implementation, the rules are classified by port or protocol, the groups of rules related to that port/protocol will be checked the packet which is to be transmitted. The above partitioning of the traffic will help to save memory and reduce the CPU usage. This is giving the information about the traffic alertness.

B. "Real world traffic analysis for Hardware module sizing":

Internet service provider can provide the information about the real world traffic. By consider the above information, we can analyze the traffic. According to the brief idea of the classification policies, it gives the quantitative analysis of how the traffic can be splits. By this we can determine the worst case results and decide the minimum criteria for the classification. For the required throughput, set the appropriate input rate necessities for sizing each SME hardware module. By using the adoptive algorithm, we can develop the hardware based methodology, relying on the traffic mix up, it is second-hand to enhance the software based Intrusion Detection System. But, we can't do such an analysis once for all system dimensioning, rather than we can go through the methodological advance. In the life time of the NIDS there are many changes can happen in the traffic mix and these variations are also depends on the particular operator's utilization. This does not occur in the practical case, the content matching engine rules can be updated or re run weekly once. But, in violations in the traffic mix up are very slow in order of several weeks. If any major violations in the traffic mix up are finds, it results the system to be restore for executing the synthesis with associate to the intermittent rule update.

C. "Hardware Module Implementation and Relevant tradeoffs":

For the specific SME we have performed many syntheses, for it gather insight in the up-and-coming area/speed tradeoffs for the exact rule set. If multiple copies of SMEs are used to achieve high throughput, the choice of the optimization of the engine related to speed and area is not distinctive, but it rely on which circuit to be designed and implemented. In some cases area/delay tradeoffs are quite unexpected. The use of low speed SMEs for multiple copies will allow the optimization of area of particular engine, greatest operating frequency, over all throughputs.

II. NETWORK INTRUSION DETECTION SYSTEM

Intrusion Detection System (IDS) is a hardware/software system. It controls the events/processes; these are appearing in the PC or network system. It analyses the events statically and dynamically for the compromise to security. As we know network security is very important in now days. So, we have to take care of our system from the hackers and anomalies. This issue can make the NIDS system to be more important in every organization or network system. Intrusion means "bypass the security mechanisms of network or computer". It is also defined as the attempts to compromise the confidentiality, integrity, availability.

By the working of their functions IDS can be classified /divided into three components

- Information Source
- Analysis
- Response

– Information Source: We need some sources to check whether an instruction has taken place or not. Example

host, network and application monitors these are the some common sources for IDS system.

- Analysis: In this selection IDS analyzed and well organize the events which are derived from the information sources. IDS decide those events indicate the intrusions. The most common analyzing approaches are the anomalies detection and miss-use detection.
- Response: It is the set of actions in which indicated those errors is been detected. This is typically divided into active and passive actions.
- Active actions: - In this if any hacking or any malware is attacked. It can be act or response for those actions and the responses are called as active response and those reports will be sent to the administrator or servers. Actions involve the errors detection and corrections.
- Passive Actions: If any hacking happened in the system, it gives the information about the hacking process to the administrator or the network server. It will help us to access the process later.

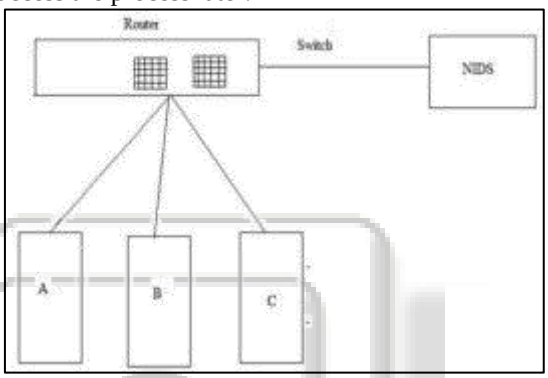


Fig. 2.1: Behavioral NIDS

From the above diagram, it can perform the detection of hacking which gives the security to our network system. In the above figure, if A is hacking the packets of C, it can be detected by the NIDS which is in connect through the switch. If "A" is sending more number of packets beyond of its behaviour. It can be detected by the NIDS and it can take actions on that device which has stolen the packets from the adjacent device or system.

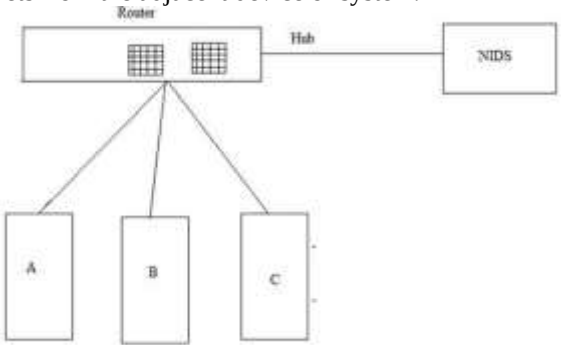


Fig. 2.2: Signature based NIDS

From the figure 3.2, it can detect the hacking it will secure our system from the hacking process. In the above diagram, all the packets are transport with the signature. It can check the signature and it will come to know that whether the packet is hacked or not. It can be identified by the use of signature only. This signature is unique for each packet. While packet transmission, this signature is used for the security purpose. The signature to be adds it to the packet.

A. String Matching Engine Synthesis

To synthesize the each module of the SME, it is implemented as five independent modules; thus, the synthesis of each module can be made independently and optimization is also possible. To identify the best area or speed trade-off a number of synthesizes are done for each module considering various speed constraints. In particular to have the better throughput reduced number of logic resources is used. The evaluation of the product of area and delay of each implementation is calculated in terms of the ratio among amount of LUTs, "the maximum operating frequency" and lastly consider the lowest area delay product. Depending on the difficulty in the synthesis process, we get different qualitative analysis.

1) Rule Header

The rule header states the criteria for matching with packet headers and the action to be carried out on finding a "match".The criteria may be of the source and destination IPaddresses, the protocol type and port numbers and, it checks whether the rule is bidirectional (<>) or unidirectional (->).

Alert is the most commonly used rule and it results in the generation of an alert signal and the incoming of the packet.

2) Rule Options

The rule *options* start up with the rule header and are enclosed with the pair of parentheses. The semicolon separates one or more options. A rule matches only if its header and all of its options are matched, i.e. "logical AND". The given below are some of the more important rule options, the full details of these rule options are to be found in the Snort manual:

3) Content

The "*content"* keyword states a fixed string to be found out for in the packet payload. This fixed string can be text or/and byte code. A single rule can consist multiple "*content* keywords", e.g.

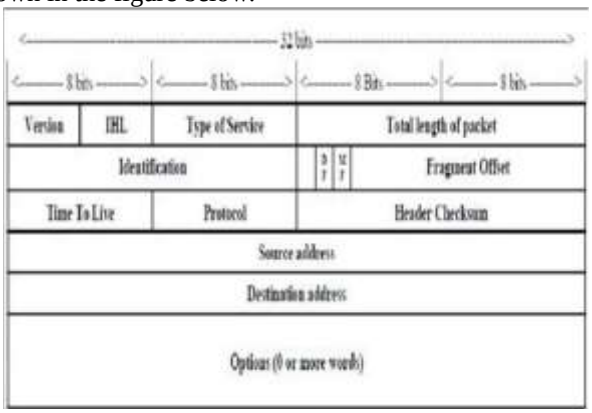
```
alert tcp any any -> any 90 (content: "Some string";)
alert tcp any any <> 10.1.1.0/24 80 (content:"|3c
ff|G|01|E";)
```

This is input patterns that are provided as input in the project.

For example "SNORT rule", the rule may be like "alert tcp any any -> any 90 (content: "f6f70536572766c65740000")" it would match all patterns, it has last 90 bits consist the value of "f6f70536572766c65740000"

Whenever the packet matches a rule, a process will be carried out as mentioned in the "*Logging/Alert"* signal;

The pattern of IP Header would be arranged is as shown in the figure below.



III. METHODOLOGY

The design flow of the project is shown below. It contains all the processes which involved in the methodology. In the Network Intrusion Detection System, involves the detection of viruses or malware present in the data. The data has specified with their bandwidth. If bandwidth is changed, it should be inform to the administrator or receiver end of the packets. The packets are classified under the information of the header content. If any changes in the header information then it classify as the other type of packets. Header information gives the destination address to the administrator.

A. Deep Packet Inspection:

It is carried out by the string matching engine. If all the bits in the string are matched then only the match signal gets high. It should carry out for all patterns of each packet. The string matching is carried out by using CAM which is helps to search location for the required data. By following this procedure the speed of the matching process will increases. Hence we can increase the speed of the design process.

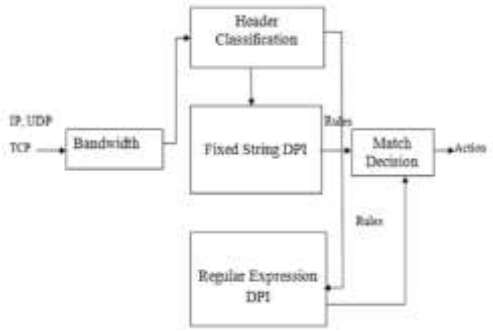


Fig 3.1 Block diagram of the methodology

1) Packets:

In networking there are many types of packets are available. These are namely, TCP (Transmission Control Protocol), FTP (File Transfer Protocol), SMTP(Simple Mail Transfer Protocol), UDP (User Datagram Protocol) etc.

2) Bandwidth:

The bandwidth information is present in the header part of the packet. If bandwidth or channel is changed, it should be specified.

3) Header Classification:

It depends on the data or content in the header of the packet.

Header has the above information, it will give details of IP address of source and destination and also it has port information. Depending upon this information the packets are classified and it will send to the destination side with the concern of traffic in the network. IP addresses are 32 bit integers which are represented in the dot based notation. The dot based notation is the decimal representation for each byte of the IP address.

Example, IP address with the Hex value 0X800A080B is represented by 128.10.8.11.

IP address consist of Host Id and Network Id

B. "Network Id":

Network address is a specific identified with host in the whole network.

C. "Host Id":

Host address is within network, it is specified protocol address.

Header classification is based on the data or information in the header of the packet. It contains the receiving end details, based on this we can classify the packets. By using this method we can increases the resource savings. Header information is same for all packets. If it is changed, bandwidth or channel

D. "Deep Packet Inspection (DPI)":

It is complete inspection and information extraction (Ix) is a form of computer, network packet filtering that examines the data part of packet as it passes an inspection point, searching for protocol non-compliance, viruses, spam or intrusions.

E. "String Matching Engine (SME)":

String matching engine gives the high throughput. It uses the less number of "logic resources". We can find out the area and delay for each implementation. It calculates the relation between "maximum operating frequency" and "the number of LUTs". By using above calculation us can elected the lowest product of area and delay. The synthesis of the process is difficult whenever we increase the circuit complexity, it effects on the outcome of the circuit. By using the lesser size SMEs the speed can increase and also we maintain the consumption of the logic resources. It differs the number of LUTs by 5%. In this way we can increase the speed and also improve the area/throughput trade-off.

F. "Snort Rules":

These are the rules are used to give the security to the system. The Snort rules are the important part in the network intrusion detection system. The network intrusion detection system gives the security by these rules. If these rules are satisfied then only we can tell as the system is the secured one.

G. "Fixed string DPI":

The deep packet inspection works on the fixed string method which is not changed in the inspection. The fixed string is used to give the security for the network against the malfunctions or errors. The fixed string deep packet inspection consists of the fixed string. It will not change in the inspection process in the network intrusion detection system.

H. "Regular Expression Deep Packet Inspection":

It has the regular expression which has to be update with the hacking techniques or error introduction method in the network. These regular expressions can weekly or monthly. Whenever the hacking techniques are changed, we have to update the rules which are to be failed in the network intrusion system.

E.g. A+B<10, A+B=5.

These two expressions are regular expression which is going to be change with the hacking techniques of the hacker. If we have the expression of A+B<10, then if it is not give the good quality of the service means, it will not give the security that is the hacker can also over come with this one so, we have to change the expression or update the rules with A+B=5.

I. "Match Decision":

If the rules are matched or not the string is matched in the system then it will gives the output high or 1 in the digital system. If will decide whether the data comes by the packet is secured or if any malfunctions or hacking takes place in the data it can be decided in this block. It is designed by the AND gate because, if both the data is fully matched then only it will give the high output.

J. "Actions":

If any malfunctions or hacking takes place in the system then it will detect the errors in the system, if it is the active actions then it will correct it.If it is passive action then it will gives the report to the server or administrator for further analysis. The string matching is carried by Serial Register Lookup method; it is matching the string with incoming packets in the system. In this method incoming data is compared with the 0 to 15 bits it is carried out by the counter which counts the number of bits and this is carried out in parallel which is to be increases the speed of operation and also it increases the system security. Here we are using the XNOR gate for our operation for matching the incoming data with the string.

IV. SIMULATION RESULTS

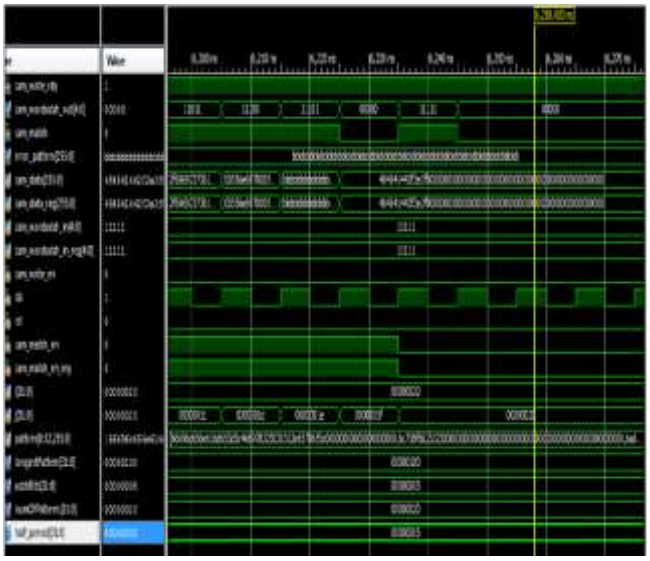


Fig. 5.11: Synthesis result of Error Detection

In the above synthesis result whenever write enable is high, here the comparison is carried out of the incoming data with the string pattern. If any error occurs, it can be find out in the single clock cycle. It will correct in single cycle. Here we are giving the data as the input and getting the address as the output.

Reference	241 MHz
Proposed	277.635MHz

Table 1: Comparison of the proposed method with existing methods

We are comparing operating frequency of previous and proposed work.

Logic Utilization	Used	Available	Utilization
Number of Slice Register	40	11440	0.099%
Number of Slice LUTs	1093	5720	19%

Table 2: Logic Utilization

V. CONCLUSION

This methodology gives significant advantages in terms of network security, which is achieved by string matching of the pattern with incoming packets. The speed of the string matching operation is improved in the design and implementation. By using the CAM in the design we can detect the error in the packets. The packet classification is based on the header information of the packet. Hence it could classify the packets with homogeneous address will save the resources in the system. The packet transition gives the information about the traffic present in the network. It helps us to control the traffic in the network.

REFERENCES

[1] "Snort: The Open Source Network Intrusion Detection System," Source fire, http://www.snort.org, 2013.
[2] Xilinx Website, http://www.xilinx.com/, 2013.
[3] S. Sinha, F. Jahanian, and J. Patel, "Wind: Workload-Aware Intrusion Detection," Proc. Int'l Conf. Recent Advances in Intrusion Detection, pp. 290-310, 2006.
[4] K. Papagiannaki, N. Taft, Z.-L. Zhang, and C. Diot, "Long-Term Forecasting of Internet Backbone Traffic: Observations and Initial Models," Proc. IEEE INFOCOM, pp. 1178-1188, 2003.
[5] P. Borgnat, G. Dewaele, K. Fukuda, P. Abry, and K. Cho, "Seven Years and One Day: Sketching the Evolution of Internet Traffic," Proc. IEEE INFOCOM, pp. 711-719, 2009
[6] A.V. Aho and M.J. Corasick, "Efficient String Matching: An Aid to Bibliographic Search," Comm. ACM, vol. 18, no. 6, pp. 333-340, June 1975.
[7] B.L. Hutchings, R. Franklin, and D. Carver, "Assisting Network Intrusion Detection with Reconfigurable Hardware," Proc. IEEE Symp.Field-Programmable Custom Computing Machine, pp. 111-120, 2002.
[8] J. Bispo, I. Sourdis, J Cardoso, and S. Vassiliadis, "Synthesis of Regular Expressions Targeting FPGAs: Current Status and Open Issues," Proc. Third Int'l Conf. Reconfigurable Computing: Architectures, Tools and Applications, 2007.
[9] J. Moscola, J. Lockwood, R.P. Loui, and M. Pachos, "Implementation of a C Content-Scanning Module for an Internet Firewall," Proc. 11th IEEE

