私立東海大學資訊工程與科學研究所

碩士論文

指導教授：呂芳懌

入侵預防與遠端保護系統

Intrusion Prevention and Remote Protection System

研究生：李志揚

中華民國 九十八年 六月

摘要

近年來，許多人透過網站來獲取有用的資訊、收發電子郵件、購物等等。網路對我們日常生活已經是不可或缺的。在我們利用網路方便的同時，有許多威脅，像是阻斷攻擊或分散式阻斷攻擊，來破壞網路系統，導致企業或公司財務上的損失。目前有許多入侵偵測系統發展出來保護網路系統，但有時候面對大量的攻擊時，難免會失去它們的偵測能力。

本研究提出一個入侵預防系統，利用 Cumulative Sum 演算法來偵測從本地端或遠端的攻擊。此系統不僅可以偵測外出網域與轉送的封包來保護遠端重要的系統，也可以偵測進入網域與送往相同網域的內部封包來保護自身網域的安全。實驗結果顯示此系統在防禦環境下可以帶來更高的安全性。

關鍵字：阻斷攻擊，分散式阻斷攻擊，入侵偵測系統，累計式總和，內部攻擊，入侵預防

Abstract

In recent years, networks are essential particularly for our daily life. More and more people access useful information, receive e-mail, purchase high-tech products, etc., through websites. However, when we enjoy network convenience, networks on the contrary also conduct threats for us, like Denial of Service (DoS) and Distributed Denial of Service (DDoS), resulting in bringing us inconvenience or financial loss, e.g., enterprises or companies' huge amount of financial loss or missing their business opportunities. IDSs can protect network systems. But they often suffer from losing their detection effectiveness and capabilities when processing enormous network traffic. In this article, we proposed an intrusion prevention system, named Cumulative-Sum-based Intrusion Prevention System (CSIPS) which detects malicious behaviors, attacks and distributed attacks launched to local and remote servers/hosts based on intrusion detection techniques and Cumulative Sum (CUSUM) algorithm. Experimental results show that CSIPSs can carry out a higher security level for a united defense environment.

Keyword: DoS, DDoS, IDS, CUSUM, Inner Attack, Intrusion Prevention

Table of Contents

List of Tables

List of Figures

Chapter 1. Introduction

Recently, networks have brought us a convenient environment for data access and communication. Many people communicate with others and /or retrieve data through networks almost everyday. However, when people exploit convenience provided by networks, more and more threats are now threatening networks, e.g., hackers issue Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks to destroy a system, or make victims unable to work properly, resulting in an enterprise or a company' huge amount of financial loss or missing its business opportunities. Many famous attack packages provide friendly user interfaces and can be conveniently downloaded from the Internet. That is why hackers can very often easily launch attacks, even they are naïve attackers.

In recent years, Intrusion Detection Systems (IDSs) have been developing quickly. But few Distributed Intrusion Detection Systems (DIDSs) have been proposed since their architectures are always complicate, and to distribute their functions to each component is a hard work. A DIDS often employs many distributed components to individually deal with detection tasks so bandwidth consumption problem can be reduced and avoided, respectively. However, complicate attacks have been newly developed day by day. An IDS/DIDS can not detect all kinds of attacks. In addition, IDSs /DIDSs only detect attacks passively. An intrusion prevention system [22] has accordingly been proposed to solve this problem. Wuu et al. [6] proposed an approach to detect suspicious packets at the source end. Their system checks and adopts actions at the victim side. However, this system does not detect attacks real time. Leu et al. [13] proposed an approach to build a profile for each user by recording his/her usage habits as forensic features. They use data mining techniques to check to see whether the user's current inputs carry malicious behaviors or not. Further, many researches [4][14][15][16] adopted CUSUM algorithm to detect DoS and DDoS attacks. They claimed that CUSUM can effectively detect these types of attacks.

In this article, we proposed an intrusion prevention system, named Cumulative-Sum-based Intrusion Prevention System (CSIPS) which detects malicious behaviors, attacks and distributed attacks based on Cumulative Sum (CUSUM) algorithm [7]. CSIPS can also detect inner attacks and attacks launched toward remote hosts that are collaboratively protected.

The contributions of this article include 1). collaboratively protecting remote networks or hosts in advance to prevent them from being damaged or destroyed by DoS and DDoS attacks; 2). detecting inner attacks in which attackers and victims are users of the same subnet, or different subnets but served by the same router.

The rest of this article is organized as follows. Section 2 introduces IDS systems and the Cumulative Sum algorithm. Section 3 describes CSIPS framework and its detection algorithms. Experimental results are shown and discussed in section 4. Section 5 concludes this article and addresses our future research.

Chapter 2. Background and Related Work

2.1 Cumulative Sum (CUSUM) Algorithm

The CUSUM is able to detect something that sharply but continuously increases. Some of its assumptions are given below. First, let X_n be the packets collected by IDS within a sampling time Δn and $\overline{X}$ the mean value of random sequence X , $X = \{X_n, n = 0,1,2...\}$. Second, let $Z = \{Z_n \ , n = 0,1,2...\}$ with α, where $Z_n = X_n - \alpha$ and α is the peak value of normal traffic. Hence, all elements of Z are negative or zero that makes $\overline{Z}$ become negative.

When a change, such as a flooding-based attack, occurs, Z_n will suddenly become positive, as illustrated in Figure 1. $Z_k \geq \overline{Z} + h$ indicates the moment an attack may be starting, where k is the smallest n and h the threshold of abnormal network traffic. Δ_k is then considered to be the change point. $y_{n-1} + Z_n \leq 0$ shows there is no attack. The CUSUM accumulates Z_n, $n \geq k$, with formula (1), which is the recursive version of the non-parametric CUSUM algorithm [11].

$$y_n = (y_{n-1} + Z_n)^+, y_0 = 0 \qquad (1)$$

where $x^+ = x$ if $x > 0$ and 0 otherwise. $Z_n, n > k$, may now be positive or negative.

The decision function at Δp, e.g., $d_p(y_p)$, is as follows.

$$d_p(y_p) = \begin{cases} 1 & \text{if } y_p > N \\ 0 & \text{else} \end{cases} \quad (2)$$

where N is attack threshold and '1' indicates there is an attack.

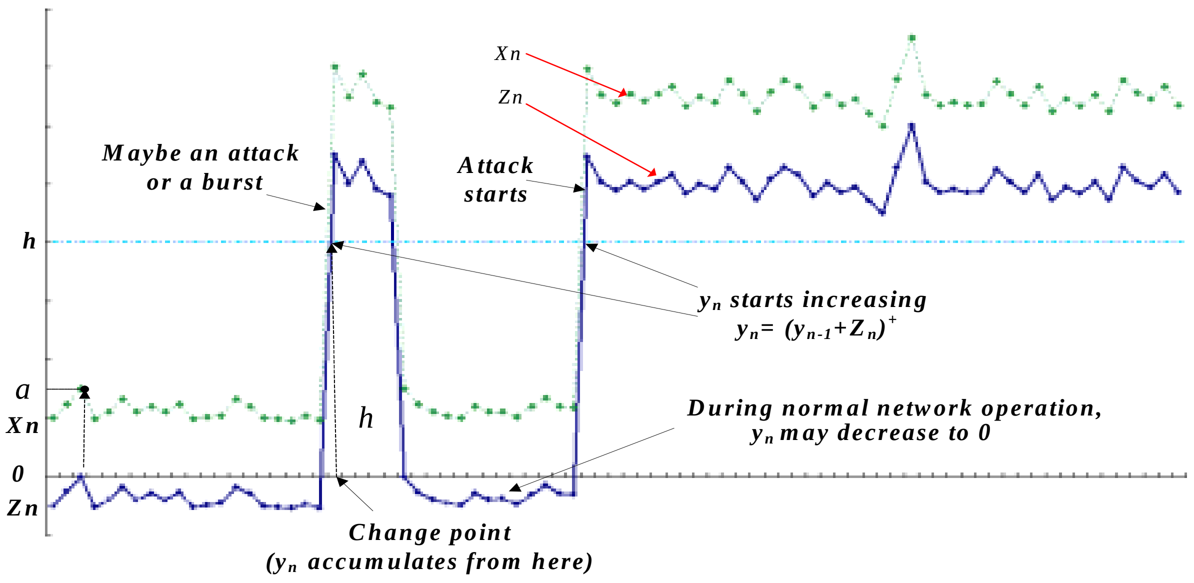

Figure. 1 The behavior of the CUSUM

2.2 Intrusion Detection Systems

IDSs play an important role in network security. When hackers intend to invade, penetrate or attack a system, an IDS can detect the malicious behavior and inform administrators to adjust detection policies for IDSs, or even directly request firewalls to resist the malicious behaviors/connections. A traditional IDS is a centralized detection system which has single point fault problem and poor scalability [2]. Nowadays, few distributed IDSs have been proposed [10][20][21], because they need to employ many resources to deal with attack detection, and their architectures are often complicate. However, they often provide better detection performance. Generally, IDSs can be classified by the following ways.

(1) According to location:

A Host-based IDS (HIDS), installing in a computer, collects audit data and analyzes the data to see whether there is an attack. A Network-based IDS (NIDS), installing on the throat point of a network management unit (NMU), monitors network traffic flowing through the management unit. When finding abnormal events, it notifies administrator to respond appropriately. In recent years, hybrid IDSs, which mixes HIDS and NIDS, have been widely deployed to protect networks [19]. Further, many hackers invade a Grid system to exploit the Grid's abundance of resources to launch attacks. So Grid-based IDSs (GIDSs) [5] have also attracted researchers attention.

(2) According to detection methods

Mostly, there are two methods to detect attacks, misuse and anomaly detection. The former is performed by comparing known signatures collected in databases with a user's current behaviors. The latter detects whether a user's current behavior is different from an ordinary user's normal behaviors, and checks to see whether abnormal conditions are over a predefined threshold. If so, an attack is suspected. For a higher detection rate, network administrators often mix the two methods [17][18].

2.3 Intrusion Detection Techniques

Many studies use hierarchical architecture and agent platform to detect attacks. [8][9][10][11] employed mobile agents to improve efficiency of DIDS because mobile agents have many advantages [11], like overcoming network latency, reducing network load, autonomous execution, platform independence, dynamic adaptation, static adaptation and scalability. Other detection methods, e.g., PGIDS [3] and AAFID [1], have been proposed and employed to achieve specific levels of security and detection capability. In PGIDS [3], packets are sent to IDSs according to the IDSs' detection capabilities, and an IDS checks packet statistics to detect attacks. [4][14][15][16] implemented CUSUM algorithm which cumulates number of packets, and then compares the number with predefined threshold to detect DoS and DDoS attacks.

Chapter 3. System Framework

For each autonomous network management unit [3], such as an enterprise Intranet and a campus network, we employ a CSIPS as the security system. Figure 2 shows the configuration. A CSIPS monitors packets of a system to detect in-bound, out-bound, forwarded and inner malicious behaviors which include DoS, DDoS and/or ordinary attacks. The latter is also called logical attacks. In addition, neighbor, adjacent and/or nearby CSIPSs cooperatively detect DoS/DDoS attacks. They together form a united defense environment [3][12]. That is, a CSIPS not only protects its own network management unit, but also helps to detect malicious behavior launched toward a remote network management unit which is a member of a united defence environment. The purpose is more effectively protecting a network system.

In a network management unit, we use switches that have mirror ports as the packet duplication components to collect data. When packets flow through a switch, the switch duplicates the packets, sends the original packets to their destinations and delivers the duplicated packets to CSIPS. A CSIPS on receiving the duplicated packets classifies them into 1). in-bound packets which are packets duplicated from those coming from outside world of the underlying network management unit; 2). out-bound packets which are packets duplicated from those sent to the outside world of the underlying management unit by hosts in the management unit; 3). forwarded packets which as shown in Figure 2 are packets duplicated from those sent by hosts in a network management unit other than underlying one to the third network management unit; 4). inner packets which are packets duplicated from those sent between two hosts that belong to a subnet or two subnets served by a router. A CSIPS individually monitors the four types of packets to detect attacks. The purpose of detecting malicious behaviors on duplicated packets is to avoid degrading delivery performed of original packet

In this study, switches based on their functions and locations in a network management unit are classified into type-1 and type-2. A type-1 switch, denoted by S1 shown in Figure 2, is placed on the link connecting an edge (a border) router to another network management unit. A type-2 switch, denoted by S2, is located on the link connecting an edge router (a border) to one of its subnets.

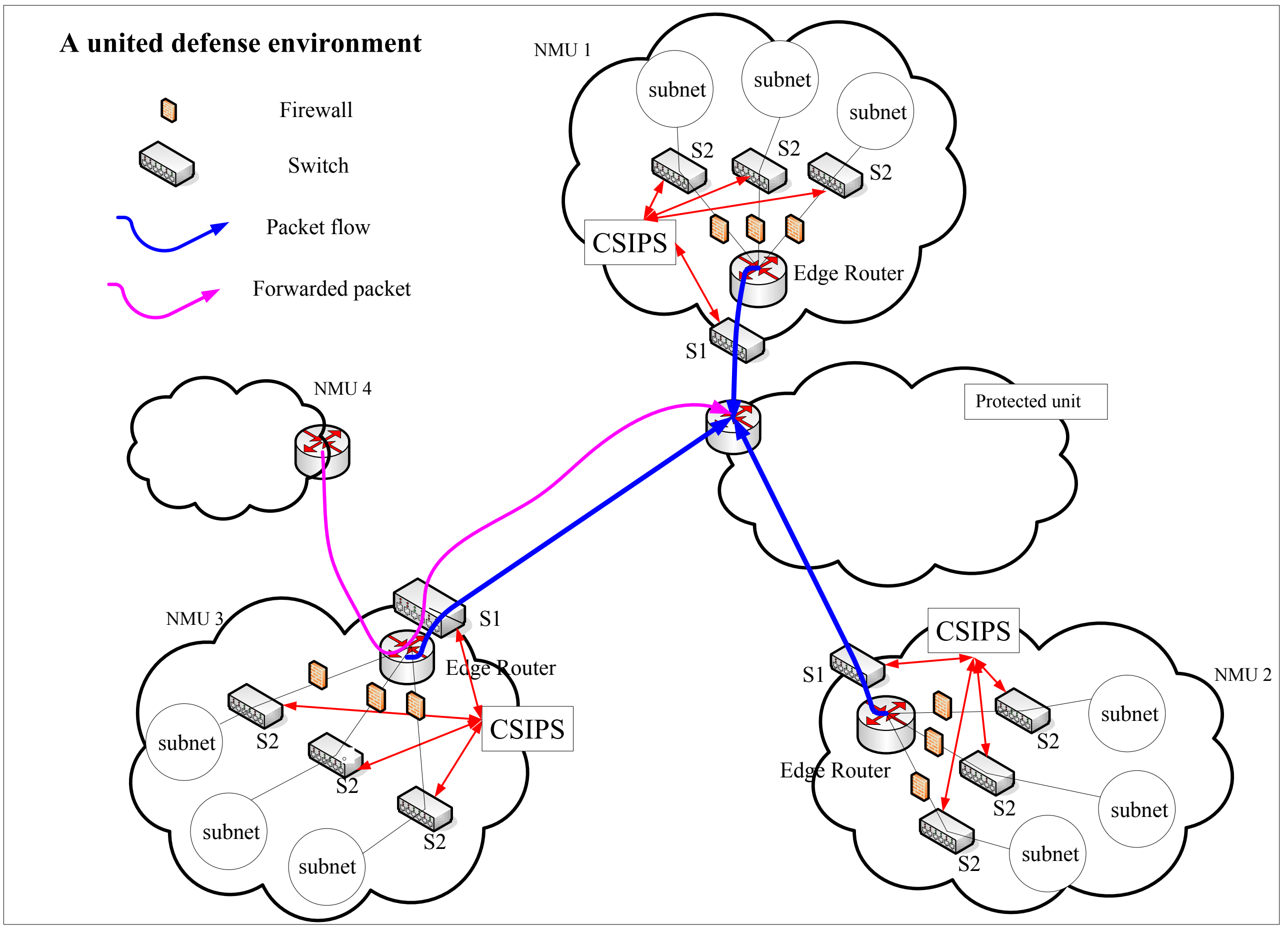

Figure 2. Overview of CSIPS and a united defense environment

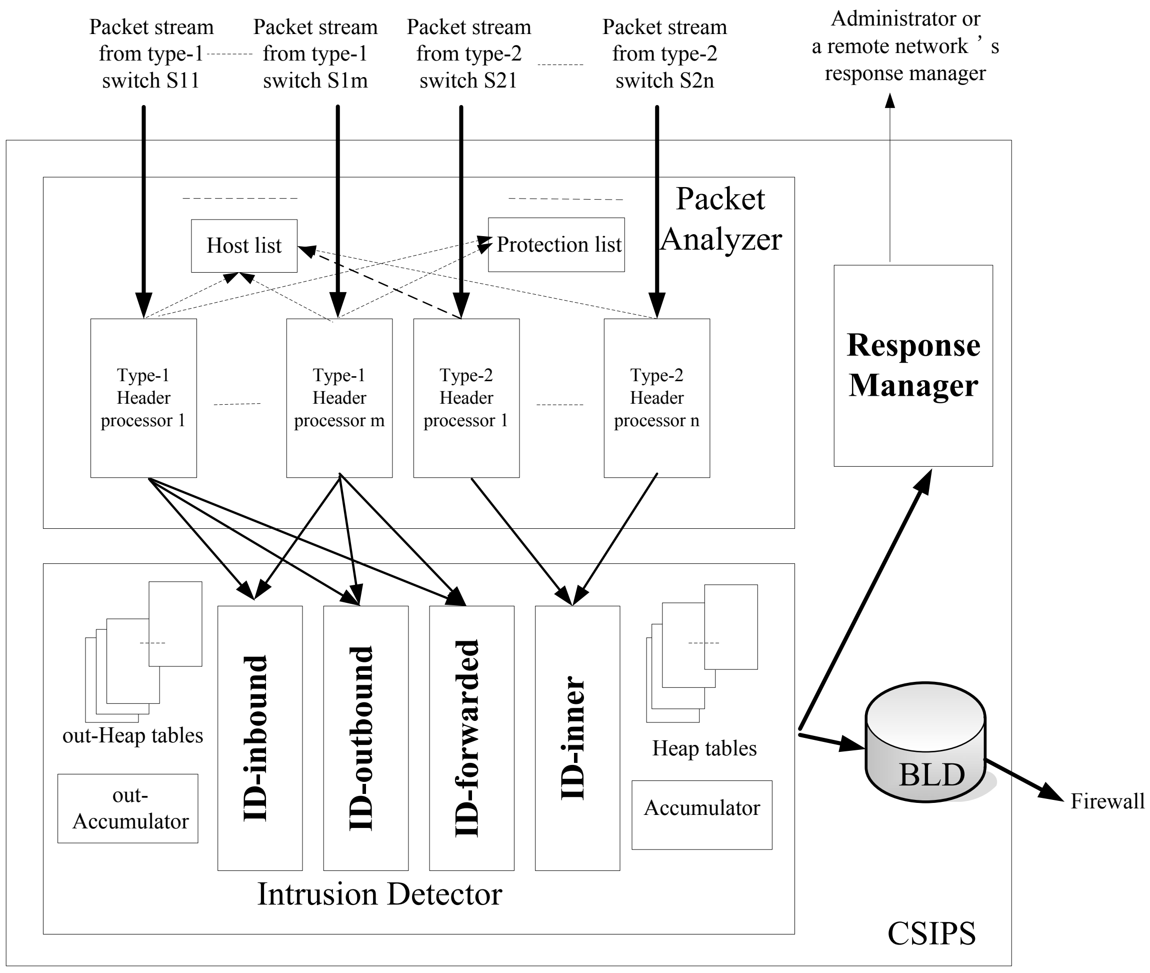
Packet stream from type-1 switch S11
Packet stream from type-1 switch S1m
Packet stream from type-2 switch S21
Packet stream from type-2 switch S2n
Administrator or a remote network ' s response manager
Packet Analyzer
Host list
Protection list
Type-1 Header processor 1
Type-1 Header processor m
Type-2 Header processor 1
Type-2 Header processor n
Response Manager
out-Heap tables
out-Accumulator
ID-inbound
ID-outbound
ID-forwarded
ID-inner
Heap tables
Accumulator
BLD
Firewall
Intrusion Detector
CSIPS

Figure 3. The CSIPS architecture

A CSIPS as shown in Figure 3 consists of Packet Analyzer, Intrusion Detector, Response Manager and Black List Database (BLD). Packet Analyzer is directly connected to mirror ports of type-1 and type-2 switches. On receiving a packet it analyzes the packet header and classifies the packet into one of the four types, in-bound, out-bound, forwarded or inner. Intrusion Detector, taking charge of packet analysis, packet detection and notification, is composed of four detection subsystems , including ID-inbound, ID-outbound, ID-forwarded and ID-inner, which as their names receive corresponding classified packets from Packet Analyzer to detect corresponding malicious behaviors. Once an attack is detected, e.g., by subsystem X, X sends an alerting message to Response Manager, which on receiving the message may either alert local administrator to response properly, or notify the protected remote network's CSIPS. Black List Database not only records hackers' IPs and their intrusion information so local firewalls can accordingly discard packets sent by known hackers, but also keep packet statistics received from the ID-inbound and ID-inner subsystems on which hackers' malicious behaviors can be observed, e.g., by deploying data mining techniques [23][24], on a long-term base.

3.1 Packet Analyzer

As shown in Figure 3, Packet Analyzer is composed of type-1 Header processors, type-2 Header processors, Protection List and Host List. Host List as shown in Figure 4 consists of three fields, IP address, MAC address and subnet ID, which together are used to records address information for local hosts protected by a CSIPS. Hosts in a network management unit are grouped as subnets which can be identified by their subnet address, e.g., 140.128.101.XX. In other words, a tuple in the list retains information concerning a host or a subnet. Protection List keeps IP addresses of protected remote hosts, subnets and network management units. Header processors on receiving packets sent by switches retrieves K fields from each packet header, including Source IP, Destination IP, Protocol, TTL …etc, and classifies them into the four classes by comparing the packets' source IPs and destination IPs with Host List. Table 1 lists the classification.

A type-1 Header Processor is connected to a type-1 switch to pick up in-bound, out-bound and forwarded packets. Type-2 Header Processors are connected to type-2 switches to filter inner packets. After classifying a packet, a Header Processor sends the packet to its corresponding intrusion detection subsystem. Header Processor algorithms are as follows.

Algorithm1: type-1 Header Processor /*distinguishes in-bound, out-bound and forwarded packets*/

```
Input: A packet P sent by a type-1 switch; Host List; Protection List
Output: P is classified into in-bound, out-bound, or forwarded
{if (P' source IP is in Host List) /*P is an out-bound or inner packet*/
   {if (P' destination IP is in Protection List) /*P is an out-bound packet*/
      P is sent to ID-outbound subsystem to detect malicious behavior;}
else /*P is an in-bound or a forwarded packet*/
   {if (P' destination IP is in Host List) /*P is an in-bound packet*/
      send P to ID-inbound subsystem to detect malicious behaviors;
```

```
  else /*P is a forwarded packet*/
     if (P' destination IP is in Protection List)
        P is sent to ID-forwarded subsystem to detect malicious behavior;}}
```

```
Algorithm2: type-2 Header Processor /*filters inner packets*/
Input: A packet P sent by a type-2 switch; Host List; Protection List
Output: P is an inner packet
{If (P' source IP and destination IP are both in Host List) /*P is an inner packet*/
   send P to ID-inner subsystem to detect malicious behavior;}
```

Table 1. Packet classification

IPs / Packets	Source IP	Destination IP
Inner	in Host List	in Host List
Out-bound	in Host List	not in Host List
In-bound	not in Host List	in Host List
Forwarded	not in Host List	not in Host List

IP address	MAC address	Subnet ID

Figure 4. Host List which has three fields to record local hosts' address information

3.2 Intrusion Detector

ID-inner and ID-outbound subsystems respectively detect whether or not local users of the underlying network management system are attacking other local hosts/servers or remote networks. ID-forwarded subsystem detects attacks launched by hosts in other network management units to a protected remote network. ID-inbound subsystem detects whether there is an incoming attack. All the four Intrusion-Detector subsystems use CUSUM algorithm to detect DoS and DDoS attacks, and both ID-inbound and ID-inner subsystems further employ ordinary Intrusion Detection Systems (IDSs) to detect attacks other than DoS and DDoS.

3.2.1. Detection in a subsystem

In this study, once an attack is found, we use Heap tables and Accumulator to check who is/are launching the attack. Heap table, a unique structure supported by MYSQL DBMS, is implemented on memory to accelerate data access and process speed. CSIPS prepares a Heap table for each local host. Each Heap table is given an ID which is the IP address of the corresponding host. A Heap table as shown in Table 2 has six attributes, S-IP, Protocol, Type, Count, Size, and sequence # respectively representing a source IP (e.g., 140.128.102.12) that sends packets to host$_j$ (e.g., j=140.128.101.111 which is also its Heap-table ID), packet protocol (e.g., TCP), packet type (e.g., SYN or Request), packet count (e.g., 35021 packets) which is number of packets sent to host$_j$ by the source IP, accumulated size of packets sent by the source IP, and sequence #s conveyed on the packets sent by the source IP. In the following, we assume that a packet P's source IP is Sr_IP, destination IP is Des_IP, protocol is T, type is Ty, and $|P|$ is P's packet size.

Table 2. The Heap Table whose ID is 140.128.101.111 (within second Q)

S-IP	Protocol	Type	Count	Size (KB)	Sequ. #
140.128.102.12	TCP	SYN	35,021	2,189	2372311437 3357477985 …
140.128.102.12	TCP	Req	80	5,032	3299181253 3335287903 …
140.128.101.20	UDP	--	1,438	978,272	No available
164.13.77.15	ICMP	Req	553	35	18176
143.128.101.2	TCP	SYN	10	0.625	658919540

```
Accumulator (X, $Sr\_IP$, $T$, $Ty$, $Des\_IP$, seq#(k)) /*seq#(k) is used in
    ID-inner subsystem*/
 /* Accumulator/out-Accumulator accumulates packet statistics, X represents Heap
    table or out-Heap table*/
```

```
{if ( $Sr\_IP$ , $T$ and $Ty$ exist in table $Des\_IP$ , e.g., tuple k)
    {if( P’s sequence number Q does not exist in seq#(k))
        { $count(k) = count(k) + 1; size(k) = size(k) + |P|;$ append Q to seq#(k)}
    else discard P;}
else insert : S_IP= $Sr\_IP$ , count=1, protocol= $T$ , type= $Ty$ , size= $|P|$ and seq# as
    a new tuple to table $Des\_IP$ ;}
```

Figure 5. The task of Accumulator

ID-inbound and ID-inner on receiving information of a packet from a Header Processor check Index_table, a table for indexing, to search the corresponding Heap table. As receiving a packet P whose destination IP is, e.g., j, if there is a record k in Heap table j which has the same source address, protocol and packet type as those of P, Accumulator increases k’s “Count” by one and “Size” by size of P. Otherwise, it inserts packet information as a new record into the table. Figure 5 summarizes the algorithm with which the Accumulator accumulates packet statistics. Table 2 lists four IPs having transmitted packets to 140.128.101.111 within a specific period of time, e.g., a specific second Q.

From this table, when an attack is found, we can identify who is issuing the DoS/DDoS attack under the assumption that attack packets are sent to victims without forging source IP addresses. For example, if the attack is an bandwidth consumption one, 140.128.101.20 will be the most suspectable IP since the packet size that has been accumulated is about 99.77% ($= \frac{978272}{2189+978272+5.032+35+0.625}$) of the size of all packets received within one second, e.g., Q. However, if it is an resource consumption attack, 140.128.102.12 will be the most suspectable one since its packet count is about 94.39% ($= \frac{35021}{35021+80+1438+553+10}$) of the total packet count within the specific second Q. Of course, if attackers forge their source IPs, then the identification will not work. So, in this study Heap tables and Accumulator are not the major components in detecting attacks.

An alerting message sent by a subsystem to notify Response Manager that an attack is found is formatted by: Subsystem-ID, S-IP address and protocol, where Subsystem-ID shows which subsystem discovers the attack, S-IP address is the IP of the hacker, and protocol shows the protocol of the attacking packets.

ID-outbound and ID-forwarded together construct another Heap table, named out-heap table, which accumulates packets sent to a protected remote IP, subnet or network management unit. We call them remote objects. The structure of a out-heap table is the same as that of a heap table. CSIPS prepares an out-heap table for each remote object. There is also an accumulator, named out-accumulator, which works similarly to the accumulator working for ID-inner and ID-inbound. Figure 6 lists the task of a detector.

```
Detector (P, X-table, $Des\_IP$, $\alpha_P$, $h_P$, $N_P$)
    /*detecting DoS/DDoS attacks, P is an inbound, an out-bound or a forwarded
      packet, and X-table is a Heap table or an out-Heap table */
{if (at least one DoS/DDoS attack is found by a detection subsystem) /*invoking
    CUSUM algorithm given $\alpha_P$, $h_P$, and $N_P$ */
   {1. out-Accumulator or Accumulator checks X- table to see which IPs are now
      issuing resource consumption DoS/DDoS attacks by calculating the packet
      count that each source IP has sent to $Des\_IP$ over total packet count in the
      underlying X-table; /*calculate percentages of packet count that each source IP
      has*/
    2. out-Accumulator or Accumulator checks X- table to see which IPs are now
      issuing bandwidth consumption DoS/DDoS attacks by calculating the packet
      size that each source IP has sent to $Des\_IP$ over total packet size in the
      underlying X-table; /*calculate percentages of packet size that each source IP
      has*/
    3. send an alerting message to Response Manager;
    4. send intrusion information to BLD; /*Firewall will block hackers' packets
       based on hackers' information in BLD*/}}
```

Figure 6. The detection task that a detector performs

The CUSUM algorithm works on Heap tables and out-Heap tables. Algorithms 3 and 4 respectively list the details of how out-bound and in-bound subsystems detect DoS/DDoS attacks.

Algorithm 3: detection on ID-outbound packets

```
Input: An out-bound packet P sent by a type-1 Header Processor /*Assume P's source
        IP is $Sr\_IP$, destination IP is $Des\_IP$, protocol is $T$, type is $T_y$ and seq#
        is Q*/
Output: packet statistics, intrusion information and an alerting message
{1. search Des_IP's corresponding out-heap table, $Des\_IP$; /* $Des\_IP$ is
    established for a remote object*/
  2. Lock (out-heap-$Des\_IP$) /*concurrency control to synchronize this algorithm
    with ID-forwarded subsystem*/
  3. call Accumulator (out-Heap table, $Sr\_IP$, $T$, $T_y$, $Des\_IP$);
  4. Unlock (out-heap-$Des\_IP$)
  5. call Detector (P, out-Heap table, $Des\_IP$, $\alpha_{outbound}$, $h_{outbound}$, $N_{outbound}$); /*find
    out out-bound DoS/DDoS attacks*/
  6. if (timer times out)
        set timer = 10 seconds;}
```

Algorithm 4: detection on ID-inbound packets

```
Input: An in-bound packet P sent by a type-1 Header Processor /*Assume P's source
       IP is $Sr\_IP$, destination IP is $Des\_IP$, protocol is $T$, type is $T_y$ and seq#
       is Q*/
Output: packet statistics, intrusion information and an alerting message
{1. search the corresponding heap table, e.g., table $Des\_IP$;
 2. Lock (heap-$Des\_IP$) /*concurrency control to synchronize this algorithm with
    ID-inner subsystem*/
 3. call Accumulator (Heap table, $Sr\_IP$, $T$, $T_y$, $Des\_IP$);
 4. Unlock (heap-$Des\_IP$)
 5. call Detector (P, Heap table, $Des\_IP$, $\alpha_{inbound}$, $h_{inbound}$, $N_{inbound}$); /*find out
    in-bound DoS/DDoS attacks*/
 6. detect ordinary attacks other than DoS/DDoS attacks by using an IDS system;
 7. If (an attack other than DoS/DDoS is found)
      { send an alerting message to Response Manager;
        send intrusion information to BLD;}
 8. if(timer times out)
      send contents of the heap table $Des\_IP$ as packet statistics to BLD;} /*timer is
      reset by ID-outbound. i.e., algorithm 3*/
```

The algorithm of the ID-forwarded subsystem is similar to algorithm 3, except that P is a forwarded packet and it does not send out-Heap table to remote CSIPSs, otherwise data will be duplicated. Note that for the same remote object ID-forwarded and ID-outbound share the same out-heap table.

Regardless of whether an attack is found or not, each subsystem sends packet statistics to BLD periodically once per 10 seconds [3]. When a CSIPS finds out that there is an DoS/DDoS attack launched toward a remote object, it alerts the object. This is helpful, particularly in two cases. The first is there is an DDoS, but the object's CSIPS does not detect the attack. The second is when a remote objects CSIPS discovers there is a DoS attack, but from its packet statistics, i.e., heap tables, it can not identified the attackers. However, by collecting packet statistics from united defense CSIPSs' heap tables, we can realize most attack packets flowing through which network management units. This is helpful in tracing back to hackers [4].

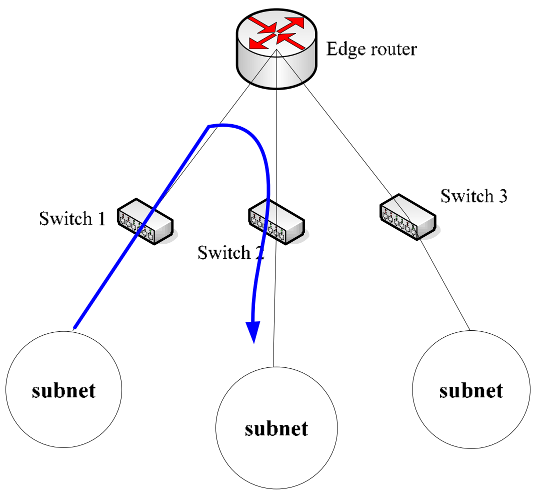

Figure 7. a subnet (e.g., subnet 1) attacks another subnet (e.g., subnet2)

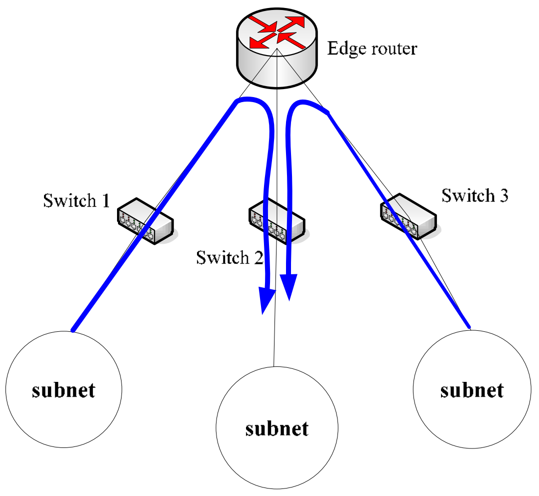

Figure 8. two or more subnets (e.g., subnets 1and 3) attack a subnet (e.g., subnet2)

3.2.2. detecting inner attacks

Traffic flowing through a type-2 switch is often fewer than that flowing through the corresponding edge router since it is very usual that more than one switch are connected to a router. There are three inner attack models. Model-1 is intra-subnet attack, i.e., a host A or a group of hosts G attack another host B, and A and B or B and G belong to the same subnet. Model-2 as shown in Figure 7 is that a subnet attacks another subnet. Model-3 as shown in Figure 8 is that more than one subnet attack a subnet simultaneously. In this study, we call a type-2 switch that directly connects to a router level-1 switch. Those downstream switches connected to level-i switches are called level-(i+1) switches, i = 1,2,3,….,n-1, where n are total hierarchical levels of switches under a router.

Packets of model-1 attack may be issued at switches of any level. To detect inter-subnet attacks, a type-2 Header Processor is connected to a level-1 switch. We call the processor, type-2' Header Processor. Also, to avoid missing model-1 attacks, all lower level (level-2 to level-n) switches also need to be monitored. However, such will introduce a problem, i.e., when a packet flows through k switches before arriving at its destination, it will be duplicated k times in packet statistics. This may conduct a false alarm. So, ID-inner should be able to check to see whether a packet P has been received or not by checking P's source IP, destination IP and sequence number. If yes, P will be discarded.

In model-3, owing to unknowing number of attacking subnets, it is hard for us to choose proper threshold if DoS/DDoS attacks are detected on sender side. Detecting on receiver side can avoid the problem since a subnet's normal incoming traffic's CUSUM parameters X_{inner}, Z_{inner}, α_{inner}, h_{inner}, N_{inner} can be observed beforehand. One may point out that we can monitor network traffic flowing toward a host to observe its X, Z, α, h and N in advance, regardless of the incoming network traffic is generated by inner hosts or outer networks. Nevertheless, we calculate the CUSUM parameters to judge whether there is an DoS/DDoS attack, also regardless of where the packets come from. In other words, we do not need to separate inner and incoming packets. It is true, but CUSUM algorithm is often implemented on router

[14][15] as a network-based IDS [16] which does not collect model-1 traffic. Seldom IDSs that implement CUSUM are host-based. That is, IDSs using CUSUM often omit inner traffic when detecting DoS/DDoS attacks since they assume attacks only come from outside world. Hackers always attack outside-world hosts. But, this assumption is not always true [13].

In this study, inner packets are also cumulated in Heap tables. Accumulator accumulated numbers of packets sent to a host in the corresponding Heap table, regardless of they are inner or incoming packets. But, ID-inner only checks the portion of tuples of which the source IPs are in Host List, i.e., local users. When ID-inner discovers that there is a DoS/DDoS attack, like that in ID-inbound, we can identify the hackers by checking the Heap table to see which inner source IPs send most packets (resource consumption attack) or the largest accumulated packet size (bandwidth consumption attack).

To discriminate whether a DoS/DDoS is an inter-subnet or intra-subnet attack, we further classify tuples in Host List into groups. Those hosts belonging to a subnet form a group. ID-inner on receiving a packet P checks P's source IP and destination IP to see if they belong to the same subnet. If not, P is an inter-subnet packet. Otherwise, it is an intra-subnet packet. The accumulator an ID-inner uses to accumulate packet statistics is the same as the ones employed by other ID subsystems, i.e., algorithms 3 to 5, with a little change as follows. Each time when receiving a packet P, the Accumulator checks the corresponding Heap table (recall, identifying the table based on P's destination IP) to see whether or not P's source IP, protocol and packet type are already in the table. If not, it inserts a new tuple into the Heap table given P's source IP, protocol, type, count =1, $|P|$ and P's sequence number. If yes, e.g., tuple k, it further checks to see whether the sequence number exists in tuple k or not. If yes, P is a duplicated packet. The Accumulator discards P. Otherwise, it appends the sequence number to k's sequence # field, and increases "count" by one and "size" by $|P|$ as a part of network traffic cumulated by CUSUM algorithm. The detection algorithms of model 1 and models 2 and 3 are as follows.

Algorithm 5: Detecting model-1 inner attack

```
Input: A packet P sent by a type-2 Header Processor; /*Assume P's source IP is
        $Sr\_IP$, destination IP is $Des\_IP$, protocol is $T$, type is $T_y$ and seq# is Q*/
Output: Whether there is a model 1 attack
{1. search the corresponding heap table, e.g., table $Des\_IP$;
 2. **Lock (heap-$Des\_IP$)** /*concurrency control to synchronize this algorithm with
    that of ID-inbound subsystem and that used to detect inter-subnet attacks*/
 3. call Accumulator (Heap table, $Sr\_IP$, $T$, $T_y$, $Des\_IP$, seq#(k));
```

```
   4. **Unlock (heap-$Des\_IP$)**
   5. call Detector (P, Heap table, $Des\_IP$, $\alpha_{inner}$, $h_{inner}$, $N_{inner}$); /*find out inner
      DoS/DDoS attacks*/
   6, 7 and 8 are respectively the same as steps 6~8 in algorithm 4, i.e., detection on
   ID-inbound packets.
```

Algorithm 6: Detecting model-2 and model-3 attacks

```
Input: A packet P sent by type-2' Header Processor /*Assume P's source IP is $Sr\_IP$,
       destination IP is $Des\_IP$, protocol is $T$, type is $T_y$ and seq# is Q*/
Output: Whether there is a model 2 or model 3 attack
{1. search the corresponding heap table, e.g., table $Des\_IP$;
 2. **Lock (heap-$Des\_IP$)** /*concurrency control to synchronize this algorithm with
    that of ID-inbound subsystem and that used to detect intra-subnet attacks*/
 3. call Accumulator (Heap table, $Sr\_IP$, $T$, $T_y$, $Des\_IP$);
 4. **Unlock (heap-$Des\_IP$)**
 5. call Detector (P, Heap table, $Des\_IP$, 2*$\alpha_{inner}$, 2*$h_{inner}$, 2*$N_{inner}$); /* find out
    inner DoS/DDoS attacks, but why two times? Since a packet flows through two
    level-1 switches, i.e., for entering and leaving their router*/;
 6, 7 and 8 are respectively the same as steps 6~8 in algorithm 4, i.e., detection on
```

ID-inbound packets.

3.3 Response Manager

Response Manager takes charge of sending messages to alert its administrators, that there is an attack. It also notifies administrators of remotely protected network management units if attacks are launched to the remote units.

Algorithm 7: Response Manager deals with alerting message

```
Input: an alerting message M from an ID subsystem; /*M is formatted by the
subsystem ID, hacker's source IP, and protocol*/
Output: message to administrator or remote network's response manager; BLD
{1. if (M from ID-inbound or ID-inner subsystem)
      send an alerting message to administrator;
  2. if (M from ID-outbound or ID-forwarded subsystem)
      send an alerting message to administrator of remotely protected network
      managements;}
```

3.4 Black List Database (BLD)

BLD lists the definite intrusion information, such as attack time, source address, destination address, protocol and attack type, to provide firewall or router with a black list. Packets whose source IPs appear in the list will be discarded, sessions that connect hackers and hosts in underlying network management unit will be disconnected, and requests issued by hackers to establish connections will be rejected. When attacks are found, ID-subsystems send intrusion information to BLD.

Chapter 4. Experiments and Discussions

The experimental testbed comprises resources of Tunghai University. We use twelve computers to simulate the working environment which consists of one victim (i.e., in protected unit), three attackers, one type-1 Head processor, three type-2 Head processors and four intrusion detectors (acting as ID-inbound, ID-outbound, ID-forwarded and ID-inner subsystems). Table 3 shows the specifications. In this environment, four Header processors and four detection subsystems form a CSIPS. When detecting in-bound attacks, the victim and the three attackers are respectively placed inside and outside the NMU the CSIPSs protects. In detecting out-bound attacks, the three attackers are in the NMU and the victim is an outside node. In detecting forwarded attacks, both the three attackers and the victim are placed outside of NMU. When detecting inner attacks, the four nodes are all in the NMU.

Table 3. The Specifications of CSIPS

Attributes / Node	Processor	Memory (GB)
victim	AMD Athlon64*2 3800+	3
Attacker 1	Intel Pentium M 1.73G	2
Attacker 2	AMD Athlon64 3000+	1
Attacker 3	AMD Athlon64*2 3800+	2

Type-1 Header processor	Intel Q6600 2.4G	2
Type-2 Header processor 1	AMD Athlon64*2 3800+	2
Type-2 Header processor 2	AMD Athlon64*2 3800+	3
Type-2 Header processor 3	AMD Athlon64*2 3800+	3
ID-inbound subsystem	Intel E8300 2.8G	3
ID-outbound subsystem	Intel E8300 2.8G	2
ID-forwarded subsystem	AMD Athlon64*2 3600+	2
ID-inner subsystem	AMD Athlon64*2 3800+	2

Security systems to be tested include Kaspersky Anti-Hacker 1.8.180 (Kaspersky for short, by Kaspersky Labs), McAfee VirusScan Home Edition 7.0 (McAfee VirusScan forshort, by McAfee, Inc), Panda Internet Security 2010 (Panda for short, by Panda software), Snort, Fortinet 100A (FG100A hardware), and CSIPS.

Intrusion tools are used to launch bandwidth and resource consumption attacks. Table 4 lists the attack details, where ANP/sec stands for average number of launched packets per second. Each attack is issued twenty times.

A total of three experiments were performed. The first experiment evaluates performance of the detection of in-bound resource and bandwidth consumption attacks. The second evaluated performance of the detection of out-bound, forwarded and inner resource and bandwidth consumption attacks. The third measures detection accuracies of resource and bandwidth consumption attacks.

Table 4 The Information about Attacks

Attack Type	Attack Period	ANP/sec
Resource consumption attack	10 second	7,604
		15,645
		17,075
Bandwidth consumption attack		2,160
		2,685

4.1 Detecting In-bound Attacks

(1) Resource Consumption Attacks

Table 5 lists the detection results of resource consumption attack on 7,604 ANP/sec. ART/SD, ATT/SD, AWT/SD, Max/Min PL, APL, Max/Min ML and AML respectively represent average response time (the period from attack begins to attack is first discovered)/standard deviation, average turnaround time (the period from attack begins to detection finishes)/standard deviation, average waiting time (the period packets wait to be detected)/standard deviation, Maximum/Minimum processor load, average processor load, Maximum/Minimum memory load, and average memory load. CSIPS has the lowest ART on CUSUM parameter a=500, h=1000 and N=2000, and it detects an attack almost immediately, i.e., AWT=0. FG-100A and Kaspersky detect number of connections, and their minimum scales of response times are in seconds. McAfee and Panda do not provide detection methods to detect ARTs. FG-100A's processor load is the highest, because we do not limit number of connections. However, Kaspersky, McAfee VirusScan, Panda Titanium, Snort and FG-100A do not support measuring mechanisms to evaluate AWT.

McAfee and panda 無法知道偵測方法. FG-100 發現攻擊時, 防火牆還是允許聯線通過, 所以 load 非常高

Table 5. The in-bound detection results of resource consumption attacks on 7,604 ANP/sec

Statistics / Secu. Systems	ART/SD (sec.)	ATT/SD (sec.)	AWT/SD (sec.)	Max/Min PL (%)	APL (%)	Max/Min ML (%)	AML (%)
Kaspersky	2/0	10/0	--	36/26	28.3	32/30	31.1
McAfee VirusScan	2/0	10/0	--	53/43	52	76/65	67.2
Panda	2/0	10/0	--	33/30	31.4	30/29	29.5
Snort	6/0	13/0.03	--	30.5/23.3	26.9	15.9/15.6	15.7
FG-100A	1/0	10/0	--	99/73	90.8	43/40	41.7
CSIPS on α=500, h=1000, and N=2000 packets	0.64/0.023	10/0	0/0	38/24	28.6	38/25	29.6

Table 6 and Table 7 respectively show the results of attack intensities on 15,645 ANP/sec and 17,075ANP/sec. Figure 9 shows the ARTs when different CUSUM parameters N, α and h are individually given different values. All AWTs and ATTs of CSIPS on different parameters are the same, i.e., AWT=0 and ATT=10 sec. The sensitively of ART for N is defined as $\frac{ART_{N1} - ART_{N2}}{N_1 - N_2}$ on specific values of a and h, where ART_{N1} and ART_{N2} are $ARTs$ on $N = N_1$ and $N = N_2$, respectively. The sensitivity of ART for α (or h) is defined as $\frac{ART_{\alpha 1} - ART_{\alpha 2}}{\alpha_1 - \alpha_2}$ (or $\frac{ART_{h1} - ART_{h2}}{h_1 - h_2}$) where $ART_{\alpha 1}$ and $ART_{\alpha 2}$ (ART_{h1} and ART_{h2}) are respectively $ARTs$ on $\alpha = \alpha_1$ and $\alpha = \alpha_2$ ($h = h_1$ and $h = h_2$). The average sensitivity of the ART for N is 0.00275. That for α is 0.00048 and that for h is 0.00056. The best point is when N=2000, h=1000 and α=500. That is why they are employed in this experiment.

Table 6. The in-bound detection results of resource consumption attack on 15,645 ANP/sec

Statistics / Secu. Systems	ART/SD (sec.)	ATT/SD (sec.)	AWT/SD (sec.)	Max/Min PL (%)	APL (%)	Max/Min ML (%)	AML (%)
Kaspersky	2/0	10/0	--	36/29	32	33/30	31.7
McAfee VirusScan	2/0	10/0	--	54/44	52.2	77/64	67.5
Panda	2/0	10/0	--	33/31	31.9	30/29	29.7
Snort	8/0	15/0.02	--	73.3/22.8	41.5	16/15.9	15.9
FG-100A	1/0	10/0	--	99/79	92.2	43/40	40.9

CSIPS on α =500, h=1000, and N=2000 packets	0.56/0.02	10/0	0/0	41/22	31.2	38/27	32.4

Table 7. The in-bound detection results of resource consumption attack on 17,075 ANP/sec

Statistics / Secu. Systems	ART/SD (sec.)	ATT/SD (sec.)	AWT/SD (sec.)	Max/Min PL (%)	APL (%)	Max/Min ML (%)	AML (%)
Kaspersky	2/0	10/0	--	37/29	33.2	33/30	32.1
McAfee VirusScan	2/0	10/0	--	55/43	52.3	78/65	68.4
Panda	2/0	10/0	--	35/32	33.5	30/29	29.7
Snort	8/0	15/0.73	--	80/30.1	51.5	16/15.4	15.8
FG-100A	1/0	10/0	--	99/85	93.9	43/40	41.1
CSIPS on α =500, h=1000, and N=2000 packets	0.54/0.032	10/0	0/0	40/26	31.8	36/26	31.5

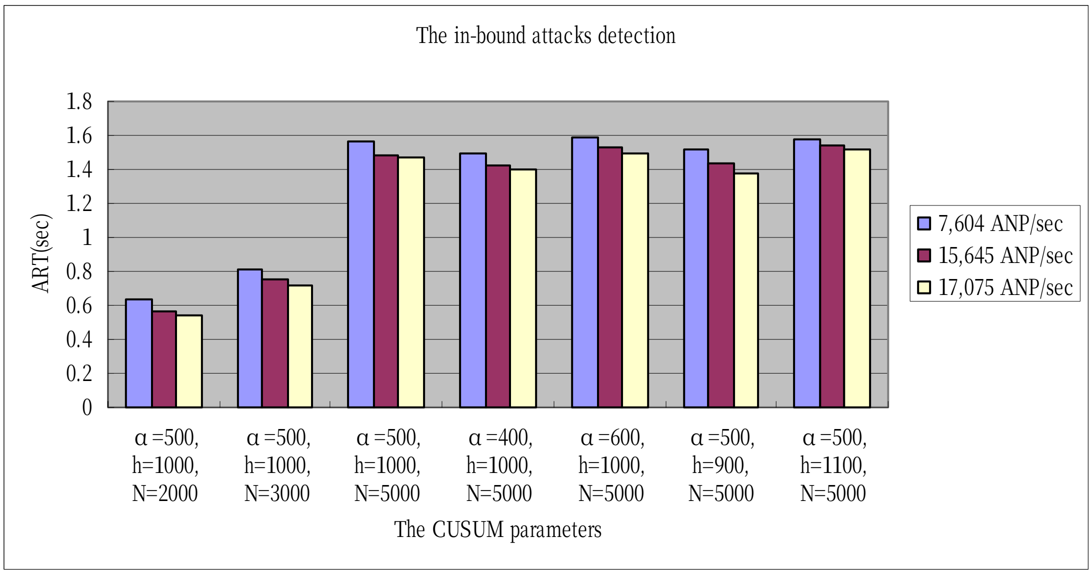

Figure 9. ARTs of in-bound resource consumption attacks against different CUSUM parameters on different values given different attack intensities

(2) Bandwidth Consumption Attacks

The experimental results of bandwidth consumption attack on 2,160 ANP/sec and 2,685 ANP/sec are shown in Table 8 and Table 9, respectively. Compared with Tables 5~7, FG-100A's processor load decreases quickly from more than 90% to less than 30% since when FG-100A discovers that there is a bandwidth consumption attack, it throws the packets directly, and terminates the corresponding sessions. Figure 10 shows the ARTs when CUSUM parameters N, α and h are individually given different values. The average sensitivity of ART for N, α and h are respectively 0.000275, 0.00035 and 0.000575 given the three parameters different values. The best point is when N=2000, h=900 and α=500. That is why we use them as the parameter values to do the second experiment.\

Table 8. The in-bound detection results of bandwidth consumption attack on 2,160 ANP/sec

Statistics / Secu. Systems	ART/SD (sec.)	ATT/SD (sec.)	AWT/SD (sec.)	Max/Min PL (%)	APL (%)	Max/Min ML (%)	AML (%)
Kaspersky	1/0	10/0	--	22/11	17	30/28	28.7
McAfee VirusScan	2/0	10/0	--	50/44	46	64/58	59
Panda	2/0	10/0	--	34/32	33	32/31	31.6
Snort	5/0	13/0.01	--	32/24.6	25.3	15.4/15.1	15.3
FG-100A	1/0	10/0	--	31//23	28.2	43/43	43
CSIPS on α =500, h=900, and N=2000 packets	0.62/0.03	10/0	0/0	40/36	38.1	30/21	25.9

Table 9. The in-bound detection results of bandwidth consumption attack on 2,685 ANP/sec

Statistics / Secu. Systems	ART/SD (sec.)	ATT/SD (sec.)	AWTSD (sec.)	Max/Min PL (%)	APL (%)	Max/Min ML (%)	AML (%)
Kaspersky	1/0	10/0	--	22/13	17.5	50/44	45.2
McAfee VirusScan	2/0	10/0	--	50/46	48	65/59	59.3
Panda	2/0	10/0	--	35/32	33.2	33/32	32.6
Snort	5/0	13/0.02	--	34/24.8	26.1	15.6/15.2	15.4
FG-100A	1/0	10/0	--	33/22	27.6	43/43	43

CSIPS on α=500, h=900, and N=2000 packets	0.61/0.05	10/0	0/0	41/36	38	30/22	27.2

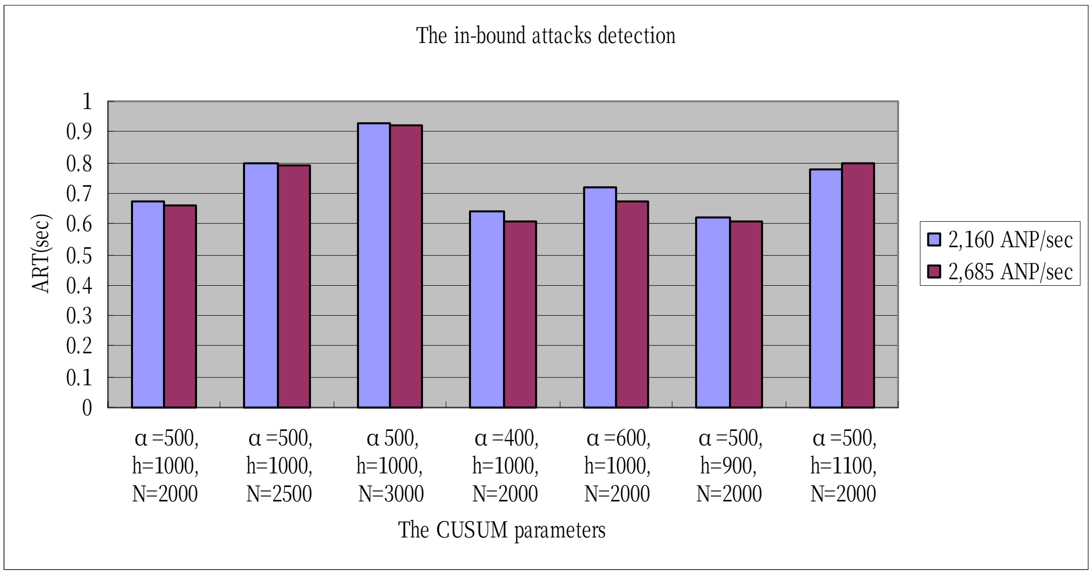

Figure 10. ARTs of in-bound bandwidth consumption attacks against different CUSUM parameters on different values given different attack intensities

4.2 Detecting Out-bound, Forwarded and Inner Attacks

(1) Resource Consumption Attacks

Tables 10, 11 and 12 respectively show detection results of resource consumption attacks on 7,604 ANP/sec, 15,645 ANP/sec and 17,075 ANP/sec. No other security systems are compared since they do not provide mechanisms to detect the three types of packets. Figures 11 show the ARTs of out-bound attacks when CUSUM parameters N, α and h are individually given different values. The average sensitivities of ARTs for N, α and h are respectively 0.000274, 0.00053 and 0.00033. Figures 12 and 13 respectively show the ARTs of forwarded attacks when CUSUM parameters N, α and h are given different values. The average sensitivities of ARTs for N, α and h are respectively 0.000258, 0.00035 and 0.00038.

Table 10. Detection results of resource consumption attack on 7,604 ANP/sec when α=500, h=1000 and N=2000

Attacks	ART/SD (sec.)	ATT/SD (sec.)	AWT/SD (sec.)	Max/Min PL (%)	APL (%)	Max/Min ML (%)	AML (%)
out-bound	0.7/0.02	10/0	0/0	37/28	33.5	36/32	34.6
forwarded	0.72/0.04	10/0	0/0	35/26	30.8	34/29	30.5
inner	0.69/0.02	10/0	0/0	36/23	29.8	34/25	29.1

Table 11. Detection results of resource consumption attack on 15,645 ANP/sec when α=500, h=1000 and N=2000

Attacks	ART/SD (sec.)	ATT/SD (sec.)	AWT/SD (sec.)	Max/Min PL (%)	APL (%)	Max/Min ML (%)	AML (%)
out-bound	0.5/0.04	10/0	0/0	37/30	33.8	34/32	33.5
forwarded	0.55/0.03	10/0	0/0	36/28	31.2	34/29	30.8
inner	0.52/0.06	10/0	0/0	37/25	32.2	38/29	32.6

Table 12. Detection results of resource consumption attack on 17,075 ANP/sec when α=500, h=1000 and N=2000

Attacks	ART/SD (sec.)	ATT/SD (sec.)	AWT/SD (sec.)	Max/Min PL (%)	APL (%)	Max/Min ML (%)	AML (%)
out-bound	0.47/0.02	10/0	0/0	37/30	33.5	33/30	32.8
forwarded	0.5/0.06	10/0	0/0	35/28	31.6	35/28	31.1
inner	0.46/0.02	10/0	0/0	39/26	32.6	38/31	33.5

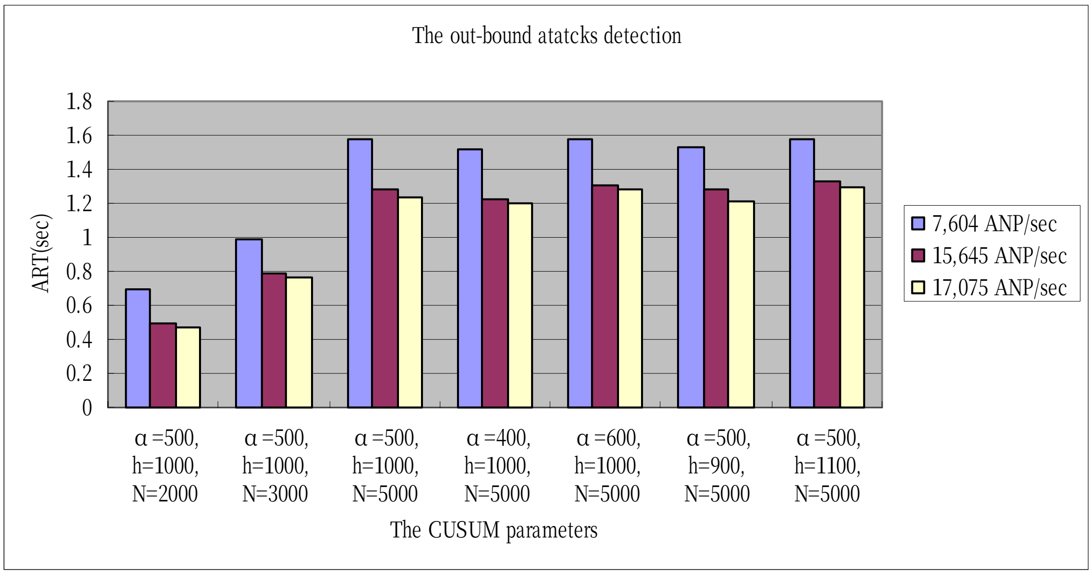

Figure 11. ARTs of out-bound resource consumption attacks against different CUSUM parameters on different values given different attack intensities

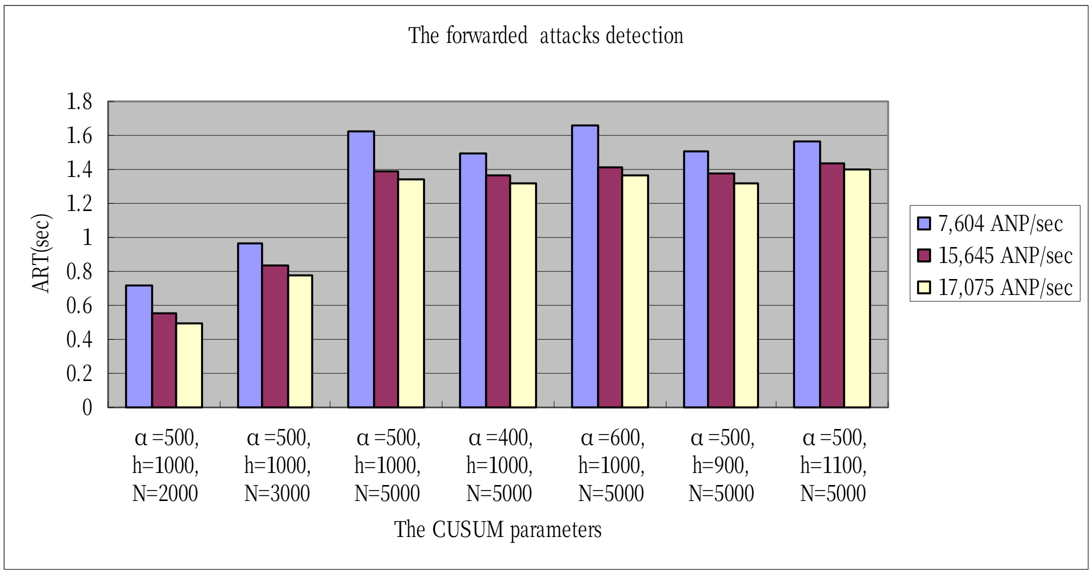

Figure 12. ARTs of forwarded resource consumption attacks against different CUSUM parameters on different values given different attack intensities

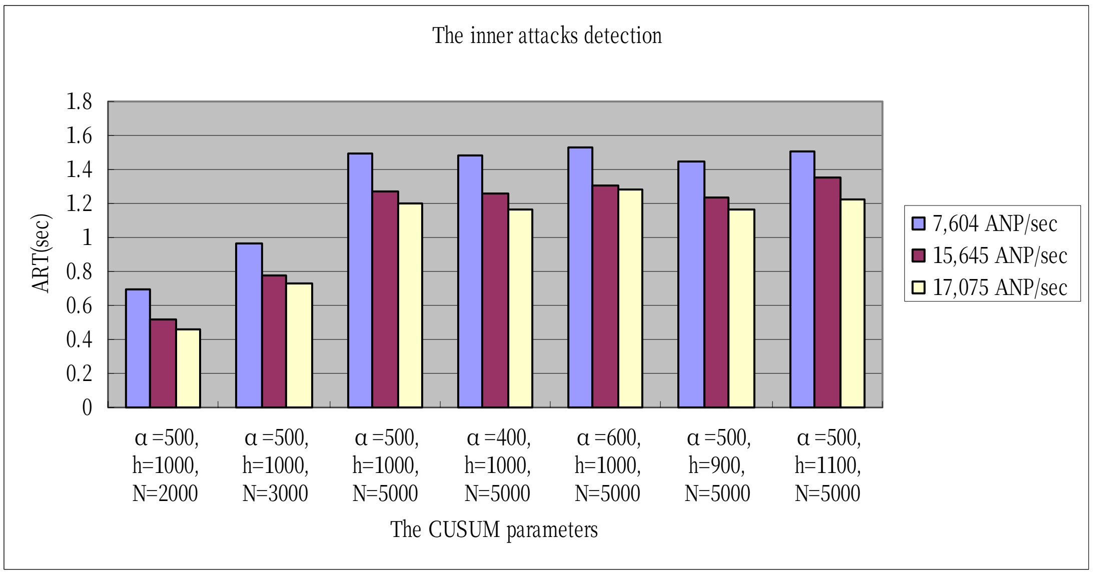

Figure 13. ARTs of inner resource consumption attacks against different CUSUM parameters on different values given different attack intensities

(2) Bandwidth Consumption Attacks

The detection results of bandwidth consumption attacks on 2,160 ANP/sec and 2,685 ANP/sec are listed in Tables 13 and 14, respectively. Figures 14 shows ARTs of out-bound attacks on different values when parameters N, α and h are individually given different values. The average sensitivities of ARTs for N, α and h are respectively 0.000285, 0.000247 and 0.000265. Figures 15 and 16 respectively shows ARTs of forwarded attacks on different values when parameters N, α and h are given different values. The average sensitively of inner attack's ARTs for N, α and h are respectively 0.000275, 0.0003 and 0.0002. The experimental results shown in Figures 13 and 14 are not significantly different from those shown in Figure10. The best point is almost the same.

Table 13. Detection results of bandwidth consumption attack on 2,160 ANP/sec when α=500, h=900 and N=2000

Attacks	ART/SD (sec.)	ATT/SD (sec.)	AWT/SD (sec.)	Max/Min PL (%)	APL (%)	Max/Min ML (%)	AML (%)
out-bound	0.675/0.05	10/0	0/0	42/37	39.8	31/24	26.8
forwarded	0.68/0.04	10/0	0/0	40/37	38.8	31/22	26.4
inner	0.64/0.041	10/0	0/0	40/34	38.1	29/21	24.7

Table 14. Detection results of bandwidth consumption attack on 2,685 ANP/sec when α=400, h=1000 and N=2000

Attacks	ART/SD (sec.)	ATT/SD (sec.)	AWT/SD (sec.)	Max/Min PL (%)	APL (%)	Max/Min ML (%)	AML (%)
out-bound	0.678/0.03	10/0	0/0	42/37	39.2	31/24	27.5
forwarded	0.7/0.062	10/0	0/0	42/36	38.9	30/22	27.1
inner	0.64/0.024	10/0	0/0	40/36	37.8	30/21	26.2

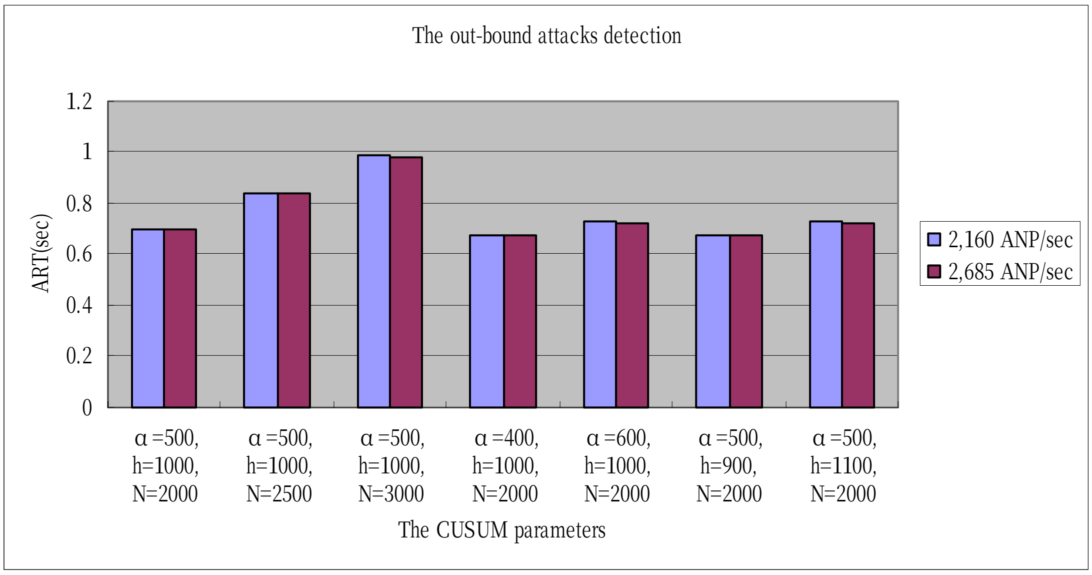

Figure 14. ARTs of out-bound bandwidth consumption attacks against different CUSUM parameters on different values given different attack intensities

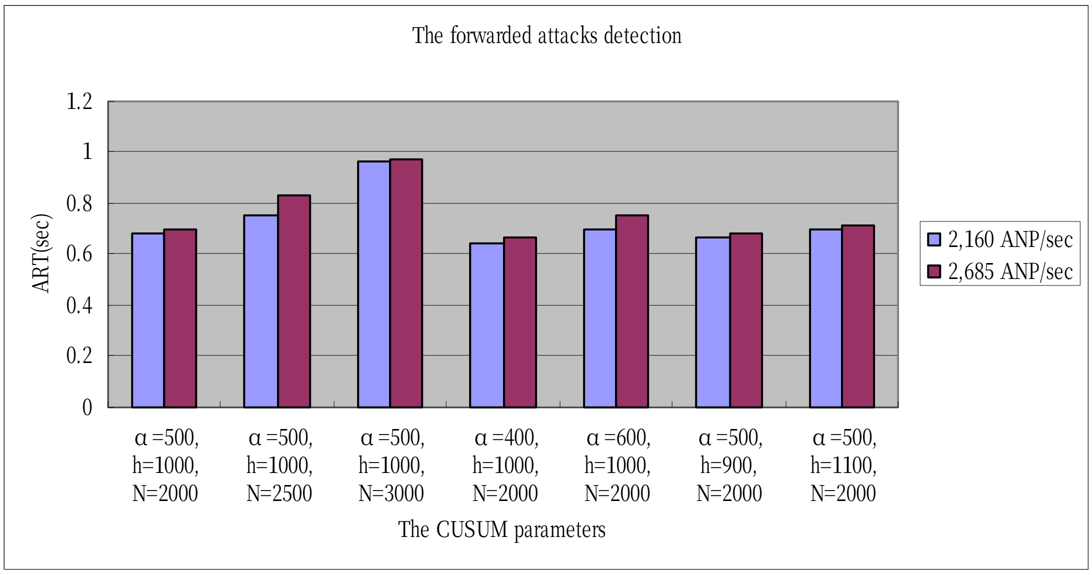

Figure 15. ARTs of forwarded bandwidth consumption attacks against different CUSUM parameters on different values given different attack intensities

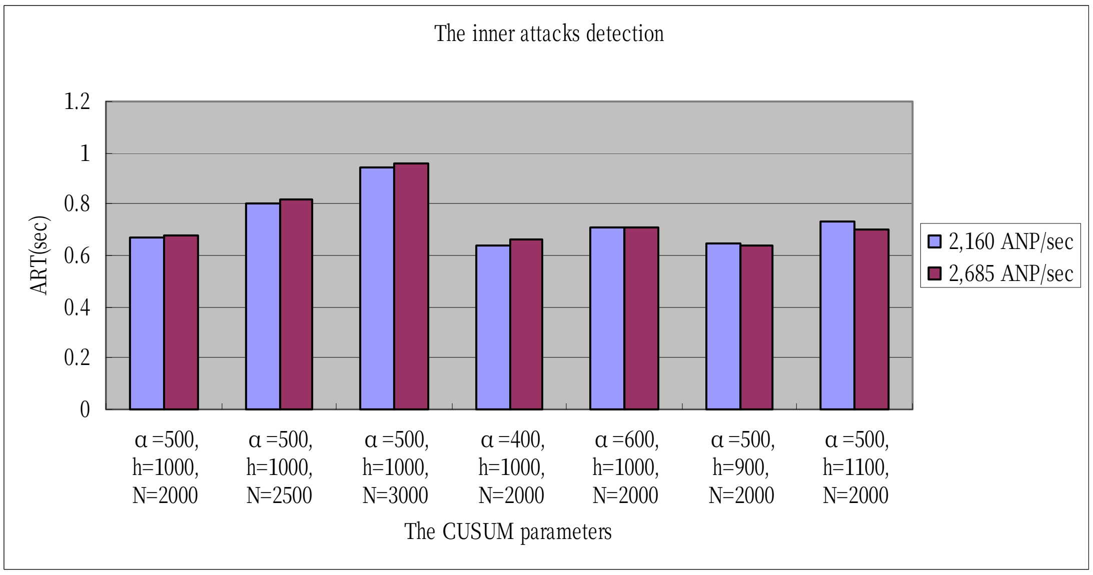

Figure 16. ARTs of inner bandwidth consumption attacks against different CUSUM parameters on different values given different attack intensities

4.3 Detection Accuracy

The third experiment is measuring detection accuracies of resource/bandwidth consumption attacks. We gather 2000 times of normal and attack traffic of 10 seconds respectively, including DoS/DDoS resource consumption attack (i.e., TCP flood and ICMP flood) and DoS/DDoS bandwidth consumption attack (i.e., UDP flood). The attack intensities range from 500 to 15,000 packets/sec [25]. Table 15 shows the detection accuracy. Kaspersky performs the best (99.7%). The inbound attacks detection accuracy of CSIPS on $\alpha=500$, $h=1000$ and $N=5000$ is 98.4%. True Positive (True Negative) represents how many attack (normal) packets are accurately detected. False Positive (False Negative) represents the ratios that normal (attack) packets are detected as attack (normal) packets. In CSIPS, True Negative is not 100% because some normal packets have attack pattern.

Table 15. Detection Accuracy of Resource/Bandwidth Consumption Attack

Statistics / Secu. Systems	True Positive	True Negative	False Positive	False Negative	Detection Accuracy
Kaspersky	99.4%	100%	0%	0.6%	99.7%
McAfee VirusScan	89.5%	100%	0%	10.5%	94.75%
Panda Titanium	87.7%	100%	0%	12.3%	93.85%
Snort	89.5%	95.2%	4.8%	10.5%	92.35%
FG-100A	94.49%	100%	0%	5.51%	97.25%
CSIPS-inbound	100%	96.8%	3.2%	0%	98.4%
CSIPS-outbound	100%	99.2%	0.8%	0%	99.6%
CSIPS-forwarded	100%	99.2%	0.8%	0%	99.6%
CSIPS-inner	100%	98.7%	1.3%	0%	99.35%

Chapter 5. Conclusion and Future work

This article proposes the architecture of CSIPS which uses CUSUM algorithm to detect DoS/DDoS attacks. In order to protect its management unit in the underlying united defense environment and important remote management units, two packet classification algorithms that distinguish in-bound, out-bound, forwarded and inner packets and five detection algorithms that detect DoS/DDoS attacks and/or logical attacks by invoking ordinary IDSs are proposed. Three experiments were performed to evaluate the CSIPS. We also final the best combination of N, h and α, which is also the best parameter values to detect resource/bandwidth consumption attacks. In third experiment, True Positive should not all 100% because initially when attacks start and number of accumulated packets do not exceed N, there attack packets are considered as normal packets. But we launch attacks continuously, so the results of True Positive are all 100%.

In the future, we would like to know when a type-1 Header Processor, type-2 Header Processor or an Intrusion Detector fails, how to increase a CSIPS's reliability? And, how to redistribute the original detection tasks to other ID-subsystems. We also like to derive a CSIPS's behavior and reliability models so users can predict a CSIPS's behavior and reliability before using it.

Reference

[1] J.S. Balasubramaniyan, J.O. Garcia-Fernandez, D. Isacoff, E. Spafford and D. Zamboni, “An Architecture for Intrusion Detection using Autonomous Agents,” the Annual Computer Security Applications Conference, pp.13-24, 1998.

[2] P.C. Chan and V. K. Wei, “Preemptive Distributed Intrusion Detection Using Mobile Agents,” the IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, pp.103-108, 2002.

[3] F.Y. Leu, J.C. Lin and M.C. Li, “A Performance-Based Grid Intrusion detection system,” the International Computer Software and Applications Conference, pp.525-530, 2005.

[4] F.Y. Leu and W.J. Yang, “Intrusion Detection with CUSUM for TCP-based DDoS,” the First IFIP Workshop on Trusted and Autonomic Ubiquitous and Embedded Systems, pp.1255-1264, 2005.

[5] A. Schulter, J.A. Reis, F. Koch and C.B. Westphall, “A Grid-based Intrusion Detection System,” the International Conference on Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies, pp.187, 2006.

[6] L.C. Wuu, Y.H. Chen, C.C. Ma and I.T. Lung, “A Practice of the Intrusion

Prevention System," the IEEE Region 10 Conference TENCON, pp.1-4, 2007.

[7] B.E. Brodsky and B.S.Darkhovsky, *Nonparametric Methods in Change Point Problems*, Kluwer Academic Publishers, 1993.

[8] P. Kannadiga and M. Zulkernine, "DIDMA: A Distributed Intrusion Detection System Using Mobile Agents," the International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks, pp.238-245, 2005.

[9] M. Ketel, "Applying the Mobile Agent Paradigm to Distributed Intrusion Detection in Wireless Sensor Networks," the Southeastern Symposium on System Theory, 99.74-78, 2008.

[10] J.G. Cao and G.P. Zheng, "Research on Distributed Intrusion Detection System based on Mobile Agent," the International Conference on Machine Learning and Cybernetics, pp.1394-1399, 2008.

[11] N. Patil, C. Das, S. Patankar and K. Pol, "Analysis of Distributed Intrusion Detection Systems using Mobile Agents," the First International Conference on Emerging Trends in Engineering and Technology, pp.1255-1260, 2008.

[12] F.Y. Leu, J.C. Lin, M.C. Li and C.T. Yang, "Detection Workload in a Dynamic Grid-based Intrusion Detection Environment," Journal of Parallel and Distributed

Computing, pp.427-442, April 2008.

[13] F.Y. Leu, K.W. Hu and F.C. Jiang, “Intrusion Detection and Identification System Using Data Mining and Forensic Technuques,” Information and Computer Security, Second International Workshop on Security (IWSEC’2007), pp.137-152, 2007.

[14] H. Wang, D. Zhang and K.G. Shin, “Change-Point Monitoring for the Detection of DoS Attacks,” IEEE Transactions on Dependable and Secure Computing, vol.1, no.4, pp.193-208, Oct-Dec. 2004.

[15] H. Wang, D. Zhang and K.G. Shin, “Detecting SYN Flooding Attacks,” Joint Conference of the IEEE Computer and Communications Societies, pp.1530-1539, 2002.

[16] B.P. Lim and M.S. Uddin, “ Statistical-Based SYN-Flooding Detection Using Programmable Network Processor,” the Third International Conference on information Technology and Applications, pp.465-470, 2005.

[17] J. Zhang, M. Zulkernine and A. Haque, “Random-Forests-Based Network Intrusion Detection Systems,” IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews, vol.38, no.5, pp.649-659, sept. 2008.

[18] J. Peng, C. Feng and J.W. Rozenblit, “A Hybrid Intrusion Detection and Visualization System,” the Annual IEEE International Symposium and Workshop on Engineering of Computer Based Systems, pp.505-506, 2006.

[19] K. Lu, Z. Chen, Z. Jin and J. Guo, “An Adaptive Real-Time Intrusion Detection System Using Sequences of System Call,” IEEE Canadian Conference on Electrical and Computer Engineering, pp.789-792, 2003.

[20] J. Liu and L. Li, “A Distributed Intrusion Detection System Based Agents,” the IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application, pp.553-557, 2008.

[21] Z. Q. Wang, H. Q. Wang, Q. Zhao and R. J. Zhang, “Research on Distributed Intrusion Detection System,” the International Conference on machine Learning and Cybernetics, pp.181-184, 2006.

[22] R. Koller, R. Rangaswami, J. Marrero, I. Hernandez and G. Smith, “Anatomy of a Real-time Intrusion Prevention System,” the International Conference on Autonomic Computing, pp.151-160, 2008.

[23] F.Y. Leu and T.Y. Yang, “A Host-based Real-Time Intrusion Detection System with Data Mining and Forensic Techniques, ” the International Carnahan Conference on Security Technology, pp.580-586, 2003.

[24] J.Z. Zhao and H.K. Huang, “ An Intrusion Detection System based on Data Mining and Immune Principles, ” the International Conference on Machine Learning and Cybernetics, pp.524-528, 2002.

[25] R. Oliver, “Countering SYN Flood Denial-of-Service Attacks,” Tech Mavens,

Inc., Aug. 2001. http://www.tech-mavens.com/synflood.htm