Distributed IDS using Reconfigurable Hardware

Ashok Kumar Tummala and Parimal Patel

University of Texas at San Antonio
Department of Electrical Engineering
San Antonio, TX 78249
parimal.patel@utsa.edu

Abstract

With the rapid growth of computer networks and network infrastructures and increased dependency on the internet to carry out day-to-day activities, it is imperative that the components of the system are secured. In the last few years a number of Intrusion Detection Systems (IDS) have been developed as network security tools, both in commercial and academic sectors. These systems have used different approaches to detecting unauthorized activity, and have given us some insight into the problems that still have to be solved.

While considerable progress has been made in the areas of string matching, header processing and detecting DoS attacks at network level, complete systems have not yet been demonstrated that provide all of the functionality necessary to perform intrusion detection at each host system there by securing the entire network. In this paper we are proposing the architecture of a ***D****istributed* ***I****ntrusion* ***D****etection* ***S****ystem (DIDS) for use in high-speed networks. The proposed DIDS has Host IDS component at each host that combines the above-mentioned functionalities along with the capability of collecting the events at the application level to look for any signs of intrusion at the network level. DIDS consists of Central IDS component which performs sophisticated processing to detect any signs of distributed attacks on the entire network and update rules in each host system.*

For high speed networks it can be difficult to keep up with intrusion detection using purely software approach without affecting performance of the system intended for designed application. It is essential to use hardware systems or software with hardware accelerators. The proposed DIDS is a custom hardware implemented on Field Programmable Gate Arrays (FPGAs). This move to customized hardware-based systems allows the introduction of higher degree of parallelism than might be possible in software at a reasonable cost. The key aspects of this system are flexibility and partial reconfigurability.

The nature of future attacks to the Internet's infrastructure is difficult to predict, and partial reconfigurability feature of FPGA will allow the system to be adapted to a constant change allowing the system to adapt to new threats.

1-4244-0910-1/07/$20.00 ©2007 IEEE.

1. Introduction:

Recently, the Internet has grown rapidly, and become one of the most important infrastructures of our life. Therefore, it is essential that it operates reliably. A factor that prevents a normal operation of the Internet is security threat. Intrusion Detection System (IDS) [2, 3, 4] is a widely used tool to contain security threats. An IDS [5] run constantly in a system background, monitors computer systems and network traffic, and analyzes traffic for possible hostile attacks originating from outside the organization and also for system misuse or attacks originating from inside the organization. System administrators rely on such tools to monitor and secure their network and systems. They detect inappropriate use or activity of a network or computer system by monitoring events and sending alerts when certain events, such as scanning network to determine connected computer systems occur.

1.1 Types of Intrusion Detection Systems:

Intrusion detection can be categorized in to different types [7] depending on many factors like type of information source, analysis strategy, time aspects, architecture and response.

The source of information for an IDS can be (i) audit trails (e.g. system logs) (ii) network packets, (iii) application logs, (iv) wireless network traffic or (v) sensor alerts produced by other intrusion-detection systems. Based on the *analysis strategy* IDS can be classified in to signature based or anomaly based detector. Signature based detection is performed by looking for well-defined patterns of attack that exploit weaknesses in the system. The attack patterns are usually referred to as the “signature" of an intrusion. This type of detection is also called “misuse detection". When the IDS identifies intrusions as unusual behavior that differs from the normal behavior of the monitored system, this analysis strategy is called anomaly detection.

Time aspects categorize IDS into real time detection or off-line detection. In real time detection, IDS respond in real time and in off-line detection IDS store the monitored data, analyze it and then signal the administrator on any sign of intrusion. The *architecture* of IDSs is used to differentiate between centralized IDSs that analyze the data collected only from a single monitored system and distributed IDSs that collect information from multiple monitored systems in order to investigate global, distributed and coordinated attacks.

IDS can be classified in to active or passive based on their *Detection response.* In active response, IDS reacts to the attack by taking corrective action (e.g. closing holes) or pro-active action (e.g. logging out possible attackers, closing down services). If the IDS only generates alarms (including paging security analysts) and does not take any actions, the response is called passive.

1.2 Desirable Characteristics of an IDS:

A comprehensive intrusion detection system should address the following issues [8] regardless of what mechanism it is based on:

- It must run continually with minimal human supervision.
- It must be fault tolerant by being able to recover from system crashes, either accidental or caused by malicious activity. Upon startup, the intrusion detection system must be able to recover its previous state and resume its operation unaffected.
- It must resist subversion. The intrusion detection system must be able to monitor itself and detect if it has been modified by an attacker.
- It must impose a minimal overhead on the system where it is running, to avoid interfering with the system's normal operation.
- It must be configurable according to the security policies of the system that is being monitored.
- It must be adaptable to changes in system and user behavior over time. For example, new applications being installed, users changing from one activity to another or new resources being available can cause changes in system use patterns.

As the number of systems to be monitored increases and the chances of attacks increase, following characteristics consideration is also desirable:

- It must be scalable to monitor a large number of hosts while providing results in a timely and accurate manner.
- It must provide graceful degradation of service. If some components of the intrusion detection system stop working for any reason, the rest of them should be affected as little as possible.
- It must allow dynamic reconfiguration, allowing the administrator to make changes in its configuration without the need to restart the whole intrusion detection system.

2. Hardware Based Solution

Software based IDS, often included in the operating system of a “firewall” node, evaluate the identifications of a packet directed towards the host or network to find intruders, but these systems are limited in speed by the sequential nature of the imbedded processors employed in the system. One alternative is to develop a hardware-based IDS. Hardware based IDS gives us more parallelism and flexibility in the data width.

Some of the disadvantages of software based IDS are:

- They continuously use additional resources in the system they are monitoring, even when there are no intrusions occurring, lowering system throughput.
- Because the components of the intrusion detection system are implemented as separate software processes, they are subject to tampering (Evasion attacks). An intruder can potentially disable or modify the programs running on a system, rendering the intrusion detection system useless or unreliable.
- Most of the IDSs, are platform dependent, i.e. the intrusion detecting sensors are designed keeping in mind the OS on which they are implemented.

Our work is an attempt to use the advantages of reconfigurable hardware (FPGA Technology) and provide solution to the above-mentioned problems.

The recent trend show that the FPGA technology is gaining its share and becoming implementation technology of choice among network application designers. It also reflects mature state of FPGA devices and advantages of using them for end applications are

- Ability to reprogram on fly (partial reconfiguration)
- Low development cost
- High Speed
- Scalability
- Parallelism (any data width possible)
- Secured Configuration
- Time to Market

Today's state-of-the-art FPGA technology allows designers to satisfy almost any demand for high-speed data processing and fast data transfers that is needed in networking applications. Dedicated FPGA resources are used to capture the network packets and store them in the vast on-chip memory resources. Another appealing characteristic of FPGAs is their ability to integrate multiple modules or a complete system on a single chip. This is a generic trend in today's integrated circuit design and is referred to as system-on-chip (SoC) concept. Other features of today's high end FPGAs include embedded processors (PowerPC, ARM), which can run a variety of real time operating systems (RTOS), embedded Ethernet Media Access Controllers(EMAC) and powerful serial links, which support implementation of a multitude of open-standards protocols like Gigabit Ethernet, Fibre Channel, RapidIO, Infiniband and similar.

3. DIDS Architecture

Figure 1 shows our proposed DIDS architecture. The two main components of our design are Host IDS and Central IDS.

The Host IDS captures and processes individual packets on a particular host. Packets needing further processing are sent to Central IDS. The Central IDS takes a collective decision based on the information collected from individual hosts and conveys its decision to all the hosts present on the network.

Figure 2 shows the information source of the Host IDS with respective to the OSI model. As shown in the figure, Host IDS module captures the network packets at the data link layer and also collects the system log files from the application layer via PCI interface.

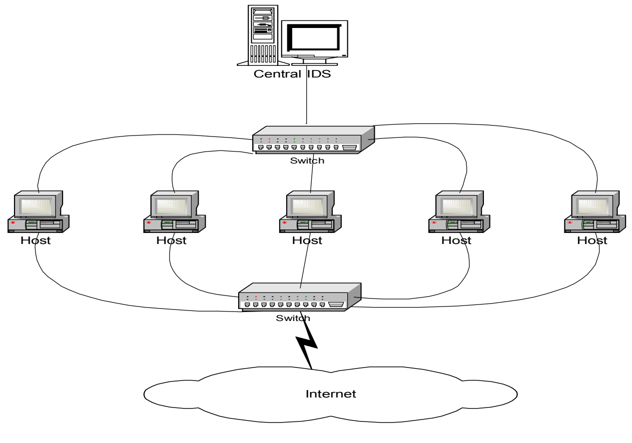

Figure 1: Overall Picture of our DIDS system

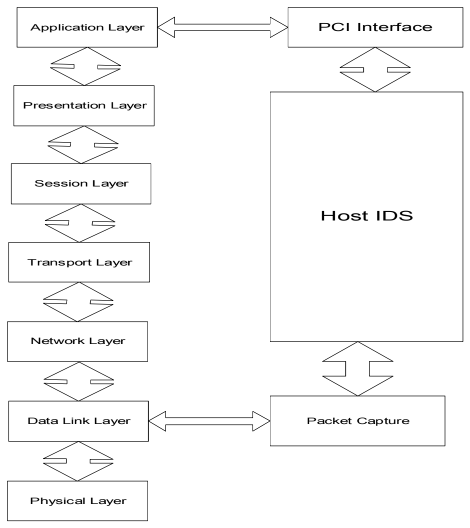

Figure 2: Implementation of Host IDS

3.1 Design Components:

Host IDS: The Host IDS module analyzes the traffic determined for its host. Figure 3 shows the block diagram of the Host IDS module. This Host IDS is hardware module, which will reside in all host or client systems.
Host IDS also limits the impact on host and offers limited opening to tampering and disruption of the actual IDS components. The main aim of this module is to perform complete per packet analysis. This means that this module looks at the individual packets coming from the network and performs header search, content search and also looks for DoS attacks or any signs of port scanning. This module also collects the attack information (audit trails or any application level event collector) from the application layer through PCI interface and further analyzes it. For strings that are present in more than one data packet, this module stores the payload data from the packet boundaries for that particular connection and looks for the complete string. For certain partial attacks like probing, DoS or DDoS, this module stores the partial threat information and those respective connection packets are sent to the Central IDS module along with the partial threat information for further processing. At Central IDS statefull analysis of these respective connections is performed and results are sent back to the individual Host IDS. The main components of the Host IDS are

- Packet Capture Unit (PCU)
- Header Search Unit (HSU)
- Content Search Unit (CSU)
- DoS Attack Detector (DAD)
- Response System (RS)
- Information Exchange Engine (IEE)

- Rule Update Unit (RUU)
- PCI Interface Unit (PCIIU)
- Log Analysis Unit (LAU)

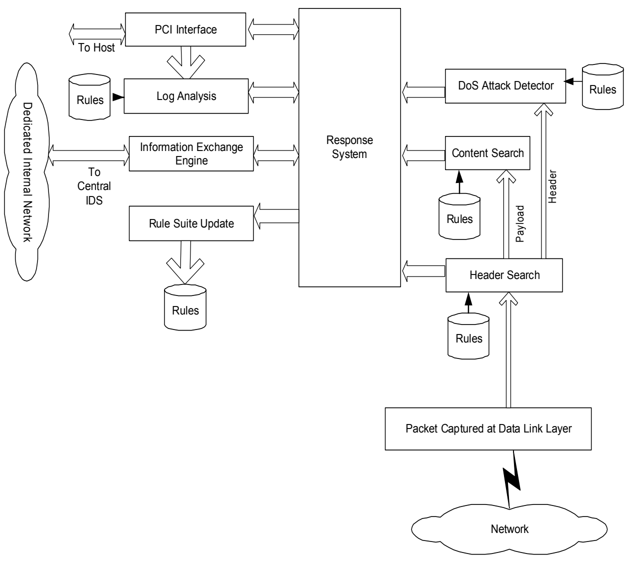

Figure 3: Host IDS

Packet Capture Unit (PCU): The role of PCU is to capture the packet from the data link layer. This captured packet contains the complete Ethernet packet along with the header information. Software tools available to capture the network traffic like in Ethereal or in software IDS like Snort uses packet driver libraries like WinPCap (Windows) or libpcap (UNIX) to capture the packet at the data link layer. Our PCU module is hardware module and makes our system usable on any host irrespective of platform (OS) it's been used.

Header Search Unit (HSU) [13]: HSU analyzes the header of the incoming packet with the rules defined in our rule database and the respective action is taken. Typical information interested in rule includes source IP, destination IP, source port number, destination port number. The obvious outcome of the analysis is to allow a packet or not to be forwarded. The results of search are sent to the response system.

Content Search Unit (CSU) [14, 15]: Once the packets are filtered based on header information, the content of the payload of the allowed packets are searched to find the presence of certain strings. These strings are predefined in our rule database. Strings of interest varying in lengths from one byte to sixteen bytes are searched. When certain strings are matched in the payload of the incoming packet, that respective information is sent to the response system. The response system then takes the respective action. This action can be something like changing the header rule database and not allowing the packets coming from that respective network and connection. The CSU will also supply the partial attack information to our response system, which will analyze that information and will send that information to the Central IDS for further processing.

CSU will also look for the strings which are distributed in two adjacent packets. Considering the memory buffer limitation only certain bytes of the payload from each packet will be stored. And these bytes can be from the boundaries of adjacent packets. The boundaries will depend upon length of the maximum string that can be detected.

DoS Attack Detector (DAD): The role of DAD module is to detect Denial of Service (DoS) attacks like SYN flooding. This module also looks for port scanning. This will have a complex state machine, which will track the TCP connections based upon their flags. When certain type of attack is detected that respective information is passed on to the response system. This module will also detect partial or suspicious attack patterns and that information will also be passed to the Response System, which can pass it to the Central IDS. The Central IDS can take a decision based upon the information collected from other hosts also.

Response System (RS): The Response System is the heart of the Host IDS. It analyzes the collected information from HSU, CSU, DAD, PCIIU and LAU, and based upon the analysis it takes the respective decision. This decision can be to update the rule database or forward that packet or state information or both to Central IDS through Information Exchange Engine.

Information Exchange Engine (IEE): The role of IEE is to communicate with the Central IDS. IEE's implementation depends upon the type of connection used to connect to the Central IDS. We are planning to connect all the Hosts to the Central IDS through Ethernet. So IEE can be an Ethernet Media Access Controller.

Rule Update Unit (RUU): The role of RUU is to update the rule databases of HSU, CSU, DAD and LAU. The RS directly communicate with this module for changes to be made to the rule database.

PCI Interface Unit (PCIIU): As shown in Figure 2, this module directly communicates with the Application Layer on the host PC through the PCI interface. Any suspicious activity at the application layer can be communicated to the Host IDS module through this interface. The application layer can tell the RS to make changes in the rule database or it can send the events of activities, which can be further analyzed using the Log Analysis Unit (LAU). RS can communicate with the operating system (OS) running on the host computer using PCIIU.

Log Analysis Unit (LAU): This unit will collect the events of activities from the host computer's application layer and analyze against the rules present in the rule database and send the respective response to RS. This module can be implemented in hardware or as an imbedded software application running on an embedded processor (like PowerPC present on Xilinx FPGAs).

Central IDS: The Central IDS module collects the information from individual hosts and further analyzes it. This information can be the state information of certain connections or complete packets belonging to certain connections. This module performs statefull analysis of the network connections that require further processing. It also collects information of partial or suspicious attacks from all the hosts and takes a collective decision about the attack. In such cases individual hosts are asked to make changes to their rule databases. The main function of this module is to reassemble the packets of particular connection and perform a content search for attacks which are distributed across several packets[17]. Since this kind of analysis takes for more than one packet, we are not bound by the constraint to perform our search within one packet latency. We will use the external memory resources to store our packet data and state information.

As discussed earlier this module also collects suspicious patterns for DoS attacks from individual hosts and then takes a collective decision. If the evidence is inconclusive and a broader search is warranted, neighboring Host IDS agents will cooperatively participate in global intrusion detection actions. Figure 5 shows the block diagram of Central IDS.

The main components of Central IDS are

- Global Information Exchange Unit (GIEU)
- Packet Reassembly Unit (PRU)
- Flow Database (FD)
- User Interface (UI)
- Global Response System (GRS)

Global Information Exchange Unit (GIEU): This unit collects the information from each individual Host IDS agent. As discussed earlier we are planning to use Ethernet to connect each Host IDS agent to the Central IDS. Host can send and receive the information as TCP packets (for reliable transmission) or as UDP packets (for faster transmission). Choice of the protocol will depend upon reliability or speed, and will be decided during implementation phase.

Packet Reassembly Unit (PRU) [16]: PRU reassembles the packets from Host IDS agents which require more sophisticated statefull analysis. Instead of the entire payload, data from the boundaries of the payload packets will be stored. And length of the maximum searchable string decides these boundaries. The reassembled packets can be stored in the on-chip memory present on the FPGA or external memory to the FPGA.

Flow Database (FD): FD maintains the records of certain suspicious DoS attack patterns. If needed information is collected from other hosts and conclusions are made for the attacks.

User Interface (UI): UI is like an administrative console. The administrator can keep track of activities and make changes to the individual Host's rule databases. Each Host IDS agent will pass the information to Central IDS for critical attacks from certain networks and based upon this information Central IDS can make changes to the rule databases of other hosts also.

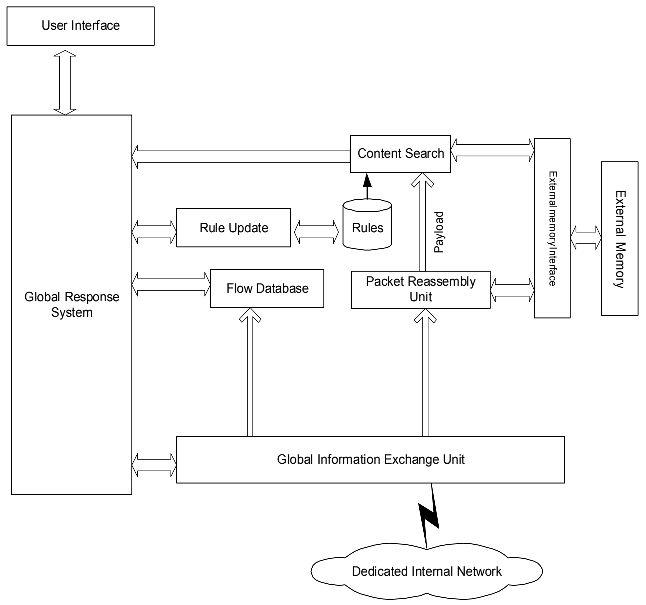

Figure 4: Central IDS

Global Response System (GRS): GRS is the heart of Central IDS and makes all the important decisions. This module collects the information from UI, FD and CSU, analyzes it and takes the respective decisions. The decisions are then communicated to each individual Host IDS.

5. Current Work and Implementation Results

Initial work suggests that using state of the art FPGA components and external memory, it is be possible to build a sizeable IDS. Using the latest available FPGAs we are able to perform per packet analysis (header search, content search and detecting port scans) separately at gigabit network traffics.

The header search unit, content search unit and port scan detector are developed independently and implemented using Xilinx's Virtex II Pro family's XC2VP20FG676-6 FPGA, which have over 564 user input/output pins, 9280 slices in which the logic can be implemented, 88 BlockRAMs where packets can be stored, and up to 8 DCMs to minimize clock skewing.

Table 1, 2 and 3 shows the FPGA resource utilization of the header search unit (HSU), content search unit (CSU) and the port scan detector (PSD).

# of Slices (9280 Total)	1038
BlockRAM (88 Total)	24
Achieved Speed	125 MHz

Table 1: FPGA resource utilization summary of HSU

# of Slices (9280 Total)	3566
BlockRAM (88 Total)	29
Achieved Speed	153 MHz

Table 2: FPGA resource utilization summary of CSU

# of Slices (9280 Total)	413
BlockRAM (88 Total)	6
Achieved Speed	125 MHz

Table 3: FPGA resource utilization summary of PSD

6. Conclusion

We have proposed architecture of a Distributed Intrusion Detection System (DIDS). In most of the existing DIDSs the role of the host IDS component is mainly passive i.e. they collect the events and forward to the Central IDS for processing. In our proposed architecture the main processing load at each host is taken care by the Host IDS component. With the increase in the network speeds there is a need for IDS with fast processing. And this fast processing can be achieved by using reconfigurable hardware like FPGAs, which give us the advantages of hardware along with option of flexibility in our system. Some of the FPGAs also have built in Multi Gigabit Transceivers (MGTs), which supports serial communication speeds up to 10Gpbs per channel. As complexity of our design will increase, it can be implemented on more than one FPGA and MGTs can be used to communicate between them.

References

1. Alfred J. Menezes, Paul C. Van Oorschot, and Scott A. Vanstone. Handbook of Applied Cryptography, CRC Press 1997.
2. G. White, E. Fisch, and U. Pooch. Cooperating security managers: A peer-based intrusion detection system. *IEEE Network*, 10(1): 20-23, 1994.
3. Harry Katzan, Jr..The Standard Data Encryption Algorithm,. Petrocelli Books, Inc. 1997.
4. William Stallings. Cryptography and Network Security: Principles and Practice. 2nd ed., Prentice-Hall, Inc. 2000.
5. John Carroll. Computer Security. 3rd ed., Butterworth-Heinemann 1997.
6. R. Bace and P. Mell. Intrusion Detection System. *NIST Special publication on Intrusion Detection Systems*, August 16 2001, http://csrc.nist. gov/publications/drafts/idsdraft.pdf.
7. W. Stallings. Network Security Essentials: Applications and Standards. Prentice Hall, pp. 280-303. 2000.
8. Aleksandar Lazarevic, Levent Ertöz, Vipin Kumar, Aysel Ozgur, Jaideep Srivastava. A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection. SDM_2003
9. Sandeep Kumar. Classification and Detection of Computer Intrusions. PhD thesis, Purdue University, West Lafayette, IN, 1995. URL:ftp://coast.cs.purdue.edu/pub/COAST/papers/sandeep-kumar/kumar-intdet-phddiss.ps.Z.
10. Anderson, James P. Computer Security Threat Monitoring and Surveillance. *James P. Anderson Co*., Fort Washington, Pa., 1980.
11. D. E. Denning. An intrusion-detection model. *IEEE Transactions on Software Engineering*, Vol. SE-13(No. 2):222-232, Feb. 1987.
12. Heberlein, L. et al. A Network Security Monitor. *Proceedings of the IEEE Computer Society Symposium, Research in Security and Privacy*, May 1990, pp. 296-303.
13. Steven R. Snapp, James Brentano, Gihan V. Dias, Terrance L. Goan, L. Todd Heberlein, Che lin Ho, Karl N. Levitt, Biswanath Mukherjee, Stephen E. Smaha, Tim Grance, Daniel M. Teal, and Doug Mansur. DIDS (distributed intrusion detection system) - motivation, architecture, and an early prototype. *In Proceedings of the 14th National Computer Security Conference*, pages 167-176, Washington, DC, October 1991. URL http://seclab.cs.ucdavis.edu/papers/DIDS.ncsc91.pdf.
14. Parimal Patel, Ashok Kumar Tummala, Hari Babu Ramineni, Wei-Ming Lin. Improved Design and Implementation of Network Intrusion Detection System for Gigabit Network Traffic Using FPGA. *CAINE 2005*: 382-386.
15. Sumit Vora. Architecture for Content Driven Packet Payload Processing Engine. M.S Thesis, University of Texas at San Antonio, August 2005.
16. M. Attig, S. Dharmapurikar, and J. Lockwood. Implementation results of bloom filters for string matching. In *IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM)*, Napa, CA, Apr. 2004.
17. M. Necker, D. Contis, and D. Schimmel. TCP-Stream Reassembly and State Tracking in Hardware. In *IEEE Symposium on Field-Programmable Custom Computing Machines (*FCCM) 2002 Poster, Apr 2002.
18. S. Li, J. Toressen, and O. Soraasen. Exploiting stateful inspection of network security in reconfigurable hardware. In *Field Programmable Logic and Applications (FPL)*, Lisbon, Portugal, Sept. 2003.