Chapter 1

Introduction

The entire world is networked together nowadays. With the development and popularization of the internet, more and more people highly depend on network to share information and exchange data almost instantly. People use internet to accomplish daily work, information gathering, online shopping and even electronic banking transaction. When sensitive information is now online, several security issues have been raised. Electronic Banks should be aware of the theft of the customer's credit card number. Insurance company should prevent the customer's private information from illegally accessed. Unfortunately, more than a quarter of a million computers are infected by the malicious code.

1.1 What is Malicioius Code

Malicious code comes in a broad variety of forms and is distributed through an unbelievable speed. Generally speaking, malicious code is a software that disrupt the normal operation of a computer. This software often executes without the user's consent. Once the computer is infected, an attacker could take control of the machine remotely, steal data and use the computer to attack others.

1.2 Types of Malicious Code

Malicious code comprised of various types intrusion code that causes problems to computers. Moreover, malicious code can be catagorized into two types according

to the reliability on host programs — Dependent and Independent. Dependent malicious codes need host programs to infect the systems, Back Doors, Logic Bombs, Trojan Horses and Viruses, for instance. Independent malicious codes are self contained programs without attaching on host programs. Worms is the best example of independent malicious code.

- Back Doors
  A back door is a secret entry point within the program that allow someone already know the back door to directly access your computer. Back doors have been used by programmers to debug the program. Unfortunately, immoral programmers may use the back door to gain unauthorized access.

- Logic Bombs
  Logic Bombs are embedded in programs that execute when certain conditions are encountered.

- Trojan Horses
  Trojan Horses are programs that pretend to be useful programs. Actually, they contain hidden codes that perform harmful function to the system. Trojan Horse programs may send your password or data to the attacker via the internet.

- Viruses
  A virus is a program that attaches itself to other programs and attacks applications while replicating itself in the process. A virus might rapidly infect every application files on an indivisual computer, but it does not intentionally try to spread itself from computers to computers.

- Worms
  Worms are much like viruses in replicating themselves. Actually, a worm is a program that propagates itself over a network, reproducing itself as it goes. Therefore, instead of spreading from files to files within the same computer, worms spread from computers to computers through internet. Generally, worms will search for other systems to infect by examing their system address. Then, the connection between host system and remote

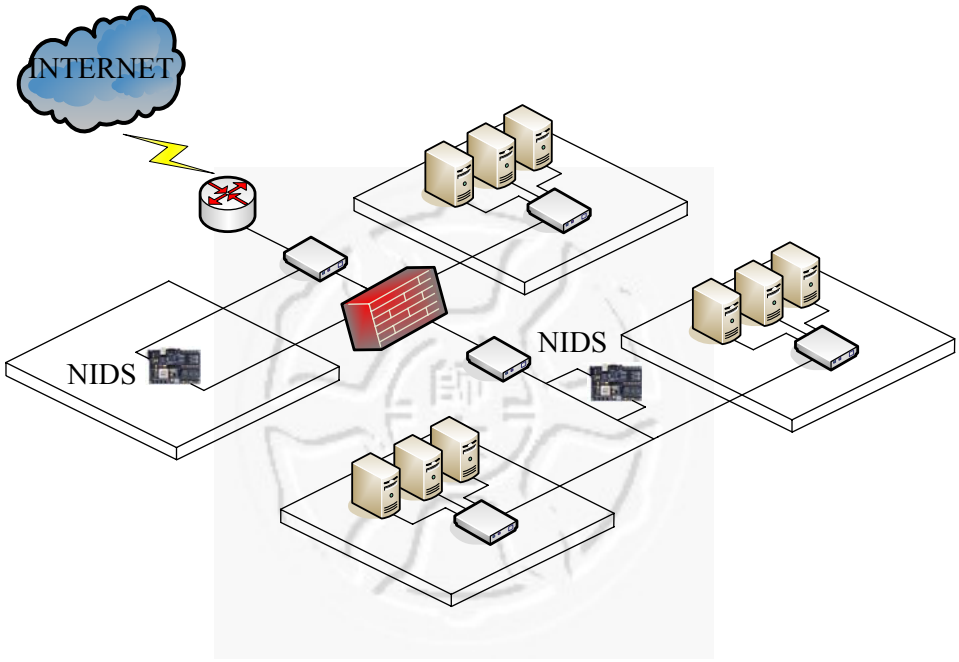

Figure 1.1: Network Intrusion Detection System

system is established. Worms will copy itself to the remote system and trigger the copy to be run.

1.3 Network Intrusion Detection System

Nowadays, restricted function of firewall resides in billions of commercial networks. Simple firewall owns the capability to only examine the header of the packet to determine whether to block or allow packet passages. Firewall can only check the port and IP address from the packet header corresponding to the well defined rules, but firewall cannot detect attacks when they happen. Unfortunately, it is not sufficient for high security commercial company. To protect an organization completely, therefore, it is necessary to provide a second line of defense. Hence, what we need is the Network Intrusion Detection System (NIDS). NIDS monitor network traffic in real time, examining packets in detail to sniff attacks before the packets reach their destination. NIDS are designed to not only inspect headers but also the payload of the packets. NIDS match the payload of

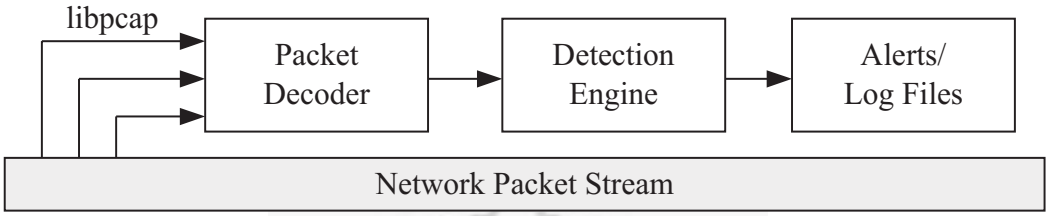

Figure 1.2: Snort Architecture

the packets from a rule set database to determine whether the packet is malicious or not. Moreover, the rule set database is updated periodically by the vendors as new attacks are digested. When suspicious activity is discoverd, NIDS will raise alerts immediately to the system administrator.

One popular type of NIDS is called Snort[1]. Snort is a packet sniffer and logger used as a network intrusion detection system. Figure 1.2 gives an overview of Snort. Snort relies on a packet capture library called libpcap to sniff the packet. After gathing the packet from the network line, they are sended to the packet decoder. The packet decoder interpret the specific protocol elements into the internal data structure of Snort. When the packet decoder procedure is finished, packet data is passed to the detection engine. The detection engine checks the packet according to the rule set database. These rules must be compared with the packet content to determine whether the packet is suspicious or not. If the packet is malicious, the system will generate an alert and write the attacked message into log files.

Snort contains numerous set of pre-defined rules to detect intrusion activities. Each Snort ruleis divided into two fields, the rule header and the rule option. The rule header describes the rule's action, protocol, source and destination IP addresses, and the source and destination ports information. An example of a Snort rule is demostrated in Figure 1.3. In this Snort rule, it looks for a TCP packet with any source IP and port 110, any destination IP and port. In order to match this rule, packet content must contain pattern "i love you".

alert tcp any 110 -> any any (msg:``Virus – Possible MyRomeo Worm``; flow:established;content:``i love you``;classtype:misc-activity;sid:726;)

Protocol	TCP
Source Address	Any
Source Port	110
Destination Address	Any
Destination Port	TCP
Payload Data	``i love you``

Figure 1.3: Snort Rule

1.4 Motivation

The Snort system running on general purpose processors can only achieve up to 60 Mbps [10] throughput because of the memory hierarchy. When the network card receives the packet from the internet, the system reassembles the series of bits to form a packet and forwards the packet for processing to data link layer. Inside the data link layer, the system will remove the data link layer header and pass it to the next layer. The processing continues until the original packet arrived the application program. After that, the packet will travel through PCI bus, memory, caches and CPU. Then, CPU will deal with the high computational string matching task. This is what the problem is! Inside the memory hierarchy, each previous cascading component is 10 times slower than the following component. Hence, the whole task is therefore a I/O bound operations. Since current high-speed network can achieve up to several Gbps, malicious packet may be dropped and cannot be detected if we still use the software based on general purpose processors. In order to accelerate the speed for intrusion detecion, various hardware based approaches have been proposed. Hardware is extremely suitable for implementing the system since the hardware speed is much more faster than software and the hardware can exploit parallelism for string matching. Using Application Specific Integrated Circuit (ASIC) to implement the circuit might be

a good idea because of the high performance of the chip. Unfortunately, ASIC lacks the characteristic of flexibility. Therefore, Field Programmable Gate Array (FPGA) is the most appropriate approach for both compromising performance and flexibility.

One popular approach for FPGA is established by regular expression which leads to low area cost and moderate throughput. Regular expression approach [5, 7, 9] requires representing as a finite automata and then translating them directly to FPGA circuit. In the regular expression method, efficient parallelism is difficult to implement since only one character is scanned at a time. Another approach to FPGA-based NIDS is the use of content addressable memory (CAM) [8, 12]. By the exploitation of CAM, pattern matching of the NIDS can be significantly accelerated with parallel processing. That is, multiple characters per cycle can be processed by utilizing CAM. This may effectively increase the throughput with higher area cost. Those approaches are discussed in detail in Chapter 2.

1.5 Scope of the thesis

The objective of this thesis is to present a novel FPGA implementation approach applied to Snort attaining both high throughput and low area cost. The proposed architecture has been prototyped and simulated by the Altera Stratix FPGA.

This thesis proposed three different sort of architecture for efficient and high-speed string matching. ROM-based architecture is introduced in Chapter 3. In Chapter 4 and 5, Bit-map encoding architecture and Partial Encoding architecture are presented respectively. Each approach is evaluated in accordance with the throughput and area cost. Moreover, the merits and the drawbacks of each architecture are discussed in the thesis. Possible improvements and alternative approaches are also considered. Finally, a brief conclusion of the thesis is discussed in Chapter 6.