AN INFRASTRUCTURE TO DEFEND AGAINST DISTRIBUTED DENIAL OF SERVICE ATTACK

by

Wan, Kwok Kin Kalman

MSc in Information Technology
The Hong Kong Polytechnic University

June 2001

Abstract of dissertation entitled:
An Infrastructure
to defend against
Distributed Denial of Service Attack
submitted by Wan, Kwok Kin Kalman
for the degree of MSc in Information Technology
at The Hong Kong Polytechnic University in June 2001.

Distributed denial of service (DDoS) attack was first seen in early 1998. This kind of attack overwhelms a target server with an immense volume of useless traffic from distributed and coordinated attack sources. In February 2000, a number of the World's largest e-commerce sites were brought offline for days by this kind of attack, even though they were designed to offer high availability. The outages had caused a huge economic loss to both the victim sites and their users. The Internet industry widely agrees that there is essentially nothing a site can do with current technology to prevent itself from becoming a victim of DDoS attacks.

In this dissertation, I propose an infrastructure to solve this problem, which involves cooperation of different parties of the Internet to detect for DDoS attack and block the attack traffic well before the attack packets reach the target site. As a result, service interruption to the target can be minimized. This dissertation is the first research work I am aware of which proposes such an infrastructure in detail. It serves as a starting point to develop a complete solution against DDoS attack.

I propose to install full-blown local detection systems (FLDS) at various strategic locations in the Internet. They communicate in peer-to-peer mode to detect for and

respond to DDoS attack. Other nodes in the Internet can install minimal local detection systems (MLDS) to work with FLDSs to block attack traffic by hop-by-hop trace back of attack sources. Since an MLDS only responds to a confirmed attack, but does not detect for suspicious attack continuously, it requires much less computing resource than that of FLDS.

An analysis of the Internet topology and *traceroute* results both show that backbone ISP gateway nodes, Internet Exchanges, and Network Access Points are suitable Internet nodes to install FLDSs because most cross-domain Internet traffic passes through these locations. Since the number of these locations is relatively small, the proposed system should be a cost-effective solution to the DDoS problem.

I propose a component approach to design a LDS based on the Common Intrusion Detection Framework (CIDF). CIDF identifies four components for a typical intrusion detection system. They are E-box, A-box, D-box, and R-box. I add four more components to fulfill the specific requirements for handling DDoS attacks. They are S-box, P-box, M-box, and C-box.

S-box is a load-balancing component which distributes IP datagrams sniffed or mirrored from the network switch at an Internet node to E-boxes and P-boxes for attack detection. This load-balancing feature is very important because the traffic volume flowing through a strategic location is very high. Therefore a FLDS must be highly scalable. The proposed S-box fulfills this requirement.

E-box detects control packets of DDoS attack tools by matching IP datagrams with

DDoS attack signatures. These control packets can provide clues to trace back to the actual attacker and provide evidence for later law enforcement. I also propose a new approach to specify and store attack signatures based on XML.

P-box detects for suspicious DDoS attack, and suspicious interface where attack traffic comes from, based on traffic volume anomalies. Algorithms are designed to cater for the dynamic nature of traffic volume to different destinations and can adjust the threshold levels of traffic volume anomalies accordingly.

A-box integrates information gathered by E-box, P-box, and remote LDSs to decide whether there is any DDoS attack in progess. The decision rules are designed to avoid false alarm but at the same time can detect DDoS attack effectively.

D-box is a XML server which stores alerts and component statuses in XML format. It supports XML query language such that the stored information can be retrieved using standard query language.

R-box uses three methods to block attack traffic, namely traffic rate limit filter, upstream LDS alert, and edge router ingress filter. It applies traffic rate limit filters to limit traffic destined for a confirmed victim at confirmed inbound interfaces of attack traffic. It also alerts an upstream LDS to detect for and respond to the attack, such that a hop-by-hop trace back of attack sources can be achieved. For a LDS at local ISP, its R-box can instruct the edge routers to install ingress filters to filter attack packets with spoofed source addresses received from the customer sites.

M-box is the only component of a LDS with Internet connection. It communicates alert and status information with other LDSs through the Internet. It maintains a list of immediate neighbor LDSs, such that alerts and heartbeats are sent to all immediate neighbors from time to time. Alerts received from a neighbor are forwarded to the A-box for analysis and to this LDS's immediate neighbor LDSs. This propagation mechanism ensures that all LDSs promptly receive the same information. The protocol also caters for failure of individual LDS.

C-box is the console for configuring various components and querying for alert and status information.

I adopt and extend the Intrusion Detection Message Exchange Format (IDMEF) as the communication language among different components in a LDS and among different LDSs. Intrusion Alert Protocol (IAP) is used as the transport protocol which provides secure communication of IDMEF messages.

I also integrate all components into a LDS network such that it can be treated as a single component to plug into the network of an Internet node.

Finally, by running simulations, I find that the proposed system is very effective in detecting DDoS attack. Moreover, the false alarm rate is very low (actually, there is no confirmed false alarm in 60 independent simulation runs). The proposed system also increases the number of normal packets that the victim can receive and process when there is an ongoing DDoS attack. This effect is particularly significant when the daemon coverage (% of network nodes with daemon) is between 2.5% and 12%. It

also avoids traffic congestion to the Internet as a whole when there is a large scale DDoS attack, as the proposed system can block over 85% of attack traffic early in the network.

ACKNOWLEDGMENTS

My many and special thanks go to my supervisor Dr. Rocky K. C. Chang for his valuable advice, comment and ideas. I would also like to thank my co-examiner Dr. John W. T. Lee. And last, but not least, for the support from my family, friends and colleagues.

Table of Contents

List of Figures

List of Tables

Chapter 1
INTRODUCTION

1.1. Introduction

Distributed denial of service (DDoS) attack was first seen in early 1998 [1]. In February 2000, a number of the World's largest e-commerce sites* were brought offline for days by this kind of attack, even though they were designed to offer high availability. The outages had caused a huge economic loss to both the victim sites and their users.

DDoS attack is based on IP-based DoS attack. IP-based DoS attack exhausts a target server's network or system resources by an immense volume of traffic. It prevents legitimate users from accessing the server. This type of attack is not new. In 1996, DoS attacks using UDP packet storm [2] and TCP SYN flooding [3] were reported. In 1998, another type of IP-based DoS attack called "Smurf" appeared [4]. Other types of DoS attacks were also reported from time to time.

In the old days, DoS was considered as a less critical security breach because it did not involve any leakage or loss of data. However, this argument is no longer valid with the emergence of e-commerce. This new generation of companies only have web-presence for doing business. To them, denial of service means "denial of business". Both the loss of revenue and loss of credibility resulted by such attacks are fatal. Imagine the consequences if your bank or securities house cannot offer any

* The Web sites that were brought down by the series of DDoS attacks in February 2000 included Yahoo.com, Amazon.com, Excite, E*Trade, eBay, CNN.com, Buy.com, and ZDNet.

services to you for a whole day.

DDoS attack is an extension to DoS attack. It deploys a large number of “zombies” distributed at various locations in the Internet to launch DoS attacks against a target server simultaneously.

In a DDoS attack, attacker begins by scanning thousands of machines connected to the Internet and looks for unprotected ports, vulnerable services, and other weaknesses that will let them gain root access. After gaining root access, the attacker can then install daemons on these intermediate machines called “zombies”. The hundred or even thousand daemons distributed at different locations in the Internet then quietly listen to network traffic. Once the command is received from the attacker, all the “zombies” will start the DDoS assault against the final victim site.

The processes of discovering vulnerable sites, compromising them, installing daemons, and concealing the intrusion can be performed in batch mode against many machines using automated tools. Moreover, as more and more machines are connected to the Internet, and most of them are vulnerable to compromise but have fast network connectivity, it will be easier for attacker to initiate DDoS attacks.

According to the Computer Emergency Response Team (CERT) in 1999, even though an organization may able to harden its own systems to prevent implantation of the daemons by attacker, there is essentially nothing a site can do with current technology to prevent becoming a victim of DDoS attack [5]. Firewall and Intrusion Detection System at victim site are useless against DDoS attacks because the site’s network

bandwidth is already exhausted when the attack traffic reaches the site. The huge volume of attack traffic will also paralyze the firewall by exhausting its system resources.

Therefore, an infrastructure, which involve cooperation of different parties of the Internet is urgently in need, such that this kind of attack can be handled automatically in real-time. The infrastructure should block the attack traffic well before the attack packets reach the target site to minimize or even avoid service interruption to the target.

1.2. Project Objective

The objective of this dissertation is to propose an infrastructure which involves cooperation of different parties of the Internet to protect the Internet from DDoS attack. The infrastructure should meet the following requirements:

1. can stop a DDoS attack quickly enough such that service interruption to the victim site can be minimized;
2. should be cost effective;
3. can detect general DDoS attack, rather than specific implementations of DDoS attack;
4. can achieve a low false alarm rate to avoid blocking of normal traffic;
5. can trace the actual attacker, and provide evidence for later law enforcement; and
6. can survive and function even the infrastructure itself is also under attack.

The infrastructure serves as a starting point to solve the DDoS attack problem. Based on this infrastructure, different DDoS attack detection and response algorithms can be incorporated. A complete solution for the DDoS attack problem can then be achieved.

1.3. Project Scope

Only DDoS attack, which is based on network-based flooding DoS attack is covered in this study. It includes DoS attack which exhaust a system's network bandwidth or other resources by flooding of IP packets. Therefore, DoS attack based on UDP, ICMP, and TCP packet flood are included in the scope of this dissertation, as they attack a victim by exhausting its network bandwidth with a huge volume of traffic. TCP SYN flood is also included in the scope, even though it does not need to exhaust a site's network bandwidth to paralyze it. Instead, it exhausts a site's buffer by numerous half open TCP connections (see next section for details). Other flooding attacks, which involve abnormally large volume of a particular type of traffic are also included in the scope. The abnormality is measured in relative term, rather than in absolute term (e.g. the case of TCP SYN flood).

Moreover, this study focuses on DDoS attack with attacking hosts distributed at different Autonomous Systems (AS). Cross AS attack is particularly difficult to handle because it allows more attacking hosts and involves multiple administrative domains.

This study does not consider DoS attacks that crash or slow down a system by exploiting system or application security holes. This kind of DoS attacks can be defended effectively by patching a host's operating system for such security vulnerabilities.

1.4. Organization of this Dissertation

This paper is composed of six chapters. Chapter 1, or this chapter, provides background information about DoS and DDoS attacks. Some common types of DoS and DDoS attacks are discussed. Current techniques to defend against these attacks are also described. Then I shall explain why the current solutions are insufficient to solve the problem, so that a new solution is required to defend against DDoS attacks.

Chapter 2 proposes the high-level infrastructure of a mechanism to defend against DDoS attack. It is an Internet-wide defense system which is composed of multiple local detection systems positioned at different strategic locations of the Internet. These local detection systems communicate with each other to detect and respond to DDoS attack.

Chapter 3 and Chapter 4 focus on the design of a local detection system. In these two chapters, I shall explain the local detection system's architecture and component design. Some suggestions on the attack detection and response algorithms will be made. However, since the objective of this paper is to propose an infrastructure which serves as a starting point to solve the DDoS attack problem, rather than a complete solution, the algorithms need not be perfect. Detection and response algorithms from future researches can be incorporated into the infrastructure to provide the complete solution.

Chapter 5 shows the effectiveness of the proposed infrastructure against DDoS attack based on software simulation. I shall use software program to simulate a large

network with several hundred network nodes. These network nodes represent the network nodes of different parties in the Internet. Some of them contain DDoS daemons such that attack traffic will be originated from them to a victim node. I shall show what is the difference between the cases that with and without the proposed infrastructure in the network.

Chapter 6 identifies the further researches required to enhance the infrastructure such that a complete solution to DDoS attack problem can be achieved. It also contains the conclusion of this dissertation.

1.5. Common Types of DoS Attack

In this section, I shall describe several common IP-based DoS attacks, including TCP SYN flood, UDP flood, and Smurf attack. These attack methods are commonly used by existing DDoS attack tools, which will be discussed in the next section. There are other common IP-based DoS attacks not included in the following description, e.g. Teardrop, Land, and Ping of Death. It is because they attack target servers by exploiting implementation bugs of the servers' TCP/IP stacks with a few malformed IP packets, rather than saturating the target servers' network or other system resources by a large volume of dummy traffic. Therefore they are out of our scope and are not considered here.

1.5.1. TCP SYN Flood

In a TCP SYN Flood attack, the attacker begins by sending a SYN message to the server (victim) with spoofed non-existing source IP address. The server then acknowledges the SYN message by sending SYN-ACK message to the spoofed IP address. Because the spoofed source actually does not exist, the server will not receive the expected ACK message and the half-open connection will be pending until time-out. By repeating this action, the attacker will successfully exhaust the server's system memory as data structures are required to store information of all half-open connections. Therefore, no more incoming connections will be allowed even they are legitimate.

1.5.2. UDP Port DoS Attack

In a UDP Port DoS attack, an attacker hooks up one system's UDP chargen service with another system's UDP echo service by spoofing. Both services are designed for testing network programs. Chargen service generates series of characters for each packet it receives. Echo service echoes any character it receives. By hooking up these two services, an attacker can produce a nonstop flood of useless data passes between the two systems. Therefore network congestion and even system outages will result.

1.5.3. Smurf DoS Attack

In a Smurf attack, the attacker sends forged ICMP echo request packets ("ping" packets) to "intermediary" networks with destination address equals to IP broadcast address and spoofed source address equals to the victim's address. All the hosts that have received the echo request will send replies to the victim. The huge volume of echo replies will cause network congestion or outages to the victim. This multiplying effect allows an attacker with small network bandwidth to bring down a target server with much larger network bandwidth.

1.6. Common Types of DDoS Attack

The first generation of DDoS attack tools includes Trinoo and Tribe Flood Network (TFN). They spawned the next generation of tools called Tribe Flood Network 2000 (TFN2K) and Stacheldraht (German for Barb Wire) [6]. I shall discuss these four DDoS tools in this section. However, it should be noted that there are other tools of DDoS attack as they are still evolving, with enhancements both in the attack approach and the attacker's identity concealing approach.

1.6.1. Trinoo (or Trin00)

Trinoo is a distributed tool used to launch coordinated UDP flood DoS attacks (flood of UDP packets to a random port for overloading the victim host and network) from many sources. A trinoo attack consists of a small number of masters and a large number of daemons installed on machines compromised by the attacker. A master stores a list of known daemons in an encrypted file named "…". An attacker after be authenticated, instructs a master to launch an attack against one or more IP addresses. The master then communicates with the daemons giving instructions to attack the specified addresses for a specified period of time.

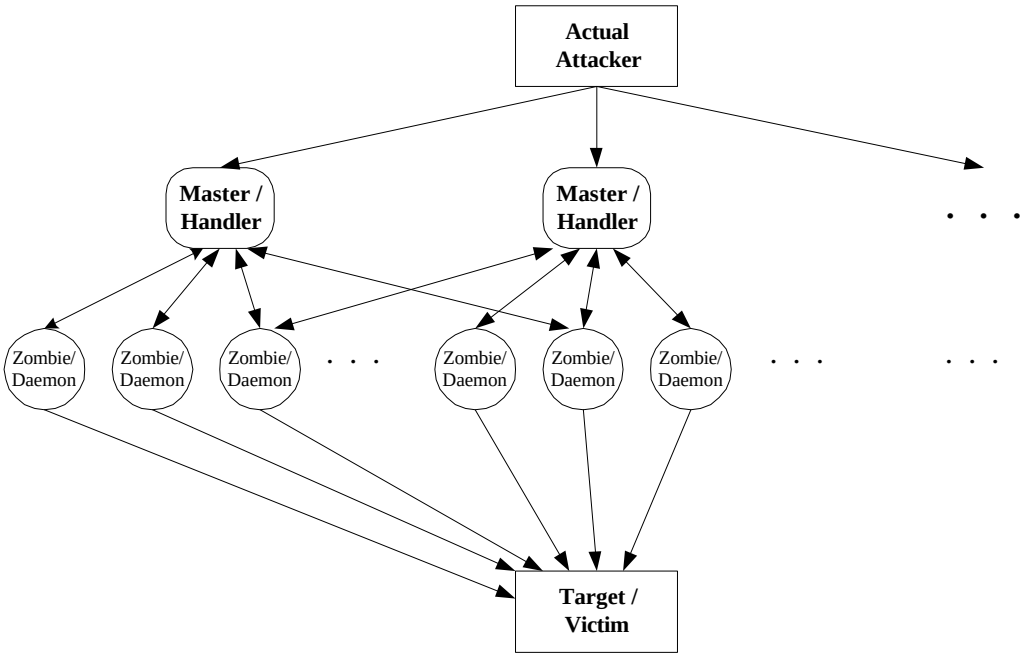


Figure 1.1: Architecture of a typical Distributed Denial of Service Attack

1.6.2. Tribe FloodNet (TFN)

The operation of TFN is similar to that of trinoo. However, TFN can initiate several types of DoS attack beside UDP flood attack, including TCP SYN flood, ICMP echo request flood, and ICMP directed broadcast (e.g. smurf). The daemons also have the capability to generate packets with spoofed source IP addresses.

The master communicates with the daemons using ICMP echo reply packets with 16 bit binary values embedded in the ID field, and any arguments embedded in the data portion of the packet. The binary values, which are definable at compile time, represent the various instructions sent between TFN masters and daemons.

1.6.3. TFN2K

TFN2K is similar to TFN. However, it includes features designed specifically to make TFN2K traffic difficult to recognize and filter. It transports TFN2K traffic over multiple transport protocols including UDP, TCP, and ICMP. It has features to confuse attempts to locate other nodes in a TFN2K network by sending “decoy” packets. In networks that employ ingress filtering, TFN2K can forge packets that appear to come from neighboring machines. It also includes strong encryption for control packets. Beside flood attack, TFN2K can also crash or introduce instabilities in systems by sending malformed or invalid packets (Targa3 attack).

1.6.4. Stacheldraht

Stacheldraht combines features of TFN and trinoo. Like TFN, stacheldraht can spoof source addresses. It can test to see if ingress filtering is in place, and if so, it only spoofs the lowest bits of the addresses. It uses encrypted TCP packets to communicate control packets. Moreover, it also adds automated remote update of the daemons.

1.7. Current Techniques to defend against DDoS Attack

As explained in the last section, in a typical DDoS attack, there are at least five parties involved. They are the actual attacker or the intruder, master, zombie or daemon, ISPs or transit Autonomous Systems (AS) which transmit the attack traffic from the zombies to the target site, and the target or the victim site. The current techniques to defend against DDoS attack can be classified according to the parties concerned. In this section, I shall describe the techniques to defend a host from becoming a master or becoming a zombie, the techniques to defend an ISP from becoming a transit provider of an attack, and the techniques to defend a host from becoming a victim site. Finally, I shall discuss the insufficiencies of these solutions.

1.7.1. Defend a Host from becoming a Master or Zombie

Basically, a system administrator should make sure three things in order to prevent the hosts he managed from becoming a master or zombie in a DDoS attack.

The first line of defense is to remove any security vulnerabilities from the hosts. In this way, he can avoid the hosts from being compromised and having master or daemons been implanted by intruder. It includes but not limits to the followings:

1. enforce strong password;
2. configure software in a secure manner;
3. turn off services that are not required;
4. install and properly configure firewall and intrusion detection system to prevent and detect intrusion;

5. keep the systems up to date on patches; and
6. scan the systems for security vulnerabilities and fix them from time to time.

The second line of defense is to scan the host from time to time to detect for master or daemons implanted on them. Since DDoS tools are evolving, the signatures used by the detection tools should also be updated frequently to keep up with the development of the attack tools.

The final line of defense is to detect for any DDoS attack originated from the masters or daemons on the hosts. It can be performed by intrusion detection system which detects for control messages of DDoS attack, packets with spoofed source addresses, and abnormal traffic of TCP SYN packets, ICMP echo request, UDP flood, etc. Boundary router can also be configured to restrict the outbound traffic rate of those packets (egress filter). Once an attack is detected, the master or daemons should be found out and killed as soon as possible.

1.7.2. Defend an ISP from becoming an Attack Traffic Carrier

There are two things that an ISP can perform to defend itself from becoming a DDoS attack traffic carrier. Firstly, it can install ingress filters in the edge routers connecting to customer sites. Ingress filter prevents influx of packets with spoofed source IP addresses from the customer sites. Only packets with source addresses that are valid in the customer networks are allowed to pass through the filters into the ISP’s network.

The second thing is to install egress filters in the edge routers connecting to other

Autonomous Systems. Similar to the case in Section 1.7.1, egress filter identifies packet streams with spoofed source addresses, abnormal traffic of TCP SYN packets, ICMP echo request, UDP flood, etc., and block them from leaving the ISP's network.

1.7.3. Defend a Host from becoming a Victim

One possible defense method that a victim can use to defend against DDoS attack is moving target defense. When it is detected that a host is under attack, the host changes its IP address. Since most DDoS attack tools set the target using IP address, the change will cause the remainder of the attack packets to be delivered to the old, now invalid IP address. The change can be done by updating the relevant DNS entries and the routing table entries in the Internet.

Another defense method is filtering defense. It uses high throughput packet filter to filter the flood packets. Since most DDoS attacks use randomly generated spoofed source IP address, one filtering option is to reject the first IP packet from any IP address. This would work for TCP-based servers (e.g. Web servers) because normal TCP packet will be resent after timeout. For attack traffic, because every packet has different source IP address, all of them will be filtered.

Bandwidth defense utilizes large network bandwidth and large distributed networks to provide enough bandwidth to survive an attack.

Besides, system administrator should turn off echo and chargen services on his hosts unless there is a specific need. It protects the system from UDP Port DoS attack. Network directed broadcast should also be blocked to defend against Smurf attack.

1.7.4. Problems of the Current Solutions

Defend from becoming a Master or Zombie

Although there are a number of effective ways to defend a host from becoming a master or zombie, it will not reduce the number of potential masters or zombies in the Internet. It is because such techniques involve high software cost or labour cost for system monitoring and configuration. Individual users or small companies connecting to the Internet cannot afford to perform those defense actions.

Even large corporations connecting to the Internet may not have the incentive to specifically defend their hosts from becoming masters or zombies. The reason is the damage to the master or zombie computer is negligible. Only some network bandwidth is stolen from them. Moreover, Internet access is usually charged on flat monthly basis, no extra Internet access fee is required even if a lot of attack traffic is generated from the zombies [10].

Defend from becoming an Attack Traffic Carrier

Ingress filter and egress filter are effective ways to block DDoS attacks which use spoofed IP addresses. Unfortunately, they are seldom used by ISPs because they will degrade the ISP’s network performance significantly. A research shows that less than 8% of ISP are filtering source IP address [10]. Moreover, it is not cost effective to have all ISPs to install the filters for preventing DDoS attacks.

Moreover, some DDoS attack tools, e.g. TFN2K and Stacheldraht, can bypass ingress filters by spoofing the source address to a valid neighbor IP address. With the emerging of such sophisticated attack tools, only ingress and egress filters are not enough to stop all kinds of DDoS attacks, even though they can simplify the trace back of attack traffic.

Defend from becoming a Victim

It is much more difficult to defend a host from becoming a victim than from becoming a zombie or an attack traffic carrier, while the loss resulted is much higher. All the existing solutions in this arena are adhoc and not effective.

The moving target defense only works against the existing attack tools. Once the intruders are aware of this defense method, they have no difficulty to modify the tools to add DNS tracing function. With such function, the enhanced attack tool can find out if the victim has changed its IP address and attack the new address accordingly.

The filtering defense is only effective against small scale DDoS attack. For large scale DDoS attack, no matter how fast the packet filter can filter the traffic, the network bandwidth will be exhausted as the number of zombie increases. The bandwidth defense has the same problem. Therefore, none of them is a good solution against DDoS attack.

Conclusion

As a conclusion, there is no way to get rid of all zombies, and it is impossible to

handle a DDoS attack at the victim site. Therefore, we need a mechanism to block the attack traffic when it is still being transmitted in the Internet, well before the attack packets arrive at the victim site. This Internet-wide mechanism is the focus of this dissertation.

1.7.5. Other Related Researches

To my knowledge, there is no other research works which propose an Internet-wide infrastructure to defend against DDoS attack in detail. However, there are related researches such as design of distributed intrusion detection system (IDS), DoS resistant IDS, and trace back of IP traffic to find intruder.

EMERALD (Event Monitoring Enabling Responses to Anomalous Live Disturbances) is an IDS research tool developed by SRI International [11]. A major goal of EMERALD is to address intrusion detection issues associated with large, loosely coupled enterprise networks. EMERALD uses a hierarchical approach to provide three levels of analysis performed by a three-tiered system of monitors: service monitors, domain monitors, and enterprise monitors. These monitors have the same basic architecture, which contains a set of profiler engines (for anomaly detection), signature engines (for signature recognition), and a resolver component that integrates the results generated from the engines. Service monitors detect intrusion for individual components within one domain. Domain monitors integrate information from the service monitors to provide a domain-wide view of intrusions, while the enterprise monitors perform inter-domain analysis to assess intrusion from a global perspective.

NetSTAT is an IDS research tool produced by the University of California at Santa Barbara [12]. It aims at real-time network-based intrusion detection in complex networks composed of several sub-networks. NetSTAT is composed of a set of probes that are responsible for detecting intrusions in the sub-networks to which the probes are attached. If an intrusion component is detected, then an event can be forwarded to other interested probes that subscribe to that event in order to get a more complete understanding of the intrusion. Thus, intrusion that involves separate sub-networks can be identified.

Another distributed IDS research makes use of a set of mobile agents to detect for intrusion [13]. The agents act independently and move through the network. Each agent observes part of the network, advises each other via messages when an action is considered suspect, and engages in reactive actions. Beside the above three researches, there are also other researches on design of distributed IDS [14][15].

Since IDS itself may also be attacked by flooding DoS attack, there are some researches on how to design an IDS architecture that is resistant to DoS attacks. One proposal is to frustrate attackers by making IDS components invisible to attackers' normal means of "seeing" in a network. Upon a successful attack, the architecture allows IDS components to relocate from attacked hosts to operational hosts thereby mitigating the attack [16].

Sources of DDoS attacks are difficult to trace because attackers usually use spoofed IP source addresses. Some researches are targeted to solve this trace back problem.

One proposal is to probabilistically mark packets with partial path information as they arrive at routers. This approach exploits the observation that attacks generally comprise a large number of packets. While each marked packet represents only a “sample” of the path it has traversed, the complete path can be reconstructed by using a modest number of such packets [17].

Although there are a number of related researches, none of them target to solve the DDoS attack problem fundamentally. Therefore, this dissertation is an important step to solve this problem. This dissertation proposes an Internet-wide infrastructure to detect for DDoS attack, block the attack traffic to minimize service interruption to the victim, and collect evidence to trace the actual attackers. In the process of designing the infrastructure of distributed detection and response system against DDoS attacks, the above researches are referenced, in particular the researches about distributed IDS design.

1.8. Contribution of this Project

In this project, I describe the mechanisms of different types of IP-based DoS attacks, and DDoS attacks. Based on the properties of DDoS attack, I explain why it is so difficult to handle. Different approaches to defend against DDoS attack are also studied. I analyze the limitations of different defense techniques. My conclusion is that there is no way to get rid of DDoS daemons in the Internet, and there is no effective solution to handle DDoS attack once it is started, based on the current technology.

In order to solve the problem, I propose an infrastructure that involves cooperation of different parties of the Internet. I also lay down the requirements of this infrastructure.

The proposed infrastructure consists of distributed full-blown local detection systems (FLDSs) and minimal local detection systems (MLDSs) in the Internet to solve the DDoS attack problem cooperatively. FLDS both detects for and responds to DDoS attack, which consumes more computing resources. MLDS only responds to DDoS attack and therefore consumes less computing resource. FLDSs should be installed at strategic locations in the Internet where most cross-domain traffic passes through, while MLDSs should be installed at other Internet nodes.

I study the Internet architecture and perform *traceroutes* to identify locations to install FLDSs. It is found that over 99% of cross-domain routes pass through at least one backbone ISP, Internet Exchange, or Network Access Point. Therefore, installation of FLDSs at these strategic locations can monitor most cross-domain traffic and detect

for suspicious DDoS attack. Since the number of these strategic locations is relatively small, the proposed system only requires installation of a few FLDSs, and therefore should be a cost-effective solution to the DDoS problem.

I also study the network structure of a typical Internet Exchange and backbone ISP to suggest how we can integrate LDSs into their networks. Furthermore, I propose the high level design of a LDS, which is a component-approach based on the Common Intrusion Detection Framework (CIDF), with enhancements.

I identify different components that are required in a local detection system. Altogether eight components are designed, including S-box, E-box, P-box, A-box, D-box, R-box, M-box, and C-box.

I propose to use S-box to distribute IP datagrams sniffed or mirrored from the network switch at an Internet node to E-boxes and P-boxes for attack detection. It solves the problem of analyzing traffic with very high transmission rates at the strategic locations where FLDSs should be installed. Traffic distribution rules that can distribute traffic evenly and facilitate merging of information captured by different devices are proposed.

For the detection design, I suggest to use both misuse detection and anomaly detection to detect for suspicious attack. In particular, traffic volume anomaly is an important indicator of DDoS attack. E-box and P-box are responsible for misuse detection and anomaly detection respectively. I propose every detection result made by E-box or P-box should have a confidence level.

For E-box, I suggest to detect control packets of DDoS attack tools by signature recognition. These control packets can provide clues to trace back the actual attacker and provide evidence for later law enforcement. I also propose a new approach to specify and store attack signature based on XML. How DDoS attack signatures can be prepared using this new approach is suggested.

For P-box, I propose mechanisms to detect for suspicious DDoS attack, and suspicious interface where attack traffic comes from, based on traffic volume anomalies. The mechanisms cater for the dynamic nature of traffic volume to different destinations and can adjust the threshold levels of traffic volume anomalies accordingly.

A-box is responsible for decision making, or confirming whether a DDoS attack is in progress. I propose how it can integrate information detected by E-box, P-box, and remote LDSs by consolidating the confidence levels of the local and remote detection results. The decision rules are designed to avoid false alarm but at the same time can detect DDoS attack effectively.

R-box is responsible for taking response action against confirmed DDoS attack. I propose to detect which interfaces the attack traffic comes from based on traffic volume anomalies once there is a confirmed attack. The traffic destined to the victim from the suspected interface should then be blocked. I propose three methods to block the traffic, namely traffic rate limit filter, upstream LDS alert, and edge router ingress filter.

R-box can apply traffic rate limit filter to limit traffic destined for a confirmed victim at confirmed inbound interfaces of attack traffic. It can also alert upstream LDS to detect for and respond to the attack, such that a hop-by-hop trace back of attack sources can be achieved. For LDS at local ISP, its R-box can instruct the edge routers to install ingress filter to filter attack packets from the customer sites with spoofed source addresses.

For the communication design, I adopt the Intrusion Detection Message Exchange Format (IDMEF) as the communication language and Intrusion Alert Protocol (IAP) as the transport protocol for communication among different LDS components and different LDSs. I extend IDMEF to specify DDoS attack-related alerts. Moreover, I design a simple network protocol such that messages from a LDS can reach every other LDS, even the LDS does not know the existence of the other. Failure of any LDS will not affect the communication. Therefore the infrastructure can function even some LDSs are brought down by an attack.

I also integrate everything into a complete picture of a LDS, together with its relationship with other LDSs in the global defense system.

Furthermore, I simulate an Internet-like network and try to find out whether the proposed global defense system can successfully detect a DDoS attack. I also try to find out how a DDoS attack can affect the volume of legitimate traffic that can arrive at and be processed by the victim, both with and without the proposed global defense system.

I find that the proposed system is very effective in detecting DDoS attack. In all the simulated runs, the attack is detected and confirmed by the system within a short period. Moreover, there is no confirmed false alarm against non-victim node in all simulation runs.

The simulation also shows that the global defense system can successfully increase the number of normal packets that the victim can receive and process when there is a DDoS attack. The effect is particularly significant when the daemon coverage is between 2.5% to 12.5%. This effect is less significant when the daemon coverage increases to above 20% because both normal and attack packets to the victim are dropped by the LDSs proportionally in such situations. However, the defense system still can block a large number of attack packets at an early time such that they cannot compete for Internet bandwidth with other legitimate traffic, which would cause traffic congestion to the backbone otherwise.

I also explain how the simulation result can be used to project the effectiveness of the system in actual deployment. The conclusion is that even better result can be achieved because of the difference between the simulated network and the actual Internet.

Finally, I identify the future works that are required for providing a complete solution to the DDoS attack problem based on the proposed infrastructure.

Chapter 2
INFRASTRUCTURE OF INTERNET-WIDE DDoS ATTACK DEFENSE SYSTEM

2.1. Infrastructure Overview

In order to solve the DDoS attack problem, I propose an infrastructure which involves installation of a number of monitors or local detection systems (LDS) at various strategic locations in the Internet to detect for and respond to DDoS attacks.

LDSs at different locations communicate in a peer-to-peer approach. It prevents single point of failure and avoids creating bottleneck in the global defense system. This point is very important as the LDSs themselves will surely be the first targets of a DDoS attack if an intruder wants to bring down other sites.

When a suspicious attack is detected, the LDS that detects the suspicious attack will communicate this information with LDSs at other locations. Based on both the local and remote information, a detection system can make decision to determine whether a DDoS attack is in progress.

If a detection system confirms that there is a DDoS attack, it can take response actions to terminate the attack. It can ask the local routers to block the attack traffic from the inbound interfaces. It can also communicate with the upstream detection systems about the attack. The upstream detection systems can then perform the same response actions. The process reiterates until the ISPs closest to the attack sources block the attack traffic directly from the sources.

In summary, the following processes are performed by a LDS:

1. detect suspicious DDoS attack;
2. communicate with other LDSs;
3. decide whether a suspicious attack is a real attack; and
4. respond to a confirmed attack.

This proposal suggests that all the above functions should be performed by LDSs in the network itself. An alternative approach is to let the victim site to detect and confirm if itself is under DDoS attack, and inform the global defense system about the attack. The global defense system is only responsible to trace back and stop the attack. This approach frees the global defense system from continuously monitoring the network traffic for attack. However, this approach has a number of weaknesses which make it less favorable than the former approach. The weaknesses are as follows:

1. When a site is under attack, it is heavily loaded and may even malfunction. It may not be able to inform other parties about the DDoS attack promptly. Human intervention may be required. Therefore the site's services will be interrupted.
2. There are millions of sites all over the World. It is not cost effective to require every site to install a DDoS attack detection system. Moreover, victim of DDoS attack is not limited to Web site, FTP site, etc. It can be any IP node.
3. If a site fails to detect an attack, this attack will not be known by the global defense system. Therefore the attack can continue and create a lot of dummy traffic. This traffic will also adversely affect the performance of other sites as the intermediate paths between the attack sources and the victim are also overloaded.

Because of the above weaknesses, I shall only focus on the former approach, i.e. detect DDoS attack by distributed local detection systems in the Internet. Nevertheless, the proposed infrastructure is flexible enough such that it can implement the later approach, i.e. detect DDoS attack by victim sites. It is because the main difference between the two approaches only involves different locations for placing the full-blown local detection systems (see next section for details of full-blown local detection system). The former approach places the full-blown systems in the Internet, while the later approach places the full-blown detection system in the network end nodes. The infrastructure and the detection system design are not affected.

In the next several sections, I shall discuss where LDS should be installed and how it can fit into the existing network structure. Then, I shall describe the high level design of LDS. It includes the basic design, detection design, communication design, decision making design, and response design.

2.2. Location of LDS in the Internet

According to the infrastructure overview, a number of LDSs should be installed at various locations in the Internet. In this section, I shall further categorize local detection system into two categories, minimal local detection system (MLDS), and full-blown local detection system (FLDS). I shall explain where MLDS and FLDS should be placed in the Internet to achieve their objectives.

2.2.1. Minimal Local Detection System (MLDS)

Ideally, all network nodes in the Internet should install local detection systems such that all Internet traffic can be monitored by some detection systems. It ensures no attack can escape from the global defense system. Besides for detecting attack, having a local detection system at every ISP also ensures attack traffic can be blocked and traced back hop-by-hop to the source of the attack.

However, a full-blown local detection system is very resource demanding (in terms of CPU, memory, and I/O), as it needs to analyse every packet passing through the network node where the system is situated. Therefore, this arrangement is not a cost-effective solution. A better alternative is that a normal network node only needs to maintain a minimal local detection system (MLDS). A MLDS only performs the functions that are absolutely necessary for normal network node in the global defense system. These necessary functions include the followings:

1. receive attack information detected by other LDS;
2. decide whether there is an attack based on the received information;

3. find out the inbound interfaces of the attack traffic given the victim address is known;
4. ask the local routers to block the attack traffic; and
5. inform the upstream LDSs about the attack.

With the capability of performing the above functions, a MLDS can effectively stop attack traffic and trace the attack sources. In addition, a MLDS is inexpensive to implement because it will be waken only if there is a confirmed attack. Moreover, since it will know the victim's IP address, it only needs to process packets related to the victim's address. Therefore, the system resources required is much lower than a full-blown detection system.

2.2.2. Full-Blown Local Detection System (FLDS)

A full-blown local detection system can perform all functions of a minimal local detection system. Moreover, it monitors the network traffic continuously to detect for suspicious DDoS attacks. It communicates the detected information with other local detection systems, such that every LDS (both MLDS and FLDS) can confirm whether an attack is in progress, based on the detected information from all FLDSs, and take the corresponding response actions.

Because a FLDS requires much more system resources to work, it should only be located at strategic locations of the Internet where most cross-domain traffic will pass through. Therefore traffic from sources of DDoS attack at various locations in the Internet, which may span multiple domains, can be readily detected.

Hierarchical Network Structure

If the Internet topology is a strict hierarchical structure, locating FLDSs at the nodes on the top-level hierarchy can satisfy the above requirements. It is because the top-level nodes switch most inter-domain traffic. Therefore, only a few FLDSs are required to set up. As a result, the proposed infrastructure is very effective for strict hierarchical network structure.

The NFSNET-based Internet in 1980’s was a strict hierarchical structure. At that time, the Internet had a three-tiered hierarchical network architecture. The architecture connected campuses and research organizations to regional networks, which in turn

connected to the NFSNET backbone linking six nationally funded super-computer centres by 56kbps leased line in 1986. Therefore, the nodes on the NFSNET backbone would route all cross-regional traffic. As a result, by locating FLDSs at the backbone nodes, all cross-regional DDoS attacks could be detected.

Fully Mesh Network Structure

Another extreme of the Internet topology is a fully mesh structure, where every ISP is directly connected to all other ISPs in the Internet. In this situation, basically every ISP is required to have a FLDS in order to detect DDoS attack since there is no strategic locations where most Internet traffic would pass through. Therefore the proposed infrastructure is not effective if the Internet topology is a highly mesh structure.

Loose Hierarchical Network Structure

Fortunately, the existing Internet topology is a loose hierarchical structure, which is much closer to the hierarchical-end of the spectrum rather than the fully-mesh-end of the spectrum.

In 1993, the National Science Foundation (NSF) issued a solicitation for bids to built Network Access Points (NAPs) where major ISPs connect their networks and exchange traffic. In this way, anyone could develop a national backbone for the connection of LANs, sell connectivity to itself, and use the NAPs as physical points to exchange traffic with all other service providers. To facilitate intra-regional traffic exchange, some regions have also built regional Internet Exchanges which

interconnects ISPs of the same regions such that intra-regional traffic do not need to route via backbone ISPs. Most recently, some backbone ISPs have begun to increase the amount of “private peering” they do between themselves, i.e. interconnecting their backbones without going through the NAPs.

Though the Internet is moving from a core or hierarchical network (NFSNET-based) to a more distributed architecture, it still has a loose hierarchical structure. There are only a small number of backbone ISPs which operate extensive high-speed cross-regional backbone networks. Moreover, this number is not going to increase a lot because of economy of scale. Therefore we can still consider the Internet as a hierarchy of networks.

The uppermost layer of the loose hierarchy consists of the NAPs which switch traffic among different backbone ISPs. There are eight major NAPs currently and they are listed in Appendix A. The next layer consists of backbone ISPs. There are approximately 50 major Internet backbones and are listed in Appendix B. Most backbone ISPs are connected to multiple NAPs. Therefore some backbone ISPs are considered to be under several NAPs in the loose hierarchical structure. Moreover, there are some peer-links between some backbone ISPs. The lowest layer of the hierarchy consists of campuses, research organizations, and regional ISPs (we do not differentiate national, regional, and local ISPs here for simplicity). They are connected to backbone ISPs for cross-regional traffic routing. Moreover, they are also connected to local Internet Exchange (IX) for local traffic routing. The loose hierarchical structure can be visually represented as follows:

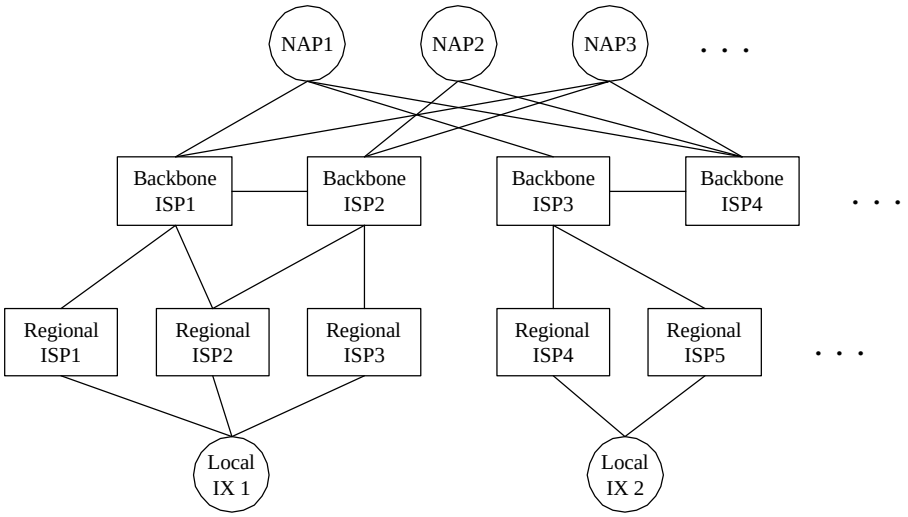


Figure 2.1: Loose Hierarchical Network Topology of Internet

For this loose hierarchical network topology, the optimal locations to place FLDSs are:

1. major NAPs;
2. local Internet Exchanges; and
3. backbone ISPs.

It is because all cross-domain traffic will route via these locations as follows:

1. Intra-region traffic: cross domain via Local Internet Exchange
2. Inter-region traffic: cross domain via NAP or Backbone ISP

According to www.ep.net, there are approximately 150 Internet Exchanges currently (50 in North America, 50 in Europe, 30 in Asia Pacific, and 20 in South America, Africa and Middle East). According to thelist.internet.com, there are almost 10,000 ISPs. Therefore, it is much more efficient to defend against DDoS attacks at the

Internet Exchanges and backbone ISPs than at every local ISP that may provide Internet connectivity to DDoS daemon machines.

Empirical Measurement of the Internet Topology

To support the above argument, I refer to an empirical study of network connectivity in the Asia-Pacific region performed by CAIDA (Cooperative Association for Internet Data Analysis) in 1999 [19]. This study used ICMP packets from nine geographically diverse monitors to trace AS paths to about 2000 destinations (mostly web servers) in the Asia-Pacific region. The result showed that four major backbone ISPs appeared in 52% of all traces. It demonstrates the hierarchical nature of the Internet topology as cross-domain traffic from local or regional ISPs are routed by backbone ISPs to the destinations.

To supplement the study performed by CAIDA, I performed *traceroute* from eighteen *traceroute* servers diverse at various locations in the globe (5 in Asia Pacific, 5 in North America, 5 in Europe, 2 in Latin America, and 1 in Middle East). The traces have fourteen destination sites diverse in different regions other than Asia Pacific (5 in North America, 5 in Europe, 2 in Latin America, 2 in Africa and Middle East). The destination sites are some of the most popular web sites in their respectively regions. Therefore 252 (18 x 14) routes are traced in this study, and for each route, 3 traces are made at different time and different days during the period 14.Sep.2000 – 23.Sep.2000 to cater for route instability.

From the 756 traces performed, it is found that AS path between a source-destination pair is very stable. 83% of the 252 source-destination pairs have the same AS path

across the 3 traces. This result is very similar to that by CAIDA (90% AS paths are stable over a day). Therefore, the further analysis is only based on one of the three sets of traces.

The analysis result shows that even though the average number of IP hops between a source-destination pair is 16.5, the average number of AS hops is only 5.4. If excluding the source and destination AS, a path has fewer than four transit ASs on average. Moreover, each path consists of 2.5 backbone ISP and 0.2 NAP/IX on average. This observation supports the argument that most inter-domain traffic are via backbone ISP and NAP/IX.

There are altogether 96 ASs appear in the traces. Excluding the source or destination ASs which do not appear in the path between any source-destination pair, there are only 70 ASs. The 70 transit ASs consist of 32 backbone ISPs, which appear in Appendix A (Backbone ISP List), and 8 NAPs or IXs. Therefore in this study, backbone ISP, NAP, and IX account for 57% of the transit ASs. Moreover, 20% of the traces consists of at least one NAP or IX. More importantly, in the 252 traces, only 3 traces do not consist of any backbone ISP, NAP and IX. It only accounts for 1% of the total sample size. All these figures support that most cross-domain Internet traffic can be monitored by installing FLDSs at backbone ISPs, Network Access Points, and Local Internet Exchanges.

The result also shows that the five major backbone ISPs[*] altogether appear in 65% of

[*] The five major backbone ISPs in this study are: Alternet (UUNET) - AS701, AT&T – AS7018, BBNPLANET – AS1, Sprintlink – AS1790, and Teleglobe – AS6453.

all the traces. This observation is very important. It shows that the proposed Internet-wide DDoS defense system can be effective even if only a small number of major backbone ISPs participate in this defense infrastructure by installing FLDSs.

Details of the empirical data and analysis can be found in Appendix C.

2.3. Integration of LDS with IX and Backbone ISP

In this section, I shall describe the network structure of a typical Internet Exchange (IX) and a backbone ISP. Then, I shall propose how a FLDS can fit into the existing network structures of Internet Exchange and backbone ISP. MLDS can integrate into other Internet node using the similar techniques.

2.3.1 IX and Backbone ISP Network Structure

For most cases, the core of an Internet Exchange is a high-speed layer-2 switching device. Every ISP or network connecting to this Exchange has a router locating at the exchange and connect its own network to the switch [20][21]. Visually, the network structure is as follows:

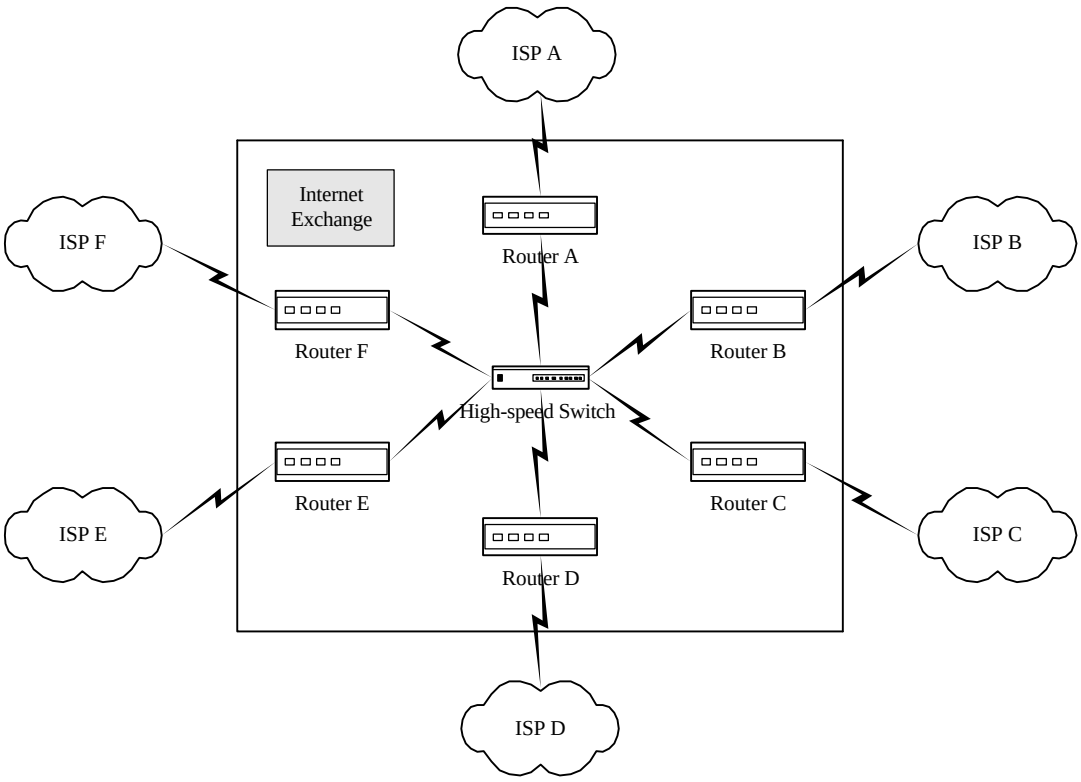


Figure 2.2: Network Diagram of Typical Internet Exchange

The network structure of backbone ISP is more complicated. It usually consists of hundreds of end nodes or points-of-presence (POP) interconnected by a number of gateway nodes or transit hubs. The following is the backbone map of AT&T which shows the network structure of a typical backbone ISP [22].

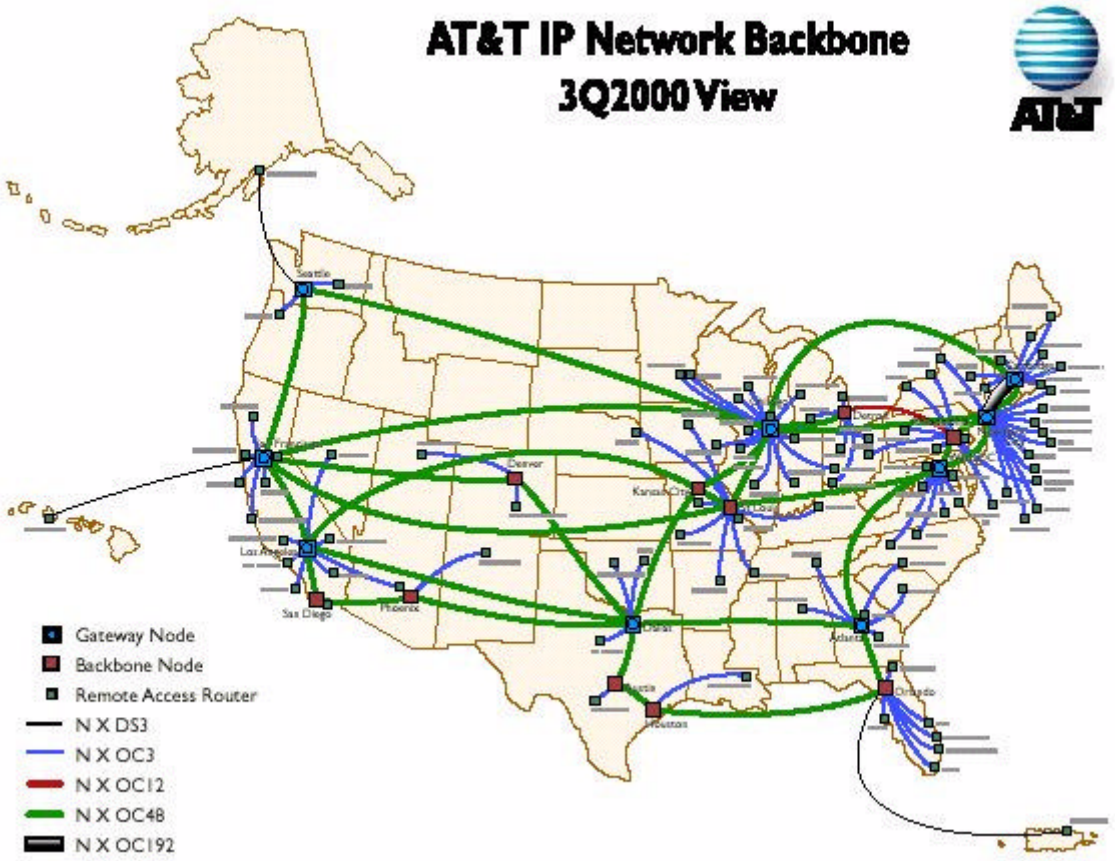


Figure 2.3: Network Diagram of AT&T Backbone

Since all inter-domain traffic should pass through gateway node, we only concern with the network structure of gateway node. At a gateway node, high-speed links from various POPs or other gateway nodes are interconnected by high-end routers and switches similar to that at Internet Exchange [23]. However, these routers and

switches are usually replicated and redundant both logically and physically to provide high capacity and robust routing infrastructure.

2.3.2. LDS in IX and Backbone ISP Network

There are at least two ways that a FLDS can fit into the network structure of backbone ISP and Internet Exchange. It can be built into the switch or router, or as a separate device connecting to switch or router of the network. The separate device approach has several benefits. It is easier to implement as it does not involve any changes to the existing switches or routers of the Internet Exchanges or backbone ISPs. More importantly, it will not degrade the network performance. Therefore this option is preferred even though it may be more difficult to carry out response actions, which usually involve modifying the routers' packet filtering rules, when DDoS attack is detected.

In some high-end layer-2 switches, there is a feature called Switched Port Analyzer (SPAN) that allows the switches to perform port mirroring for sending frames directly from a specified port to an external network analyzer [24]. Port mirroring is a feature that enables a switch to make an extra copy of the data moving through one port (the SPAN source) and transfer that data to another port (the SPAN destination). This setup allows an external probe to capture and analyze frames as if they were on the same segment as the SPAN source port. This capability is commonly referred to as roving RMON; that is, pointing an RMON agent at any traffic source on an as-needed basis.

Therefore, we can configure the switches at Internet Exchange or backbone ISP's gateway node to mirror all traffic to one or more SPAN destination ports, and connect

these SPAN destination ports to a FLDS. Such configuration allows the FLDS to listen to all traffic via the switches, and detect for any suspicious attacks.

Unfortunately, not all switches support SPAN. Even if a switch supports SPAN, it usually only supports SPAN for Ethernet source ports, but not ATM source ports. However, most backbone ISPs are using high speed ATM links as traffic carrier. Therefore, how to integrate LDS into backbone ISP's network structure is still an unsolved problem. Use of optical splitter to monitor traffic on fiber link may be a possible solution [25]. For the time being, I simply assume the ATM switches also support some feature similar to SPAN port and proceed with the design of our defense system.

Besides the interfaces connecting to SPAN ports, a FLDS can have some other interfaces connecting to the switch and routers. Via these network interfaces, the FLDS can command the switch and routers to install or remove packet filtering rules.

2.4. High-Level Design of Local Detection System

In this section, I shall describe the high level design of LDS. It includes the basic design, detection design, communication design, decision making design, and response design. Detail design of LDS can be found in Chapter 3 and 4 of this dissertation.

2.4.1. Basic Design

Traditional intrusion detection system uses a monolithic approach such that a single-thread software performs all the detection and analysis functions [26][27]. Common Intrusion Detection Framework (CIDF) is a standards effort funded by DARPA and its objective is to construct an infrastructure that allows intrusion detection and response systems to share information [28]. In contrast to the monolithic approach, CIDF defines a component-based approach such that an intrusion detection and response system consists of several components. Some components filter event data, some components analyze event data, some components are data repositories, and some components issue commands in response to attacks. These components communicate with each other using CIDF data formats.

I shall adopt the component-based approach of CIDF for designing the LDS, with some modifications. The advantage of adopting this approach is that it can be more effective in handling huge-volume traffic by dividing the workload among different components. This is very important in the huge-volume traffic environment in IX and backbone ISP. Moreover, the component-based approach is more flexible such that

MLDS and FLDS can use the same design, by just including different components.

2.4.2. Detection Design

There are two major techniques of intrusion detection: anomaly detection and misuse detection (pattern or signature recognition) [27]. Anomaly detection is based on determining patterns of “normal” behavior for networks, hosts, and users and then detecting behavior that is significantly different (anomalous). Misuse detection uses patterns of well-known attacks (signature) to match and identify known intrusion.

Since there are a number of well-known DoS and DDoS attacks, misuse detection can be used to detect these attacks effectively. However, as new DDoS tools are still evolving, only misuse detection is not enough to handle new types of attack. Anomaly detection is required even though it may be more difficult to design and less effective than misuse detection against well-known attacks.

Therefore, both misuse detection and anomaly detection should be used for detecting DDoS attacks. There should be a database of DDoS attack signatures. Moreover, the database should allow easy addition of new signatures. Currently, there is no standard way to store the intrusion signatures. Different intrusion detection systems use different ways. A new method, which makes use of XML, will be proposed in this dissertation.

Since the eventual result of an IP flooding DoS attack is a flood of IP packets to a victim, I propose to use the volume of traffic to any particular IP address to determine whether the traffic is normal. It is a kind of anomaly detection. It involves continuous

monitoring of traffic volume to various IP destinations by FLDSs.

2.4.3. Communication Design

Reasons to Communicate

There are two main reasons why detection systems at different locations need to communicate. The first reason is that they need to share information to confirm whether a DDoS attack is in progress. If only one FLDS observes abnormal traffic to a particular IP address, it may not be conclusive enough to confirm a DDoS attack is in progress. However, if detectors at different locations observe similar abnormalities, they can communicate with each other and conclude that there is a high probability that an attack is ongoing.

The second reason is that different detectors can co-operate to trace back DDoS attack in real-time and respond accordingly to stop the attack.

Information to Communicate

To satisfy the two reasons to communicate, the following information should be transmitted among LDSs:

1. information of suspicious attack;
2. detection and analysis decision;
3. response action; and
4. control information, e.g. heartbeat message.

Communication Language

There are some language standards in the market for communicating intrusion detection related information. Two of them are Common Intrusion Specification Language (CISL) [29] and Intrusion Detection Message Exchange Format (IDMEF) [30]. Other standards include OPSec by Checkpoint and ANSA by ISS.

The CISL is the principal result of the project Common Intrusion Detection Framework (CIDF). CIDF is funded by DARPA and its objective is to construct an infrastructure that allows intrusion detection and response systems to share information.

The IDMEF is an IETF project. Its purpose is similar to that of CISL. It is based on XML. Because of XML's extensibility and the wide availability of software tools for parsing and validating XML, IDMEF is the preferred language for this project.

Communication Security

In order to protect the communicating information from hacking by intruders, all communication among LDSs should be authenticated and encrypted. We shall adopt the Intrusion Alert Protocol (IAP) for communicating alert information among LDSs. IAP is the protocol for exchanging IDMEF messages proposed by IETF. It makes used of Transport Layer Security (TLS) to achieve secure communication among different parties.

Relationship among LDSs

As stated in Section 2.1, different LDSs should maintain a peer-to-peer relationship to communicate information with each other. However, there may be hundreds or more LDSs in the Internet. It is not efficient to manually configure a LDS to know all other LDSs in the Internet. Instead, I propose that each LDS should only be manually configured to know several immediate neighbor LDSs.

When a LDS initializes, it informs its immediate neighbors about its presence. The neighbors then respond and inform the newcomer their own immediate neighbors. Under normal circumstance, a LDS should only communicate with its immediate neighbors. Therefore if it detects a suspicious attack, it should inform all its immediate neighbors. A neighbor receiving this message should forward the message to its own immediate neighbors except the one sending the message to it. As this process repeats, a message from a LDS can reach every other LDS even it does not know the existence of the others. This communication approach allows each LDS to communicate with every other LDS but causes little administrative overhead and generates little network traffic.

In other words, all LDSs are connected by a non-partitioned graph, which may contain loops. To avoid message goes into infinite loop, each message should be identifiable by the source of the message and a timestamp when this message is created. Duplicated or old message received by a LDS should be discarded.

All immediate neighbors should communicate heartbeat message periodically. If a LDS has not received heartbeat messages from a particular neighbor for a certain

period of time, this neighbor is assumed dead. The LDS should then reconfigure its neighbor list to replace the dead LDS by the immediate neighbors of the dead LDS. It should also inform its neighbors about the new neighbor list. This mechanism ensures the global defense system is robust and can survive even some LDSs have crashed. Later on, if heartbeat is received from the failed neighbor LDS again, it is added back to the neighbor LDS list, while the neighbor's neighbors should be removed.

2.4.4. Decision Making Design

Rules should be set up to detect for attack signatures and traffic abnormalities as discussed earlier. Based on these rules and the traffic patterns observed locally, a FLDS can decide whether there is a suspicious attack and what is the confidence level of this suspicion.

Suspicious attack alerts are transmitted to LDSs at remote locations. A LDS then consolidates the decisions from different LDSs (may include itself if it is a FLDS), and determines the confidence level of the suspicion based on all available information. If the confidence level is higher than a certain threshold, it is considered that an attack is ongoing and response actions should be taken.

2.4.5. Response Design

General Mechanism

When a LDS confirms that there is a DDoS attack, it should find out the inbound interface of the suspicious packets and request the routers or switches at the corresponding Internet Exchange or backbone ISP to filter the suspicious packets. The

LDS should also request the LDS at the upstream network node to perform similar action. This mechanism re-iterates until it traces back to the sources of attack.

Implementation Approach

LDS can ask local routers to install access control list entry for filtering suspicious packets based on destination IP address and port number. However, filtering rules may degrade the performance of the routers.

To solve this problem, it is proposed to have every access control entry be valid for several minutes only. During this period, the upstream LDS is informed about the attack and should install similar access control entries to its local routers. Therefore the attack traffic cannot arrive at the downstream routers anymore and the control entries are no longer required. This process repeats with the next upstream LDS and so on. Finally, only the ISP that is directly connected to an attack source should maintain the access control entry until the attack source is taken down or fixed.

Chapter 3
ARCHITECTURE OF LOCAL DETECTION SYSTEM

3.1. System Components of LDS

As stated in Chapter 2, the functions of a local detection system include the followings:

1. To detect for suspicious DDoS attack by signature detection.
2. To detect for suspicious DDoS attack by anomaly detection.
3. Communicate information with other local detection systems.
4. Determine whether there is a DDoS attack based on both local and remote information.
5. Respond to a confirmed attack.

The first and the fifth functions are performed by any typical intrusion detection systems (IDS). Typical IDS also performs the fourth function except it only uses local information for making decision. Therefore, our local detection system should minimally include the components of a typical IDS. Following the CIDF, we have at least four components:

1. E-box, or event generator, which collects and filters event data, and pushes out report. In our context, it performs signature detection for DDoS attacks.
2. A-box, or analyzer, which receives reports and performs analysis. In our context, it analyzes local data to find out suspicious attacks. It also analyzes both local and remote data to determine if a DDoS attack is in progress.
3. D-box, or database component, which is a repository for any kind of data, both raw and processed.

4. R-box, or response component, which takes the input of E, A, and D-boxes and issues commands in response to attacks.

In order to handle high volume traffic, I add a component called S-box, or traffic distributor. It distributes traffic destined for different destination IP addresses to different E-boxes. By distributing the workloads among multiple boxes, our system can be more scalable.

In order to perform anomaly detection, I add one more component to the system. It is called P-box, or packet capturer. P-box is responsible for capturing packet header of every packet and write out traffic anomalies event records to A-box and D-box periodically. In our context, the anomalies refer to abnormal IP, TCP, UDP or ICMP traffic rate to a particular IP address. Strictly speaking, P-box is a type of E-box in CIDF. However, in order to distinguish it from the signature detection engine, I call this logical component as P-box.

For communication among different local detection systems, I add one more component called M-box to the system. It is responsible to send and receive control packets to and from other local detection systems periodically. It also sends out information to other systems when a suspicious attack is detected or response action involving other LDSs is needed.

I propose that a LDS should be centrally managed by a GUI console called C-box. Administrator can change the traffic distribution rule of S-box, update the signature database of E-box, monitor the results found by A-box, etc., through this console.

3.2. FLDS & MLDS

In Section 2.2, I have categorized LDS into FLDS and MLDS. FLDS is located at the strategic locations in the Internet. It is composed of all the components identified in the previous section. It performs both detection of and response action against DDoS attack. MLDS is located at non-strategic locations. It only performs response actions to block DDoS attack, but not the attack detection. Therefore, it does not have E-box. Moreover, its P-box is only responsible for detecting suspicious attack interface (for attack response), but not suspicious attack (for attack detection). The other components of MLDS are identical to that of FLDS.

3.3. Network Design of LDS

Most Internet Exchanges and backbone ISPs need to handle Gigabits of data traffic per second. Moreover, the traffic volume is expected to increase at high rate in future. It is a great challenge for any systems attempt to analyze traffic data at these strategic locations.

Since it may not be feasible to use a single device to detect and analyze all traffic, I propose to use a multiple-device approach to design LDS. This approach makes a system more scalable.

Consider a switch in an Internet Exchange, we can use the switch's SPAN ports to mirror the ingress traffic of each interface to the distributor or S-box of our LDS network. The S-box is connected to one or more LAN segments depending on the volume of traffic need to handle. On each LAN segment, there is an E-box and a P-box. E-box and P-box can be integrated into one physical device or run on two separate physical devices. E-box and P-box listen to the traffic on this LAN segment promiscuously. I call this LAN segment external LDS segment.

All the E-boxes and P-boxes are connected with the A-box, D-box, R-box, M-box, and C-box on another LAN segment. I call this LAN segment internal LDS segment. All the boxes communicate with each other through this internal LDS segment. The R-box has another interface connecting to the switch for dynamically installing filters in response to confirmed DDoS attack. The M-box has another interface to connect to the Internet for communication with other LDSs. The M-box will not forward any IP

packet from the Internet to the LDS network, such that no one in the Internet can access the other boxes. It makes sure our LDS network is a secure network. The A-box, D-box, R-box, C-box, are just logical components. Physically, they can reside on the same machine.

The network structure of a LDS that I propose therefore looks like the following:

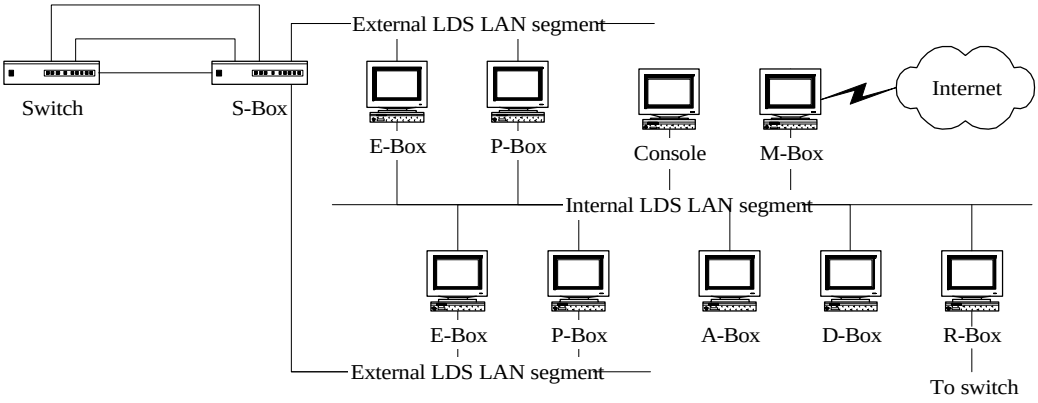


Figure 3.1: Network Diagram of LDS

3.4. Communication Language and Protocol

As proposed in Section 2.4.3, the language used for communicating information among different components in a LDS and among different LDSs are based on IDMEF (Intrusion Detection Message Exchange Format) [30]. IDMEF messages are transmitted over Intrusion Alert Protocol (IAP) [31], an application-level protocol for exchanging intrusion alert data. Both IDMEF and IAP are under development by Intrusion Detection Working Group (IDWG) of IETF. In this section, I shall briefly describe these two draft standards.

3.4.1. Intrusion Detection Message Exchange Format (IDMEF)

IDMEF Alert

IDMEF is based on XML. In each IDMEF document, there is always a root element of <IDMEF-Message>. Under this root element, there is a list of <Alert> elements which specify the alert information need to be communicated between the sender and the receiver.

The following is an IDMEF message about a DoS attack, It is extracted from the IDMEF Internet Draft [30]:

```
<IDMEF-Message version="0.1">
  <Alert alertid="12345.123456789" impact="successful-dos">
    <Time offset="-0500">
      <ntpstamp>0x12345.0x67890</ntpstamp>
      <date>2000/03/09</date>
      <time>10:01:25.93464</time>
```

```
    </Time>
    <Analyzer ident="12345">
      <Node category="dns">
        <location>Headquarters DMZ Network</location>
        <name>analyzer01.bigcompany.com</name>
      </Node>
    </Analyzer>
    <Classification origin="bugtraqid">
      <name>124</name>
      <url>http://www.securityfocus.com</url>
    </Classification>
    <Source>
      <Node  ident="12345.s7beae779" category="dns">
        <name>badguy.hacker.net</name>
        <Address category="ipv4-addr">
          <address>123.234.231.121</address>
          <netmask>255.255.255.255</netmask>
        </Address>
      </Node>
    </Source>
    <Target>
      <Node ident="12345.tde796f70" category="dns">
        <Address category="ipv4-addr-hex">
          <address>de796f70</address>
        </Address>
      </Node>
    </Target>
  </Alert>
</IDMEF-Message>
```

In the <Alert> element, the <alertid> attribute is the unique serial number of alert generated by a given analyzer. The <impact> attribute is the impact of this event. The <Time> element specifies the time zone, date and time when the alert is generated. The <Analyzer> element specifies the entity sending this alert. The <Classification> element specifies the name of the event that caused this alert to be generated. The

<origin> attribute specifies the origin of the name of the classification, which should be “bugtraqid” for the Bugtraq ID naming scheme (www.securityfocus.com), or “cve” for the Common Vulnerability Enumeration naming scheme (www.cve.mitre.org), or “vendor-specific” for a vendor-specific name. The <Source> and <Target> elements contain information about the source and target of the event.

IDMEF Heartbeat

IDMEF has also defined a <Heartbeat> element which is used to provide status information about a component. In its simplest form, the heartbeat simply indicates that the component is still up and running. The following is an example of IDMEF heartbeat message:

```
<IDMEF-Message version="0.1">
  <Heartbeat heartbeatid="123456789">
    <Time offset="+0000">
      <ntpstamp>0x12345.0x67890</ntpstamp>
      <date>2000/03/09</date>
      <time>14:07:58</time>
    </Time>
    <Analyzer ident="12345">
      <Node category="dns">
        <name>Ebox1.polyu.edu.hk</name>
      </Node>
    </Analyzer>
  </Heartbeat>
</IDMEF-Message>
```

3.4.2. Intrusion Alert Protocol (IAP)

The Intrusion Alert Protocol (IAP) is an application-level protocol for exchanging

alert data. The protocol is designed to provide the necessary transport and security properties to allow sensitive alert data to be sent across IP networks. It uses the TCP as its underlying layer mechanism.

IAP can be divided into two major phases. The first phase is the setup phase. In this phase, TCP connection is set up, security (e.g. TLS handshaking) and channel parameters (e.g. role of the peers in the connection) are agreed upon by the peers.

In the second phase, encoded IDMEF alerts are sent from the sender to the receiver over the TLS record layer. Termination can be initiated by either peer by sending a TLS close-notify alert.

IAP uses a subset of the HTTP/1.1 syntax to send IDMEF alerts. Its request-response protocol is modeled on HTTP.

3.5. Interaction among LDS Components

The following diagram summarizes the interaction among different components of a LDS when a DDoS attack is ongoing and attack traffic is passing through the network node monitored by this LDS:

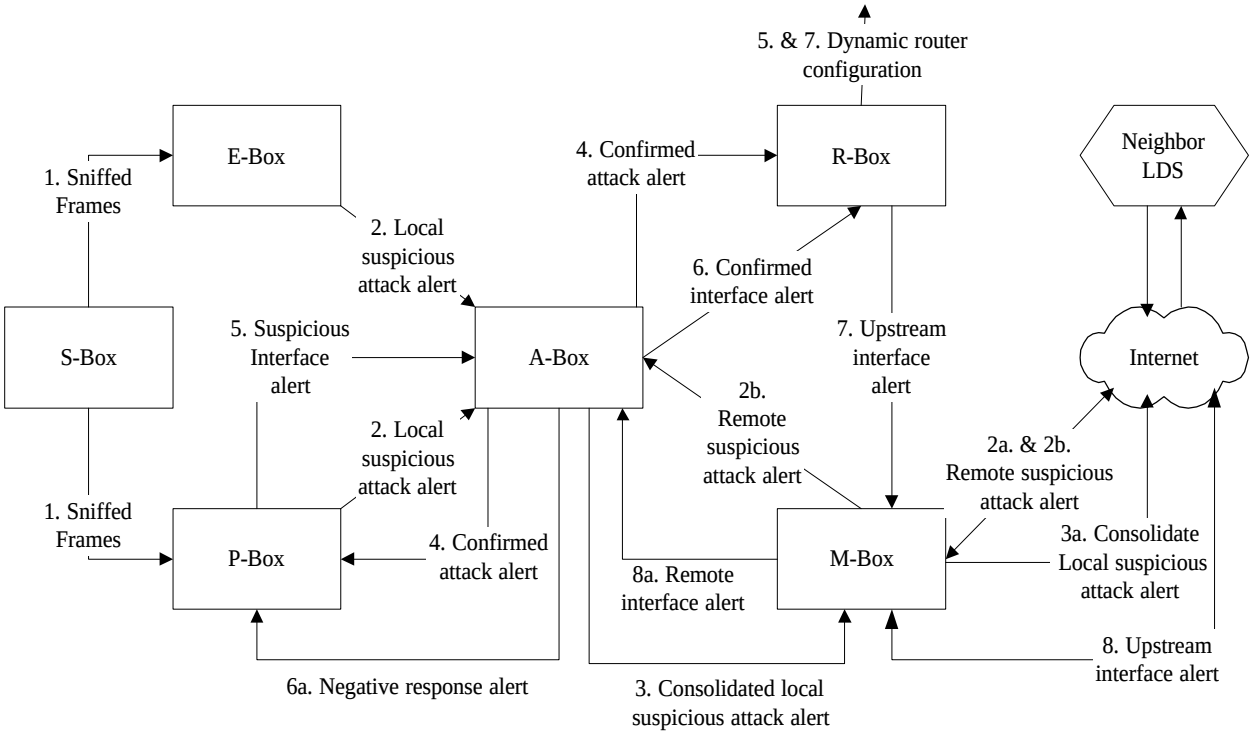


Figure 3.2: Data Flow Diagram of LDS

The data flow sequence is briefly described as follows:

1. S-box sniffs frames passing through the switch of the network node (Internet Exchange or backbone ISP node) and forwards them to E-box and P-box without any modification to the frame (including header).
2. E-box finds suspicious DDoS control packets and P-box finds suspicious traffic volume anomalies. Both generate local suspicious attack alerts and send to A-

box.

2a. On the other hand, suspicious attack could also be detected by a remote LDS and it sends a remote suspicious attack alert to M-box.

2b. M-box forwards the remote suspicious attack alert to A-box and neighbor LDSs, except the source of the alert.

3. A-box consolidates the local suspicious attack alerts and generates consolidated local suspicious attack alert. It then sends the alert to M-box.

3a. M-box forwards the consolidated local suspicious attack alert to all neighbor LDSs.

4. A-box consolidates consolidated local suspicious attack alerts and remote suspicious attack alerts to decide whether there is really an attack. If so, and if there is local suspicious attack alert against the victim in current or last sample period, it generates confirmed attack alert and sends it to P-box and R-box.

5. On receiving the confirmed attack alert from A-box, R-box installs traffic rate limit filters on the switch or routers to limit traffic destined to the victim from all inbound interfaces. P-box after receiving the confirmed attack alert, begins to monitor for suspicious interface. If it finds traffic volume anomalies to the victim from particular interfaces, it sends suspicious interface alerts to A-box.

6. A-box consolidates the suspicious interface alerts. If there are enough evidence, it generates a confirmed interface alert and sends it to R-box. Otherwise, a negative response alert is sent to P-boxes (6a) such that they will increase the maximum traffic rate counters for the victim at the end of this sample period (to raise the anomaly alert trigger threshold).

7. On receiving the confirmed interface alert from A-box, R-box installs traffic rate limit filter only on switch interfaces or routers corresponding to the confirmed

attack interfaces, and remove filters from the others. R-box also generates upstream interface alert and sends it to M-box.

8. M-box forwards the upstream interface alert to the specified upstream LDS.

8a. If upstream interface alert is received, M-box forwards it to A-box. A-box then sends confirmed attack alert to P-box to start interface monitoring and R-box to start response action, just like step 4.

For simplicity, the above discussion leaves out D-box. Actually, all alerts should be sent to D-box for recording. Heartbeat messages are also ignored in the diagram. In fact, every component should send heartbeats to A-box and D-box, such that A-box can know if different boxes are functioning normally and D-box can record down the status of the boxes. A-box also sends Heartbeat message to M-box, which will then be sent to neighbor LDSs. The next Chapter will explain the design of different components in more detail.

Chapter 4
SYSTEM COMPONENTS OF LOCAL DETECTION SYSTEM

4.1. S-Box Design

A LDS has multiple detection devices (E-boxes and P-boxes) such that the packet and event capturing functions can be distributed over multiple devices and the system can be more scalable. I propose to use S-box, or traffic distributor, to sniff the ingress frames (link layer) of the target switch of Internet Exchange or backbone ISP. It then distributes the traffic to the detection devices evenly and at the same time facilitates the merging of the information extracted by different detection devices. In this section, I shall discuss the detail design of the proposed S-box.

4.1.1. Frames Sniffing

As suggested in Section 2.3.2, we can achieve frames sniffing by connecting the S-box to the SPAN ports of the target switch if the switch supports this feature. Alternatively, the S-box needs to sniff the fiber links connected to the switch directly using optical splitters.

S-box needs to sniff every ingress frame to the switch. It needs to send the frame to the external LDS LAN segment without any modification. In other words, the identical link layer frame should be forward to the external LDS segment. It is because E-boxes and P-boxes need to analyses the IP header to detect for suspicious attacks and the link layer header to find out which interface the frame entered the switch from if it is an attack packet.

4.1.2. Frames Distribution

In order to achieve the objectives of evenly distributing the traffic and facilitating information merging, I propose traffic distribution rule for S-box to be based on a packet's destination IP address. For example, if two detection devices are adequate to handle the traffic volume, the distribution rule is as follows:

1. Perform bit-wise OR for the IP destination address and the netmask 255.255.255.254.
2. Detection device A handles this packet if the bit-wise OR result is 255.255.255.255.
3. Detection device B handles this packet if the bit-wise OR result is 255.255.255.254.

If four detection devices are required, the netmask to be used is 255.255.255.252. A packet is handled by one of the four devices depending on the bit-wise OR result as follows:

1. Detection device A - 255.255.255.255;
2. Detection device B - 255.255.255.254;
3. Detection device C - 255.255.255.253; and
4. Detection device D - 255.255.255.252.

More detection devices can be added as the traffic volume increases and only the netmask needs to be modified.

The proposed distribution rule has several advantages. First, it allows easy merging of information captured by different detection devices. Since traffic anomaly for any

particular destination address is one major factor to monitor, we need to merge information collected by different detection devices based on destination address. This distribution rule saves the information merging cost as all traffic to a particular destination address is always processed by the same detection device.

Second, by using the rightmost bits of the destination IP address to distribute traffic, we can distribute the traffic to different detection devices evenly in normal case. When there is a DDoS attack in progress, the traffic volume to the victim site will increase sharply. Since all traffic to any particular address will be processed by the same detection device, the sudden increase in traffic will not affect other detection devices. For the detection device which handles the traffic to the victim, it may not be able to process all packets if the traffic volume increases to an extremely high level. However, under such case, the detection device does not need to process all packets before concluding there is a suspicious attack. The detected traffic volume is sufficient to trigger an alert.

4.1.3. Communication with Other Components

Besides distributing traffic among different detection devices, a S-box should also send heartbeat messages to the A-box and D-box periodically to show that it is alive. IDMEF Heartbeat message is used for this purpose. The heartbeat interval is proposed to be 30 seconds.

4.2. E-box Design

In this section, I shall propose the design of E-box, or event generator. Moreover, I shall propose a new approach to specify and store intrusion signatures based on XML. I shall also explain how DDoS attack signatures can be prepared using this new approach.

E-Box detects for suspicious DDoS events based on signature recognition. It listens to every packet on the external LDS segment connected to the S-box in promiscuous mode, compares the packet to all signature patterns stored in the signature database, and records it down when signature match is found. At the end of each sample period, alerts are sent to the A-box to communicate the detected information.

4.2.1. Signature Database

The signatures can be prepared by studying the existing DDoS attack tools' control messages, which are transmitted between DDoS attacker/master, and master/daemon. We can also study the attack packets originated from the daemons and prepare the signatures. However, there is usually no well-defined signature in the attack packets because they actually are dummy packets generated by daemons randomly.

Let us use one of the first generation DDoS attack tools, trinoo, as an example. According to CERT Incident Note IN-99-07 [7], an attacker communicates with a master using destination TCP port 27665, while a master communicates with a daemon using destination UDP port 27444. Moreover, all communications with the

daemon require the UDP packets to contain the string “l44”. All these facts about trinoo can be used to prepare the signatures for detection.

In the market, there are a number of research and commercial intrusion detection systems (IDSs). These systems use different data formats for their signature databases because of different detection techniques being used and because of proprietary nature of commercial products. For example, EMERALD (Event Monitoring Enabling Responses to Anomalous Live Disturbances), a research IDS, is based on the P-BEST expert system for writing decision rules to detect suspicious activities [32]; NFR (Network Flight Recorder), an IDS with both public domain and commercial version, has its own language, called N-code, for writing pattern matching filters [33].

Since E-box only needs to detect for signatures related to DDoS events, its signature database should be simpler than that of the other IDSs. And due to the huge volume of traffic needs to be processed, the design of the signature database should emphasize on simple, easy and fast processing.

4.2.2. XML Signature Database

I propose to use XML as the data format of the signature database as it can satisfy the above needs and provides the extensibility to incorporate new signatures easily.

For speedy processing, E-box should analyze on packet-by-packet basis, rather than on flow-of-packets basis. The following packet information are interested to us:

1. protocol type;
2. source IP address;

3. source port;
4. destination IP address;
5. destination port;
6. packet data with specific pattern, e.g. string” l44” in trinoo control packet; and
7. other information specific to the protocol type, e.g. SYN flag of TCP segment.

The Document Type Definition (DTD) for the signature database should therefore look like the following:

```
<!ENTITY % ext.attvals.adtype         "">
<!ENTITY % attvals.adtype             "
    ( unknown | boolean | byte | character | date | integer |
     ntpstamp | real | string | time
    %ext.attvals.adtype; )
">

<!DOCTYPE SignatureDB [
  <!ELEMENT SignatureDB (Signature*)>
  <!ELEMENT Signature (Name, Description, AlertTool, Protocol,
    SourceAddr?, SourcePort?, DestAddr?, DestPort?,
    DataPattern?, AdditionalData*)>

  <!ATTLIST Signature
    id                  CDATA                          #REQUIRED
    confidence          ( 0 | 1 | 2 | … | 100 )        “0”
    occurrence          ( 0 | 1 | 2 | … | 100 )        “0”
    attackeraddr        ( SOURCE | DEST | UNKNOWN )    “UNKNOWN”
    masteraddr          ( SOURCE | DEST | UNKNOWN )    “UNKNOWN”
    daemonaddr          ( SOURCE | DEST | UNKNOWN )    “UNKNOWN”
    victimaddr          ( SOURCE | DEST | UNKNOWN )    “UNKNOWN”
  >

  <!ELEMENT Name (#PCDATA)>
```

```
  <!ELEMENT Description (#PCDATA)>
  <!ELEMENT AttackTool (#PCDATA)>
  <!ELEMENT Protocol (#PCDATA)>
  <!ELEMENT SourceAddr (#PCDATA)>
  <!ELEMENT SourcePort (#PCDATA)>
  <!ELEMENT DestAddr (#PCDATA)>
  <!ELEMENT DestPort (#PCDATA)>
  <!ELEMENT DataPattern (#PCDATA)>
  <!ELEMENT AdditionalData (#PCDATA)>

  <!ATTLIST AdditionalData
    type                %attvals.adtype;                'unknown'
    meaning             CDATA                           #IMPLIED
  >

] >
```

It specifies the format of a XML signature database document. A document contains one or more signatures. Each signature has a name, a description, and the name of the associated attack tool.

A signature specifies the criteria of matching packets. They include the transport protocol of the payload of the matching IP packet, the IP address and the port number of the source and destination in the IP header, and the data pattern in the matching IP packet. Other matching criteria can be included in a signature using the <AdditionalData> element. The definition of the <AdditionalData> is borrowed from that of IDMEF.

The attributes of a signature specify additional information about the signature. Each signature has an ID attribute for identification. Other attributes specifies the address of the attacker, master, daemon, and victim (SOURCE means source address of the

matched packet, DEST means the destination address of the matched packet). The confidence attribute specifies the confidence level (0 - 100) that the signature really related to a DDoS event if the number of occurrence of the matched packet equals to or is more than the specified occurrence. Normally, only level 1-99 are used. Level 0 is considered as unknown confidence level and 100 is reserved for special use.

Back to the example of signatures for trinoo, the following is a XML signature database with signatures of trinoo. Three signatures are included. The first signature specifies if 10 or more TCP segments with the same source and destination IP address, and destination port equals to 27665 are detected within a sample period, we have a confidence of 10 that the source is a trinoo attacker, and the destination is a trinoo master. The second signature specifies if 10 or more UDP datagrams with the same source and destination IP address, and destination port equals to 27444 are detected within a sample period, we have a confidence of 10 that the source is a trinoo master, and the destination is a trinoo daemon. The third signature specifies if 5 or more UDP datagrams with the same source and destination IP address, and have a string of “l44” in the payload, we have a confidence of 30 that the source is a trinoo master, and the destination is a trinoo daemon.

```
<?xml version=”1.0” standalone=”no”>
<!DOCTYPE signatureDB SYSTEM “signatureDB.DTD”>
<SignatureDB>
  <Signature id=”1” confidence=”10” occurrence=”10”
    attackeraddr=”SOURCE” masteraddr=”DEST”
    deamonaddr=”UNKNOWN” victimaddr=”UNKNOWN”>
    <Name>DDoS - Trinoo Master Port</Name>
    <Description>Trinoo master’s port for communication
     with intruder.</Description>
```

```
    <AttackTool>Trinoo</AttackTool>
    <Protocol>TCP</Protocol>
    <DestPort>27665</DestPort>
  </Signature>

  <Signature id=”2” confidence=”10” occurrence=”10”
    attackeraddr=”UNKNOWN” masteraddr=”SOURCE”
    daemonaddr=”DEST” victimaddr=”UNKNOWN”>
    <Name>DDoS - Trinoo Daemon Port</Name>
    <Description>Trinoo daemon’s port for communication
     with master.</Description>
    <AttackTool>Trinoo</AttackTool>
    <Protocol>UDP</Protocol>
    <DestPort>27444</DestPort>
  </Signature>

  <Signature id=”3” confidence=”30” occurrence=”5”
    attackeraddr=”UNKNOWN” masteraddr=”SOURCE”
    daemonaddr=”DEST” victimaddr=”UNKNOWN”>
    <Name>DDoS - Trinoo Daemon Pattern l44</Name>
    <Description>All communications with Trinoo daemon
     have string “l44”.</Description>
    <AttackTool>Trinoo</AttackTool>
    <Protocol>UDP</Protocol>
    <DestPort>27444</DestPort>
    <DataPattern>l44</DataPattern>
  </Signature>
</SignatureDB>
```

4.2.3. Detection of Signature Match

For every sample period, if signature match with enough occurrences (as specified in the signature) is found, E-box should send an alert message to the A-box for analysis and the D-box for recording. However, the same signature with same source and destination addresses should only trigger one alert message within each sample period.

For example, if 10 packets (>= required occurrences) which match a particular signature are found within a sample period, one alert message is generated, if 20 such packets are found, also only one alert is generated. The alert includes information such as signature's name, corresponding attack tool, IP addresses of the concerning parties, and the alert confidence, etc. (see Section 4.2.4 for the detail format of the alert). The proposed length of the sample period is 1 minute. It is called the normal sample period.

Most signature matched packets are control message packets transmitting among attackers, masters, and daemons, rather than attack packets transmitting from daemons to victim. Therefore, this kind of alerts is more importantly be used for collecting information about the actual attacker for later law enforcement, rather than for real-time detection and blocking of DDoS attacks. For example, if messages from the attacker to some masters are captured, we may able to tell the IP address of the attacker. Alternatively, based on messages between masters and daemons, we can locate the addresses of the masters. Files on these machines usually have a list of other masters and daemons. Some of these machines may have detail system logs, such that we can trace the IP address of the actual attacker using these logs.

4.2.4. Communication with Other Components

Signature Match Alert

Signature match alerts are sent from E-box to A-box and D-box in form of IDMEF message. For example, if an E-box detects a trinoo daemon port signature, it will send out the following IDMEF message.

```
<IDMEF-Message version="0.1">
  <Alert alertid="12345.123456789" impact="successful-ddos">
    <Time offset="-0500">
      <ntpstamp>0x12345.0x67890</ntpstamp>
      <date>2000/03/09</date>
      <time>10:01:25.93464</time>
    </Time>
    <Analyzer ident="12345">
      <Node category="dns">
        <location>PolyU LDS</location>
        <name>Ebox1.polyu.edu.hk</name>
      </Node>
    </Analyzer>
    <Classification origin="vendor-specific">
      <name>DDoS - Trinoo Daemon Port</name>
      <url>http://www.comp.polyu.edu.hk</url>
    </Classification>
    <Source>
      <Node  ident="12345.s7beae779">
        <Address category="ipv4-addr">
          <address>123.234.231.121</address>
          <netmask>255.255.255.255</netmask>
        </Address>
      </Node>
    </Source>
    <Target>
      <Node ident="12345.tde796f70">
        <Address category="ipv4-addr-hex">
          <address>de796f70</address>
        </Address>
      </Node>
    </Target>
    <ToolAlert>
      <name>Trinoo</name>
    </ToolAlert>
    <AdditionalData type=”string” meaning=”SourceRole”>
      Master
```

```
    </AdditionalData>
    <AdditionalData type=″string″ meaning=″TargetRole″>
      Daemon
    </AdditionalData>
    <AdditionalData type=″integer″ meaning=″Confidence″>
      10
    </AdditionalData>
    <AdditionalData type=″bytes″ meaning=″IP-Packet″>
      189A03275EA3 ...
    </AdditionalData>
  </Alert>
</IDMEF-Message>
```

Different elements in the alert are described below:

- In the <Alert> element, the <impact> attribute equals to “successful-ddos”. This value is not defined in the current version of IDMEF, but should be added for communicating DDoS alerts.
- The <Analyzer> element specifies the entity sending this alert. In our case, it is the E-box.
- The <Classfication> element specifies the name of the event that caused this alert to be generated. In our case, it is the name of the signature that a match is found. The <origin> attribute should equal “vendor-specific” since the classification name is not registered in the Bugtraq ID naming scheme (“bugtraqid”) or the Common Vulnerability Enumeration naming scheme (“cve”).
- The <ToolAlert> element specifies the attack tool, which is extracted from the <AttackTool> element in the signature.
- <AdditionalData> elements are used to specify the roles of the source and destination in the attack. Since a DDoS attack usually involves four parties: attacker, master, daemon, and victim, rather than two parties, depending on the

signature, we need to use <Source> element to represent: 1) Attacker; 2) Master; or 3) Daemon, and <Target> element to represent: 1) Master; 2) Daemon; or 3) Victim. In this case, the source is a master while the target is a daemon.

- Another <AdditionalData> element is used to specify the confidence level that the E-box has in the alert. The value is simply copied from the corresponding signature.
- Another <AdditionalData> element, with meaning equals to “IP-Packet”, attaches the complete IP packet into the alert for later analysis. It is in form of a series of bytes. Type “bytes” for AdditionalData is not defined in the current version of IDMEF and therefore should be added as user definition.

Heartbeat Message

Besides IDMEF Alert messages, E-box should also send IDMEF Heartbeat messages to A-box and D-box periodically such that they know the E-box is alive. The heartbeat time interval is 30 seconds.

4.3. P-box Design

In this section, I shall propose the design of P-box, or packet-capturer. The objective of P-box is to detect for any traffic volume anomalies. I propose that P-box should scan each IP packet's header and record the extracted information in a large hash table. When abnormality is observed from the table, it informs the A-box and D-box. When A-box confirmed that there is an attack against a particular victim, P-box should monitor the packets (including link layer header) destined for this victim and detect if there is any suspicious inbound interfaces where the attack packets come from.

4.3.1. Detection of Traffic Volume Anomalies

A P-box can maintain a very large hash table of, say, 1 million slots. Each slot is related to one or more IP addresses based on a hash function. This hash table can limit the memory requirement of P-box, compared with the case that one slot is used per IP address. For each slot, the following information are kept:

1. number of ICMP packets to the IP addresses correspond to this slot in the current period, and the maximum number over the previous periods;
2. number of UDP packets to the IP addresses correspond to this slot in the current period, and the maximum number over the previous periods;
3. number of TCP packets to the IP addresses correspond to this slot in the current period, and the maximum number over the previous periods;
4. number of TCP SYN packets to the IP addresses correspond to this slot in the current period, and the maximum number over the previous periods; and

5. number of TCP RST packets from the IP addresses correspond to this slot in the current period, and the maximum number over the previous periods.

For each IP packet listened by P-box, the destination IP address and the protocol type are extracted from the packet. The P-box then performs the hash function on the IP address to find out the corresponding hash slot. Based on the protocol type, the corresponding counter is incremented. If the packet contains a TCP segment, checks for the SYN and RST flag. If it is a SYN segment, also increments the SYN counter. If it is a RST segment, update the RST counter of the slot corresponds to the destination IP address.

Before further proceed with the design of P-box, I shall first define a number of parameters that are used by P-box. They are as follows:

Parameter	**Description**	**Suggested Value**
Normal sample period	Current counters will be compared with maximum counters at the end of each normal sample period. Suspicious attack alerts will be generated if anomalies are found.	1 min (same as that of E-box)
Response sample period	Shorter sample period for collecting traffic anomalies per interface when there is a confirmed attack. At the end of a response sample period, suspicious interface alert will be generated if anomalies are found.	5 sec
Alert ratio	If current counter >= maximum counter x alert ratio, it is considered that traffic volume is abnormally high and suspicious	depends on result of Internet traffic analysis

	attack alert is generated, given the current >= minimum alert threshold.	
Minimum alert threshold	Alert will be generated only if current counter >= minimum alert threshold. It avoids false alarm caused by sudden increase of traffic but of negligible volume.	depends on result of Internet traffic analysis
Interface alert ratio threshold	Suspicious interface alert will be generated if the victim-skew ratio of an interface is greater than this threshold.	2

Table 4.1. Parameters used by P-box

After every normal sample period, P-box should compare each counter with the corresponding previous maximum, and performs the followings:

1. if (current counter < maximum counter)
   - 1.1 maximum counter = maximum counter - 1;
   - 1.2 reset current counter;

   else
2. if (maximum counter <= current counter < maximum counter x alert ratio

   OR current counter < minimum alert threshold)
   - 2.1 maximum counter = current counter;
   - 2.2 reset current counter;

   else
3. if (current counter >= maximum counter x alert ratio

   AND current counter >= minimum alert threshold)
   - 3.1 identify the address corresponding to this hash slot with the highest traffic rate in the coming short period (e.g. 5 sec) as the suspicious victim and

send an alert to A-box and D-box, with confidence = f (current counter, maximum counter),

e.g. Min((current counter / maximum counter / alert ratio x 10), 99) ;

3.2 reset current counter;

3.3 if a negative response interface alert is received from A-box within the last response sample period, i.e. no interface traffic volume anomalies are detected (see Section 4.4 for details),

set maximum counter = maximum counter x 7/8 + current counter x 1/8.

The above decision rule detects for sudden increase of traffic destined for a particular address, which is a good indicator of an actual DDoS attack against the address.

TCP RST packets are included in the analysis to cater for the case that daemons attack a victim using TCP packets, and spoof source addresses to valid neighbor IP addresses, or not use spoofed addresses at all. Under such cases, the victim will send a lot of TCP RST back to the daemons on receiving unexpected attack packets from them.

The normal sample period should be long enough to avoid false alarms caused by burst traffic. It should also be long enough such that the computing resources requirement is reasonable. However, it cannot be too long; otherwise, the system will not be responsive enough. Here, I propose to use the same normal sample period used by E-box, i.e. 1 minute.

The alert ratio and minimum alert threshold should be set to balance the false positive

(false alarm) and false negative (undetected attack) ratio. Data mining on traffic pattern in actual or simulated attacks can provide this information.

The above anomaly detection rule has the following strengths:

1. It is simple and feasible for processing huge traffic volume.
2. It is adaptive to different traffic volumes of different destination IP addresses, as the peak traffic rates to different IP addresses are recorded periodically and used to calculate the alert threshold.
3. The use of the past peak traffic rate to calculate the alert threshold aligns with our objective to minimize service interruption of victim, since it is an indicator of the maximum capacity of the concerning site.
4. If no interface traffic volume anomalies are found for the switch, the maximum counter will be revised upward. It caters for web sites that are not high-volume sites on a continuing basis, but have to deal with unprecedented load levels for certain periods of time (e.g. NASA site during the Mars landing).

Since this dissertation emphasizes on the whole infrastructure to defend against DDoS attacks, only one anomaly detection rule is suggested. If future studies identify other more efficient rules, they can be incorporated into this infrastructure. The component-based approach of this infrastructure allows incorporation of new detection algorithms without any changes to the components other than the P-box.

4.3.2. Detection of Inbound Interface with Traffic Anomalies

When the A-box confirmed that a particular victim (an IP address) is under DDoS attack, and if some specific conditions are satisfied, it will send a corresponding

confirmed attack alert to the P-box (see Section 4.4 for the conditions and the alert format). On receiving this alert, the P-box should start to find out which interfaces have abnormally high traffic volume coming in and targeted at the victim. This finding will be a good indicator that attack traffic is coming from those interfaces.

To achieve this objective, P-box needs to maintain the following counters about ingress packets to the switch after receiving the alert:

(1) total number of ingress packets of each protocol type (ICMP, UDP, TCP, TCP SYN) excluding packets destined to the victim, and egress TCP RST packets excluding packets from the victim;

(2) total number of ingress packets of each protocol type (ICMP, UDP, TCP, TCP SYN) from each interface excluding packets destined to the victim, and egress TCP RST packets to each interface excluding packets from the victim;

(3) number of ingress packets of each protocol type (ICMP, UDP, TCP, TCP SYN) destined to the victim, and egress TCP RST packets from the victim; and

(4) number of ingress packets of each protocol type (ICMP, UDP, TCP, TCP SYN) destined to the victim from each interface and egress TCP RST packets from the victim to each interface.

The source link layer address of the frame encapsulating a packet can be used to determine the inbound interface of the frame. The destination link layer address specifies the interface that the packet leaves the switch.

At the end of each response sample period, P-box should calculate the victim-skew ratio, Rv, for each interface of the switch and each protocol type:

$Rv_{(interface,\ protocol\ type)}$ = [(4) / (3)] / [(2) / (1)]

Victim-skew ratio measures if an interface has abnormally high proportion of traffic destined to a victim (or TCP RST from a victim) relative to the overall traffic, which is a good indicator that if the interface is a suspicious source of DDoS attack traffic.

The confidence level of a suspicion should be an increasing function of the ratio Rv (e.g. confidence = Rv / interface alert threshold ratio x 10). It can be formulated by performing statistical analysis on the traffic via Internet Exchange and backbone ISP. For any suspicion with confidence level higher than 10, an alert should be sent to the A-box and D-box. The traffic rate of the traffic from this interface and of this protocol type, i.e. (4) / response sample period, is also included in the alert for used by R-box to take response actions.

Alerts of confirmed DDoS attack received from A-box are valid only for two normal sample periods. Therefore, if no more alerts about a particular confirmed victim are received from A-box for two consecutive normal sample periods, P-box no longer needs to monitor the traffic for interfaces with traffic anomalies to this victim.

4.3.3. Communication with Other Components

Traffic Volume Anomaly Alert (Suspicious Attack Alert)

P-box sends traffic volume anomaly alerts to A-box and D-box when they are detected. The format of traffic volume anomaly alert is very similar to signature match alert by E-box, except for the following differences:

1. The classification name is always “DDoS – Traffic Volume Anomaly”.

2. There is no <Source> element. Since we check for suspicious attacks based on destination IP address, we have not collected any information about the source of the traffic. Moreover, since most DDoS attacks use spoofed agents, it is not meaningful to include the <Source> element in the message.
3. There is no <ToolAlert> element because traffic volume anomalies cannot tell which tool is used in the attack.
4. There is an <AdditionalData> element which specifies the protocol of anomalous traffic.

An example IDMEF message is as follows:

```
<IDMEF-Message version="0.1">
  <Alert alertid="12346.123456789" impact="successful-ddos">
    <Time offset="-0500">
      <ntpstamp>0x12346.0x67890</ntpstamp>
      <date>2000/03/09</date>
      <time>10:01:25.93464</time>
    </Time>
    <Analyzer ident="12346">
      <Node category="dns">
        <location>PolyU LDS</location>
        <name>Pbox1.polyu.edu.hk</name>
      </Node>
    </Analyzer>
    <Classification origin="vendor-specific">
      <name>DDoS – Traffic Volume Anomaly</name>
      <url>http://www.comp.polyu.edu.hk</url>
    </Classification>
    <Target>
      <Node ident="12346.tde796f70">
        <Address category="ipv4-addr-hex">
          <address>de796f70</address>
```

```
        </Address>
      </Node>
    </Target>
    <AdditionalData type=”string” meaning=”TargetRole”>
      Victim
    </AdditionalData>
    <AdditionalData type=”string” meaning=”Protocol”>
      UDP
    </AdditionalData>
    <AdditionalData type=”integer” meaning=”Confidence”>
      15
    </AdditionalData>
  </Alert>
</IDMEF-Message>
```

Interface Traffic Volume Anomaly Alert (Suspicious Interface Alert)

At the end of each response sample period, interface traffic volume anomaly alerts are sent from P-box to A-box and D-box, if there are any. All the alerts have classification names of “DDoS – Interface Traffic Volume Anomaly”. Each alert contains the concerned victim in the <Target> element. <AdditionalData> elements are used to specify the protocol of the anomalous traffic, the detected traffic rate (packet per second), and the interface that the suspicious packets come into the switch. The following is an example of the alert message:

```
<IDMEF-Message version="0.1">
  <Alert alertid="12345.123456789" impact="successful-ddos">
    <Time offset="-0500">
      <ntpstamp>0x12345.0x67890</ntpstamp>
      <date>2000/03/09</date>
      <time>10:01:25.93464</time>
    </Time>
    <Analyzer ident="12345">
      <Node category="dns">
```

```
        <location>PolyU LDS</location>
        <name>Ebox1.polyu.edu.hk</name>
      </Node>
    </Analyzer>
    <Classification origin="vendor-specific">
      <name>DDoS – Interface Traffic Volume Anomaly</name>
      <url>http://www.comp.polyu.edu.hk</url>
    </Classification>
    <Target>
      <Node ident="12345.tde796f70">
        <Address category="ipv4-addr-hex">
          <address>de796f70</address>
        </Address>
      </Node>
    </Target>
    <AdditionalData type=”string” meaning=”Protocol”>
      UDP
    </AdditionalData>
    <AdditionalData type=”real” meaning=”TrafficRate”>
      1247.5
    </AdditionalData>
    <AdditionalData type=”string” meaning=”FromInterface”>
      Interface A
    </AdditionalData>
    <AdditionalData type=”integer” meaning=”Confidence”>
      50
    </AdditionalData>
  </Alert>
</IDMEF-Message>
```

Heartbeat Message

Similar to the other components, P-box also needs to send IDMEF Heartbeat message to A-box and D-box periodically (every 30 seconds) to show it is alive.

4.4. A-box Design

In this section, I shall propose the design of A-box, or the analyzer. A-box is the brain of a LDS. It receives IDMEF messages from other components of the LDS, and also from LDSs at somewhere else in the Internet through the M-box. By consolidating and analyzing these messages, it confirms whether there is really a DDoS attack in progress such that response actions should be taken. It also monitors the heartbeat from different components and informs the console and other LDSs through M-box when some of them are down.

4.4.1. Alert Analysis of Suspicious Attack

There are three sources of suspicious attack alert messages to A-box. They are E-boxes, P-boxes, and other LDSs via the M-box. A-box needs to perform the followings on receiving these alert messages:

1. Consolidate local alert messages from E-boxes and P-boxes over a normal sample period and generate consolidated alert messages. Then send them to D-box for recording and other LDSs via the M-box at the end of a normal sample period.
2. Further consolidate the consolidated local alert messages and remote alert messages (from other LDSs via the M-box) to identify DDoS attack.
3. Inform the D-box and R-box if there is a confirmed attack for recording and response actions to be taken.

The above three tasks are described in detail in the following paragraphs.

Consolidate local alerts

At the end of each normal sample period, A-box should sum up the confidence levels of local alerts by IP address and role in an attack, with an upper limit of confidence level equals to 99. By limiting the confidence level of local alert, we can ensure that alerts from only one or a few FLDSs are not conclusive enough to confirm an attack, even they are of very high confidence level.

For example, assume three messages are received from E-boxes and P-boxes during a normal sample period with the following information:

1. Master = 111.222.111.222

   Daemon = 123.222.123.222

   Confidence = 10
2. Master = 111.222.111.222

   Daemon = 123.111.123.111

   Confidence = 10
3. Victim = 234.222.111.222

   Confidence = 10

A-box will generate four consolidated local alert messages with the following information and send to D-box and M-box:

1. Master = 111.222.111.222

   Confidence = 20 (= 10 + 10)
2. Daemon = 123.222.123.222

   Confidence = 10
3. Daemon = 123.111.123.111

Confidence = 10

4. Victim = 234.222.111.222

   Confidence = 10

Consolidate local and remote alerts

The A-box will then consolidate the consolidated local alert messages with remote alerts which are received within the last normal sample period. The consolidation process is similar to the consolidation of local alerts. However, there is no upper limit for the consolidated confidence level.

For example, in addition to the above local messages, if three remote alert messages are received with the following information:

1. Victim = 209.222.123.222

   Confidence = 15

2. Victim = 234.222.111.222

   Confidence = 30

3. Victim = 234.222.111.222

   Confidence = 20

the A-box will come up with the following conclusion:

1. Victim = 209.222.123.222

   Confidence = 15

2. Victim = 234.222.111.222

   Confidence = 60 (= 10 + 30 + 20)

Confirm an attack is in progress

The A-box should confirm there is a DDoS attack against an IP address if there are suspicious attack alerts with high confidence levels from many FLDSs distributed at different locations of the Internet. To put it in a formula, there is a confirmed attack if the consolidated confidence level for a particular victim is greater than:

Confidence Threshold x Total number of FLDS in the Internet x Coverage Percentage Threshold

Confidence threshold, coverage percentage threshold, and interface alert confidence threshold, which will be used for interface alert analysis, are defined as follows:

Parameter	**Description**	**Suggested Value**
Confidence threshold	Confidence level that is considered as high.	30
Coverage percentage threshold	Confirmed alert should be generated if more than this percentage of FLDSs in the Internet have generated confident suspicious alerts.	10%
Interface alert confidence threshold	Confirmed interface alert is generated if the consolidated confidence level of the suspicious interface alerts for an interface is greater than or equals to this threshold.	50

Table 4.2. Parameters used by A-box

For example, if the confidence threshold = 30, total number of FLDS = 100, and the coverage percentage threshold = 10%, the required minimum confidence level to trigger a confirmation is 300 (30 x 100 x 10%). If such a condition is satisfied, an IDMEF Alert message will be generated and sent to D-box for recording. Under any of the following two conditions, A-box will also send a confirmed attack alert to R-

box for response action:

1. The local P-box has also reported traffic volume anomaly against the victim in the last normal sample period; or
2. An upstream interface alert has been received from the downstream LDS within the last normal sample period, e.g. LDS-B (downstream LDS) alerts LDS-A (upstream LDS) that attack traffic is flowing from the Internet node A, where the LDS-A resides, to the Internet node B where the LDS-B resides.

4.4.2. Alert Analysis of Suspicious Interface

If an A-box needs to send a confirmed attack alert to R-box for response action, it will also send a confirmed attack alert to every P-box. The P-boxes will then start to monitor packets destined to the concerned victim IP address to detect for suspicious inbound interfaces where the attack traffic come from.

As described in Section 4.3, P-box will send suspicious interface alert (traffic volume anomalies from specific interface) messages to A-box if there are enough evidence. A-box then consolidates these findings to conclude if some interfaces are really forwarding attack traffic to a particular DDoS victim as follows:

$$\text{Confidence}_{(\text{Victim, Interface, current period})} = \frac{1}{2} \times \sum_{\substack{\text{suspicious interface alerts} \\ \text{from different P-boxes where} \\ \text{victim = Victim and} \\ \text{interface = Interface}}} \text{Confidence of suspicious interface alerts for the current period} + \frac{1}{2} \times \text{Confidence}_{(\text{Victim, Interface, last period})}$$

There may be multiple alerts for the same victim and attack interface from the same P-box, but for different protocols of traffic to the victim. Besides, TCP RST packets from the same victim and to the same interface may also be processed by different P-boxes. Therefore, we need to sum up the confidence of all the alerts for the same victim and interface in order to conclude if there is enough evidence to confirm there is an attack interface.

Introduction of the last period confidence in the equation avoids false alarm caused by a sudden burst of normal traffic.

If the consolidated confidence level for a particular victim and interface exceeds the interface alert confidence threshold, e.g. 50, a confirmation is made. A confirmed interface alert message is sent to D-box for recording and R-box for reaction. The confirmed interface alert message should specify each protocol type and the corresponding traffic rate that contributes to more than 25% of the aggregate confidence level.

For example, if three alerts are received from E-boxes and P-boxes for the same victim and attack interface, where: (1) protocol is UDP, confidence is 65; (2) protocol is ICMP, confidence is 10; and (3) protocol is TCP, confidence is 5. The aggregate confidence level is 80. If the last period confidence level is 70, the current period confidence level is 75, which is over the threshold 50. A confirmed interface alert is generated. Since the alert (1) contributes to 81.25% (65/80) of the total confidence, the alert should specifies protocol type UDP and the corresponding traffic rate. It may happen that a confirmed alert contains several protocol types, which is different from

the suspicious alerts received from E-box and P-box, which can only specify one protocol type.

However, if an A-box has alerted the P-boxes to start interface monitoring, but there are no sufficient evidence to confirm there are attack interfaces, the A-box should send a negative response alert to the P-boxes. The negative response alert is used to inform the P-boxes that the attack alert detected earlier is probably due to normal traffic, rather than attack traffic. Therefore, the P-boxes can adjust the maximum traffic rate counter for the "victim's" address upward if necessary, as discussed in Section 4.3.1.

4.4.3. Communication with other Components

A-box communicates with all other components in the LDS. It receives alert messages from: 1) E-boxes; 2) P-boxes; and 3) M-box. It also sends alert messages to 1) D-box; 2) R-box; and 3) M-box. The following table provides more detail view of what type of alerts are received from and sent to the other parties:

Alert Type	E-box	P-box	M-box	R-box
Suspicious attack alert	from	from	from & to	
Suspicious interface alert		from		
Confirmed attack alert		to		to
Confirmed interface alert				to
Upstream interface alert			from	
Negative response interface alert		to		

Table 4.3: Communication between A-box and other components

The format of suspicious attack alert message to M-box is identical to that in the last two sections. The format of confirmed attack alert message to P-box and R-box is similar to that of suspicious attack alert except that:

1. the confidence level is always 100;
2. the classification name is “DDoS – Confirmed Attack”;
3. there is no <AdditionalData> element for protocol type; and
4. there is no <AdditionalData> element for “TargetRole” because the target always refers to victim.

The format of confirmed interface alert to R-box is identical to that of suspicious interface alert except that:

1. the confidence level is always 100;
2. the classification name is “DDoS – Confirmed Attack Interface”; and
3. multiple protocols of attack traffic can be specified.

A-box also sends negative response interface alerts to P-boxes, if it has sent confirmed attack alerts to them in the last normal sample period, but it cannot confirm any interface alert after receiving suspicious interface alerts from E-boxes and P-boxes in this response sample period. The alert is identical to confirmed attack alert except the confidence level is 0. The format of a negative response alert is as follows:

```
<IDMEF-Message version="0.1">
  <Alert alertid="12347.123456789" impact="successful-ddos">
    <Time offset="-0500">
      <ntpstamp>0x12347.0x67890</ntpstamp>
```

```
      <date>2000/03/09</date>
      <time>10:01:25.93464</time>
    </Time>
    <Analyzer ident="12347">
      <Node category="dns">
        <location>PolyU LDS</location>
        <name>Abox.polyu.edu.hk</name>
      </Node>
    </Analyzer>
    <Classification origin="vendor-specific">
      <name>DDoS – Confirmed Attack</name>
      <url>http://www.comp.polyu.edu.hk</url>
    </Classification>
    <Target>
      <Node ident="12347.tde796f70">
        <Address category="ipv4-addr-hex">
          <address>de796f70</address>
        </Address>
      </Node>
    </Target>
    <AdditionalData type=”integer” meaning=”Confidence”>
      0
    </AdditionalData>
  </Alert>
</IDMEF-Message>
```

Besides alert message, A-box also receives IDMEF Heartbeat messages from all other components in the LDS. The heartbeat time interval is 30 seconds. If no heartbeat is received from any component in the LDS for 4 consecutive heartbeat periods, i.e. 2 minutes, it is assumed that this component is down.

A-box should also send heartbeat to the D-box and the M-box for every heartbeat period. If the S-box is down, or if over 50% of E-boxes or P-boxes are down, the LDS

can no longer function normally. In such cases, the A-box should not send any heartbeat to the M-box. Therefore the M-box will know the LDS is no longer working and will not send any heartbeat to other LDSs. Details about the communication between M-box and other LDSs are explained later in this dissertation.

4.5. D-box Design

In this section, I shall propose the design of D-box, or the database component. I shall also describe how XML query language can be used to select information from D-box.

D-box is the data repository for all event data. It receives IDMEF Alert messages from E-box, P-box, A-box, and M-box. It needs to store these messages. In other words, it needs to act as an XML database server because all IDMEF messages are based on XML.

D-box also receives IDMEF Heartbeat messages from all other components in the LDS. If no heartbeat is received from a component for 4 consecutive heartbeat periods, D-box needs to log down that component is down. The log entry should also be in form of XML, such that it can be stored in the same XML database. We can use IDMEF Alert message to record this kind of information, with the <Analyzer> element specifying the D-box itself, the <Classification> element specifying a component is down, and the <Target> element specifying the component that is down. The following is an example that recording Ebox1 is down:

```
<IDMEF-Message version="0.1">
  <Alert alertid="12348.123456789" impact="bad-unknown">
    <Time offset="-0500">
      <ntpstamp>0x12348.0x67890</ntpstamp>
      <date>2000/03/09</date>
      <time>10:01:25.93464</time>
    </Time>
```

```
    <Analyzer ident="12348">
      <Node category="dns">
        <location>PolyU LDS</location>
        <name>Dbox.polyu.edu.hk</name>
      </Node>
    </Analyzer>
    <Classification origin="vendor-specific">
      <name>LDS Component Down</name>
      <url>http://www.comp.polyu.edu.hk</url>
    </Classification>
    <Target>
      <Node category="dns">
        <location>PolyU LDS</location>
        <name>Ebox1.polyu.edu.hk</name>
      </Node>
    </Source>
  </Alert>
</IDMEF-Message>
```

D-box should support XML query language such that administrators can use console to query D-box for alert messages and components status. Since W3C has not come up with a standard of XML query language, we simply use XQL, one of the existing XML query languages, to demonstrate how alert messages can be retrieved. The syntax of XQL queries used in the following examples are identical to that of location paths used in XPath V1.0 [34] and matching patterns used in XLST V1.0 [35], both by W3C.

For example, an user can select all the confirmed DDoS attacks by selecting the current alert messages where (Analyzer = A-box) and (Confidence=100) as follows using XQL:

Alert[Analyzer/@IDENT=’12347’ and

AdditionalData/@meaning=’Confidence’ and

AdditionalData=’100’]

An user can also find all the locally detected events by selecting current messages where (Analyzer = E-box1 or Analyzer = P-box1 or . . .) as follows:

Alert[Analyzer/@IDENT=’12345’ or

Analyzer/@IDENT=’12346’ or

. . .]

Moreover, an user can also check if any component is down by selecting current messages where (Classification = “LDS Component Down”):

Alert[Classification/name=’LDS Component Down’]

4.6. R-box Design

In this section, I shall propose the design of R-box, or the response component. When R-box receives confirmed attack alerts or confirmed interface alerts from A-box, it needs to take response actions to stop the attack. There are three major actions that it can take:

1. Install traffic rate limit filters on the switch or routers.
2. Inform the LDS upstream of the suspected interface where the attack traffic is coming from.
3. For R-box located at local ISP, install ingress filter at the border routers to block packets from customer site with spoofed source addresses.

4.6.1. Traffic Rate Limit Filter

Most routers support the feature of traffic rate limiting. Limit can be set by interface, protocol type, source and destination IP address. Traffic satisfying the monitoring criteria and exceeding the rate limit will be dropped.

When R-box receives confirmed attack alert from A-box, but receives no confirmed interface alert in the last response sample period, it can ask all routers connected to the switch to install rate limit filter for traffic of the protocol types specified in the alert, and destined to the victim. If the switch also supports traffic rate limit filter, R-box can simply ask the switch to install filters on all ingress interfaces, rather than ask the routers to install the filters.

The traffic rate limit can be set to, say 80%, of the current traffic rate specified in the alert. Therefore, we can progressively reduce the traffic rate if the same alerts are received repeatedly.

When R-box receives confirmed interface alert from A-box, it can then instruct only the routers specified in the alert (or the specified interface of the switch) to install the traffic rate limit filter. It should also ask other routers (or other interfaces of the switch) to remove their filters for the same victim, if there is any. Therefore, only the traffic from the confirmed attack interfaces will be controlled.

This response action is inexpensive in terms of router resources, since the checking is limited to particular router and particular protocol type, in additional to particular destination IP address.

The disadvantage of this response action is normal packets via that router, with that protocol type, and to that destination address, will also be dropped proportionally because the filter cannot distinguish between normal and attack packets. However, since we have successfully limited the installation of filters on particular router and protocol types, we can save the normal traffic to the victim via other routers or of other protocol types. Moreover, normal traffic passing through other Internet nodes with no traffic anomalies can also arrive at and be processed by the site normally. It is much better than the case that the site is brought offline by the attack and all Internet users cannot access the site.

Confirmed attack alert and confirmed interface alert received from A-box are valid

only for two normal sample periods. Therefore, if no more alerts about a particular confirmed victim (or interface) is received from A-box for two normal sample periods, R-box should remove the corresponding traffic rate limit filters.

4.6.2. Upstream LDS

As explained in Section 4.4.2, assume that the A-box of a FLDS has concluded that there is an attack and traffic volume anomalies against the victim is detected locally (i.e. there is suspicious attack alert from local P-box against the victim), it will start to monitor local switch interfaces for traffic anomalies. If interface anomalies are found, it will send confirmed interface alerts to R-box. R-box can then notify the upstream neighbor LDS(s) about the alert through M-box. There may be multiple upstream LDSs for an interface because not all network nodes has LDS installed.

For the upstream LDS, it should have received or will receive the same suspicious attack alerts that the downstream LDS has received. Therefore, the upstream LDS should have confirmed or will confirm that there is a DDoS attack against the same victim. However, since the upstream LDS may detect no traffic volume anomaly against the victim locally (i.e. no suspicious attack alert is detected by local P-box), or it is a MLDS which does not detect for suspicious attack, it has not started any response action corresponds to the confirmed attack. But once the upstream interface alert is received from the downstream LDS, the upstream LDS will begin to take response action as described in Section 4.4.2, just like any local P-box which had also detected traffic volume anomalies.

Attack traffic caused by a few DDoS daemons are usually not significant and may

escape from the detection of a LDS. However, the aggregate traffic from many daemons is overwhelming and will be detected by LDS closer to the victim. The progressive trace back approach of this response action caters for this fact, such that DDoS attack traffic can be blocked effectively hop-by-hop backward towards the attack sources.

4.6.3. Ingress Filter

Because most DDoS attack packets have spoofed source addresses, we can block such attack by filtering spoofed packets. For LDS situating at local ISP, R-box can install ingress filter in the edge routers connecting to customer sites. Ingress filter prevents influx of packets with spoofed source IP addresses from the customer sites. Only packets with source addresses that are valid in the customer networks are allowed to pass through the filters.

Since these filters are installed only when there is a confirmed attack and the downstream LDS finds that attack traffic is coming from this network node, they will not degrade the performance of the ISP in normal situation. Moreover, similar to the case of traffic rate limit filter, if no more alerts about the confirmed victim are received from A-box for two normal sample periods, R-box should remove the corresponding ingress filters.

4.6.4. Communication with Other Components

Communication with Routers

If the switch at the network node does not support traffic rate limit filter, the R-box

may need to install traffic rate limit filters on the routers connected to the switch, which have attack traffic passing through. Therefore, communication is required between R-box and the routers. For the case that the routers and the switch are located at the same place physically, R-box can be connected to the routers through an internal LAN, which can be the same LAN which is used for routers management. For the case that the routers are scattered at different locations, such management network may not exist. Then, Internet may be required for the communication (via M-box).

Because the communication involves router configuration, security is very important, particularly if internal router management network is not available. Typical router can be configured by telnet and SNMP. However, both of them are not secured enough for this purpose. Typical router also supports Secure Shell (SSH) server such that SSH client can connect to it and configure it. SSH provides strong authentication and secure data communication, which satisfies our requirement. Therefore, it is proposed that R-box should act as a SSH client for router configuration.

Communication with M-box (and D-box)

As discussed above, a R-box sometimes needs to ask M-box to notify the upstream LDS about the finding that attack traffic is flowing from the upstream network node to its network node. R-box can achieve this objective by simply sending the corresponding confirmed interface alert received from the A-box, to the M-box, with some simplifications, e.g. <AdditionalData> for traffic rate and confidence level are not required. The alert should also be sent to D-box for recording. The following is an example alert:

```
<IDMEF-Message version="0.1">
```

```
  <Alert alertid="12349.123456789" impact="successful-ddos">
    <Time offset="-0500">
      <ntpstamp>0x12349.0x67890</ntpstamp>
      <date>2000/03/09</date>
      <time>10:01:25.93464</time>
    </Time>
    <Analyzer ident="12349">
      <Node category="dns">
        <location>PolyU LDS</location>
        <name>Rbox.polyu.edu.hk</name>
      </Node>
    </Analyzer>
    <Classification origin="vendor-specific">
      <name>DDoS – Interface Traffic Volume Anomaly</name>
      <url>http://www.comp.polyu.edu.hk</url>
    </Classification>
    <Target>
      <Node ident="12349.tde796f70">
        <Address category="ipv4-addr-hex">
          <address>de796f70</address>
        </Address>
      </Node>
    </Target>
    <AdditionalData type=”string” meaning=”Protocol”>
      UDP
    </AdditionalData>
    <AdditionalData type=”string” meaning=”Protocol”>
      ICMP
    </AdditionalData>
    <AdditionalData type=”string” meaning=”FromInterface”>
      Interface A
    </AdditionalData>
  </Alert>
</IDMEF-Message>
```

M-box will then forward the alert to the upstream LDS based on the “FromInterface” value.

Communication with A-box (and D-box)

A-box sends confirmed attack alerts and confirmed interface alert to R-box whenever necessary. Details about these alerts are discussed in Section 4.4. Besides, R-box also sends heartbeat messages to A-box and D-box at the end of each heartbeat period, just like the other components.

4.7. M-box Design

In this section, I shall propose the design of M-box, or the communication component. M-box is the only component of a LDS that has an Internet IP address. It shields the other components of a LDS from the outside network, such that an additional layer of security can be provided.

M-box receives suspicious attack alert from A-box. It then acts as an application-level gateway to forward suspicious attack alert to neighbor LDSs in the Internet, after replacing the <Analyzer> element of the alerts by its own information.

M-box also receives upstream interface alert from R-box. Based on the "FromInterface" value in the alert, M-box can identify the corresponding upstream neighbor LDS. Administrator should configure the upstream LDS for each switch interface using C-box beforehand. M-box should then forward the alert to the upstream LDS after replacing the <Analyer> element of the alert by its own information.

It also receives suspicious attack alert and upstream interface alert from neighbor LDSs. When an alert is received, it should perform the followings:

1. check the alert timestamp, drop it if it is older than 3 normal sample periods;
2. check the analyzer-id and alert-id against the list of received alerts (over the last 3 normal sample periods), drop it if it has been received before (e.g. because of LDSs looping);
3. if the alert is not dropped, record down the alert timestamp, analyzer-id, and alert-

id in the list of received alerts;

4. forward the alert to A-box for analysis and D-box for recording without any modification;
5. for suspicious attack alert, forward it to all neighbor LDSs, except the one which sent this alert .

M-box also receives heartbeat message from A-box periodically, which shows that the LDS is functioning normally. In order to convey this information to neighbor LDSs, it also needs to send heartbeats to them for every heartbeat interval. Moreover, the first heartbeat that an M-box sends to a neighbor should include a list of this LDS's neighbor LDSs. The same is true if the LDS has just re-configured its neighbor list. <AdditionalData> element of IDMEF message can be used for this purpose. The following is an example of this initialization heartbeat message:

```
<IDMEF-Message version="0.1">
  <Heartbeat heartbeatid="123456789">
    <Time offset="+0000">
      <ntpstamp>0x12345.0x67890</ntpstamp>
      <date>2000/03/09</date>
      <time>14:07:58</time>
    </Time>
    <Analyzer ident="12340">
      <Node category="dns">
        <name>Mbox.polyu.edu.hk</name>
      </Node>
    </Analyzer>
    <AdditionalData type=”string” meaning=”Neighbor LDS”>
      123.250.101.155
    </AdditionalData>
    <AdditionalData type=”string” meaning=”Neighbor LDS”>
```

```
      135.101.123.99
    </AdditionalData>
  </Heartbeat>
</IDMEF-Message>
```

On receiving this initialization heartbeat, an M-box should record down the neighbor's neighbors. After that, it is expected that heartbeat will be received from this neighbor every heartbeat period.

If heartbeats are not received from the neighbor for four or more consecutive heartbeat periods, it is assumed that this neighbor is down. The M-box should then dynamically reconfigure its neighbor LDS list, by replacing the failed neighbor with the neighbors of this failed neighbor. It then sends heartbeats to the original neighbors (except the failed neighbor) and the new neighbors, with its new neighbor list as <AdditionalData> elements. This communication method ensures the network of LDSs will not be broken down even if some LDSs are down.

Later on, if heartbeats are received from the failed neighbor LDS again, it is added back to the neighbor LDS list, while the neighbor's neighbors should be removed.

4.8. C-box Design

C-box, or the console, is responsible for the central administration of the whole LDS. It provides a graphical user interface (GUI) for the LDS administrator to configure the followings:

- IDs and IP addresses of different components.
- Normal sample period, response sample period, and heartbeat period.
- Traffic distribution rule of S-box.
- Signature database of E-box.
- Mapping of link layer addresses and IP addresses to the switch interfaces connecting to the corresponding routers, which is used by P-box.
- Minimum alert threshold and interface alert ratio threshold used by P-box
- Confidence threshold, interface alert confidence threshold, and coverage percentage threshold used by A-box
- Neighbor LDS lists of M-box
- Mapping of switch interface to the upstream neighbor LDS, which is used by M-box.

C-box should also provide an graphical user interface for an administrator to query D-box for alerts and components status. It transforms the queries from administrator into XML query language supported by D-box.

Chapter 5
SIMULATION OF THE GLOBAL DEFENSE SYSTEM

5.1. Simulation Overview

In order to find out the effectiveness of our global defense system against DDoS attacks, I have run simulations of DDoS attacks with and without the defense system by software simulation. Different parties in a DDoS attack are modeled in the simulation, including normal hosts, daemons, local ISPs and backbone ISPs. The simulation targets to find out whether the proposed global defense system can detect and stop a DDoS attack successfully.

In the simulation, I shall first measure how a DDoS attack can affect the volume of normal traffic that can arrive at and be processed by the victim. The same scenarios will then be simulated again by including the global defense system. Attacks with different number of daemons are tested to find out the effectiveness of the defense system under different scales of DDoS attack.

5.2. Simulation Facilities

The Network Simulator Version 2 (ns-2) is used in this simulation. NS is mainly developed by the DARPA-funded research project, VINT (Virtual InterNetwork Testbed), a collaboration among USC/ISI, Xerox PARC, LBNL, and UC Berkeley [36]. NS is a discrete event simulator targets for networking research, and is very widely used by the networking community. It offers many facilities which simplify the simulation works of this project, including network topology generator, network traffic generator, routing mechanism, queue monitoring, and network visualization.

5.3. Simulated Network Environment

An Internet-like network is simulated, which consists of approximately 300 network nodes. Each node represents any one of the followings:

1. Local ISP
2. POP of Backbone ISP
3. Gateway node of Backbone ISP

According to the Internet topology found in Section 2.2, the simulated network should consist of a number of regions. In each region, there are several Local ISP nodes and Backbone ISP POP nodes. Some of these Local ISP nodes have peer-to-peer links with Backbone ISP POP nodes. Some Local ISP nodes may also have direct connections to Backbone ISP nodes in other regions. For simplicity, local Internet Exchange is not modeled in the simulation. It should not affect the simulation result much since our focus is on cross-regional attack.

Backbone ISP nodes at different regions are interconnected by Gateway nodes. Some POP nodes may also have peer-to-peer links to other POP nodes. Gateway nodes of different Backbone ISP may also have peer-to-peer links.

Each Local ISP node or Backbone node may be an end-point of IP traffic. It may consist of hosts (IP traffic generator and receiver), sites (IP traffic generator and receiver), and DDoS daemons (IP traffic generator).

5.4. Simulation Implementation

The simulation can be broken down into eight steps. They are the followings:

1. To generate the network topology.
2. To place normal traffic generators at all ISP nodes.
3. To place DDoS daemons at some ISP nodes.
4. To place a queue monitor at the victim node for measuring packet drops.
5. To run the DDoS attack simulation and measure the traffic condition at the victim.
6. To place LDSs in the network.
7. To re-run the simulation and measure the traffic conditions at the victim and the LDSs.
8. To analyse the collected data.

For each scenario, the simulation is repeated for three times using different nodes as victims, such that an averaged result can be taken. Scenarios with different number and positions of the daemons are run to find out how the result changes when the scale of attack changes.

5.4.1. Network Topology Generation

GT-ITM Topology Generator is used to generate the network for simulation. The transit-stub graph model of GT-ITM closely resembles the topology we want to generate. Each graph consists of stub domains (local ISP) connected to transit domains (Backbone ISP), and different transit domains are also connected by transit-to-transit links.

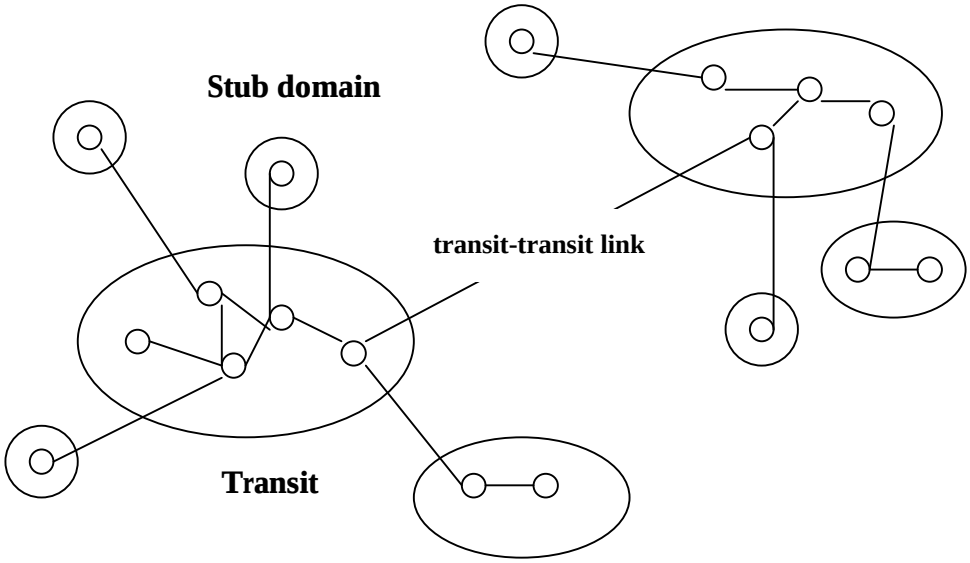


Figure 5.1: Transit-Stub Graph Model of GT-ITM

GT-ITM can generate a random transit-stub graph based on some input parameters. The parameters include the followings:

1. the average number of transit domains in the graph;
2. the average number of transit nodes in a transit domain;
3. the average number of stub domains connect to each transit node;
4. the average number of stub nodes in a stub domain;
5. the probability of having a link between any two transit domains;
6. the probability of having a link between any two transit nodes in a transit domain;
7. the probability of having a link between any two stub nodes in a stub domain;
8. the average number of extra transit-stub edges; and
9. the average number of extra stub-stub edges.

In the simulation, the simulated network has approximately one sixth of the number of regions and one sixth of the number of local ISPs per region as that of the actual Internet. Network of this size and complexity should be able to evaluate the effectiveness of the defense system, while the processing and memory resource requirement of the simulation is acceptable. Different input parameters are therefore set as follows:

Parameters	**Internet**	**Simulated Network**
1. Average number of transit domain (Backbone ISP)	appr. 50	8
2. Approximate number of region (assume one region per local Internet Exchange)	appr. 150	25
3. Average number of transit nodes per transit domain (assume coverage of an average backbone ISP = 20% of all regions for the Internet and 33% for the simulated network)	appr. 30	8
4. Average number of local ISP per region (for Internet: 10,000 ISPs / 150 regions)	appr. 60	10
5. Average number of backbone ISP with POP at any particular region [(1) x (3) / (2)]	appr. 10	3
6. Average number of stub domain (Local ISP) per transit node [(4) / (5)]	appr. 6	4
7. Average number of stub node per stub domain	-	1
8. Average number of extra transit-stub link [(2) x (4) x 10%]	900	25
9. Average number of extra stub-stub link	-	0
10. Probability of link between transit domain	-	0.3
11. Probability of link between transit nodes in a domain	-	0.3
12. Probability of link between stub nodes in a domain	-	N/A

Table 5.1: DDoS Attack Simulation – Graph generation parameters

For each stub domain, there is only one stub node because I assume a local ISP only has one network node in the simulation. Therefore the average number of nodes in a simulated network is approximately 320 (no. of transit domain x no. of transit nodes per transit domain x [1 + no. of stub domains per transit node x no. of stub nodes per stub domain]). The probabilities of having links between transit domains, transit nodes, and stub nodes, and the average number of extra transit-stub links are assigned arbitrarily, but the values should be reasonable.

Although the network generated by GT-ITM closely resembles the Internet topology, it is not good enough. First, the generated network does not differentiate the network bandwidth of transit links and local links. Therefore I need to modify the generated network topology by assigning higher bandwidth to links between transit nodes or backbone nodes (655 Mbps) and lower bandwidth to links between transit node and stub node (3 Mbps for normal stub nodes and 5 Mbps for popular stub nodes).

Second, the network does not differentiate between gateway and non-gateway node of backbone ISP. Therefore, we need to select the backbone nodes with more interfaces to other backbone nodes and treat them as gateway nodes for attaching LDSs.

Detail structure of the simulated network can be found in Appendix D.

5.4.2. Normal Traffic Generator Placement

To simulate the Internet, all the 320 nodes in the simulated network should play the roles of IP traffic generators and receivers. One objective of the simulation is to test

whether the defense system can successfully distinguish DDoS attack traffic from the normal or “noise” traffic.

NS has the feature that I can attach TCP or UDP agents (sender) to network nodes. I can also specify the destination of traffic from each agent. I can also set the traffic distribution model. For example, I can generate traffic according to a deterministic rate or alternatively according to an Exponential On/Off distribution (packets are sent at a fixed rate during on periods, and no packets are sent during off periods). I can also set the average transmission rate (during on periods), packet size, etc.

In the simulation, I write scripts to attach UDP agents to network nodes, such that for every node, the probability that it will send IP packets (normal traffic) to any other node is 0.04. Moreover, 8 specially selected nodes simulate popular network nodes such that every network node has a higher probability to send IP packets to these nodes. The probability is 0.2.

Flow-ID 0 is assigned to normal traffic to distinguish it from attack traffic. The followings are the detail settings used by the agents:

Parameters	**Value**
1. Traffic distribution	Exponential On/Off
2. Burst time (on time)	5 sec
3. Idle time (off time)	40 sec
4. Transmission rate	50 kbps
5. Packet size	500 bytes
6. Flow ID	0

Table 5.2: DDoS Attack Simulation – Normal traffic generation parameters

5.4.3. DDoS Daemon Placement

In the simulated network, a number of network nodes are selected to be daemon nodes. These nodes simulate ISPs with clients that have DDoS daemons implanted. I write scripts to attach UDP agents to these nodes to play the role of daemon. Different from the normal traffic generator, IP traffic from all daemon agents has the same destination, the victim node. Moreover, the transmission rate is much higher than that of normal traffic generator. The detail settings are as follows:

Parameters	**Value**
1. Traffic distribution	Constant Bit Rate
2. Burst time (on time)	N/A
3. Idle time (off time)	N/A
4. Transmission rate	1.5 Mbps
5. Packet size	500 bytes
6. Flow ID	99
7. Total number of daemon node	From 8 to 80 nodes, depending on which scenario

Table 5.3: DDoS Attack Simulation – Attack traffic generation parameters

5.4.4. Queue Monitor Placement

In order to find out the effect of DDoS attack on the victim, a facility offered by Ns is used to monitor the traffic condition at the victim node. This facility is Flow Monitor. By attaching a flow monitor to the link connected to the victim node, I can find out the accumulated number of IP packets arrived and dropped for different flow-id (0 for normal traffic and 99 for attack traffic) at different points of time.

Since the bandwidth assigned to transit-transit link is much higher than that of transit-stub link, no packet will be dropped before arriving at the link (transit-stub link) connected to the destination. As a result, packet drop rate can be measured simply by monitoring the link connected to the destination.

5.4.5. Local Detection Systems Modeling and Placement

Among the 320 network nodes, 18 nodes are selected as FLDS nodes as they have more connections to other backbone network nodes, 18 nodes are selected as MLDS nodes.

I writes a module (appr. 800 lines of C code and 200 lines of TCL script, refer to Appendix E for the design diagram) to play the role of FLDS. It performs the following functions:

1. Maintain a table to count the number of packets via this network node to each destination node for the current normal sample period.
2. Maintain a table of maximum number of packets via this network node to each destination node over the previous normal sample periods.
3. Generate alert to all other LDSs when traffic anomaly is detected, as proposed in Section 4.4.
4. Confirm whether there is a DDoS attack based on information received from all FLDSs, as proposed in Section 4.4.
5. Block suspicious traffic when there is a confirmed attack, as proposed in Section 4.6.
6. Detect which interface the attack traffic comes from and alert the upstream LDS, as proposed by Section 4.4 and 4.6.

The detection and decision parameters used in the simulation are as follows:

Parameter	Value
Normal sample period	2 sec
Response sample period	0.2 sec
Alert ratio (used by P-box)	4
Minimum alert threshold (used by P-box)	100 packets per sec
Confidence threshold (used by A-box)	0.35
Coverage threshold (used by A-box)	20%
Confirm threshold (used by A-box)	126 (#FLDS x 0.35 x 20%)
Interface alert ratio threshold (used by P-box)	2
Interface alert confidence threshold (used by P-box)	10

Table 5.4: DDoS Attack Simulation – Detection and Decision parameters

The normal sample period and response period used in the simulation are shorter than that proposed in this dissertation. It is because the simulated network has fewer network links and smaller link delays. The time required to send a packet from a source to a destination in the simulated network is shorter than that in the real Internet. The sample periods are shortened in order to increase the sensitivity of the defense system accordingly. Therefore, when interpreting the simulation result about how long it takes to detect an attack should be measured in terms of the number of normal sample periods required, rather than the absolute simulated time required.

The confidence thresholds used by A-box are also higher. It is because the number of FLDS node and network node in the simulated network are smaller, so the corresponding statistical deviations are larger than that for the real Internet. Therefore a less sensitive threshold should be used in the simulated network.

Since the objective of this simulation is to find out the effectiveness of the defense system on detecting and blocking attack, I have not implemented the proposed communication protocol in the simulation. Instead, procedure call is used to communicate alert information between different LDSs directly. As a result, the time required to communicate information among LDSs are ignored. It is acceptable because the time required to transmit messages from a source LDS to a destination LDS (several seconds in the Internet) is not significant when compared with the normal sample period (1 minute as proposed). When there is an attack, the paths from the attack sources to the victim may be congested. However, that should not affect the LDS message transmission much because the most critical messages are transmitting from the downstream LDS to the upstream, which are in opposite directions of the attack traffic, and the paths should not be congested.

S-box and E-box are also not modeled in the simulation. The load-balancing function of S-box is only required for actual production implementation, but is not meaningful to be included in our simulation. E-box is for collecting information from control messages of DDoS attack tools such that the actual attacker can be traced. This process involves manual intervention such as collecting and studying system logs at master and daemon machines. This area is not our interest in this simulation. In fact, our simulation only simulates attack traffic. It does not simulate any DDoS attack tool control message traffic.

For simplicity, our simulation only has UDP traffic. Therefore the simulated P-box does not classify traffic type (UDP, ICMP, etc.) when maintaining the hash tables.

For easier integration with NS, a FLDS object is attached to each inbound interface of each FLDS network node. For each FLDS node, one of the attached objects plays the manager role. It consolidates the information collected from its subordinate objects attached to other interfaces of the same network node, makes decision, and asks the subordinates to take response actions. The fundamental attack detection and response mechanism is the same as that proposed in this paper.

Another module (appr. 700 lines of code) is written to play the role of MLDS. It is very similar to the FLDS module except some functions of P-box and A-box are removed.

5.4.6. Simulation Run and Data Analysis

Each simulation lasts for 20 seconds of simulated time. Normal traffic begins to transmit at the start of the simulation. DDoS daemons begin the attack at second 5-6 randomly. The number of normal packets arrive and drop at the victim node is measured over the period of second 10-20, with and without the LDSs in place. The survival percentage of normal packets is then calculated as follows:

Survival ratio = (#normal packets arrive - #normal packets drop) /
(#normal packets arrive when there is no DDoS attack)

The packet-level average false positive ratio and false negative ratio are also measured as follows:

False positive ratio = $\sum_{LDS}$ #dropped normal packet / $\sum_{LDS}$ #normal packet

False negative ratio = $\sum_{LDS}$ #ignored attack packet / $\sum_{LDS}$ #attack packet

5.5. Simulation Result

Simulations of ten scenarios are run, with different number of daemon nodes. For each scenario, 3 independent simulations are run, with different locations of the victim node. The average result over the 3 samples is used. The whole process is repeated with and without our global defense system. In other words, a total of 60 simulations are run.

The simulation results show the followings:

1. All DDoS attacks can be detected, i.e. the false negative ratio by attack is 0%.
2. No confirmed alert is made against non-victim node by mistake, i.e. the false positive ratio by attack is 0%.
3. Each LDS can drop approximately 40% of the attack traffic, i.e. the packet-level false negative ratio is approximately 60%.
4. Less than 1% of normal traffic will be dropped mistakenly when there is a DDoS attack, i.e. the packet-level false positive ratio is less than 1% when there is an attack.
5. The ratio of normal packets that can arrive at and be processed by the victim node (packet survival ratio) increases by 60% with the defense system, when there are 5% to 12.5% of network nodes have DDoS daemons.
6. Over 85% of attack packets can be dropped before they arrive at the victim node. Therefore, congestion at the backbone can be avoided even if there is a large scale DDoS attack.

The above results are discussed in details in the following sections. The detail

simulation result can be found in Appendix F.

5.5.1. False Negative & False Positive by Attack

In all the simulated scenarios, it is found that the defense system can successfully detect that there is a DDoS attack against the victim node. Moreover, no confirmed alert is made against any non-victim node by mistake, although some suspicious alerts are made from time to time against them. It shows that even though burst traffic via some network nodes may trigger some suspicious attack alerts, it is highly unlikely that burst traffic to the same destination would occur at many different locations within a very short period. Therefore, the proposed infrastructure can effectively detect DDoS attacks with a low false alarm rate based on cooperation among distributed LDSs.

5.5.2. False Negative & False Positive by Packet

The packet-level average false positive ratio and false negative ratio at LDS nodes for the ten scenarios are summarized below:

% of nodes with daemon	False positive ratio (%)	False negative ratio (%)
0%	0.00	0.00
2.5%	0.32	81.43
5.0%	0.62	69.19
7.5%	0.51	65.24
10.0%	0.56	62.53
12.5%	0.52	67.56
15.0%	0.70	62.12

17.5%	0.62	58.51
20.0%	0.77	56.76
25.0%	0.60	57.75

Table 5.5: Simulation Result – False positive and false negative ratio

The false positive ratios are below 1% in all scenarios. Moreover, when there is no DDoS attack, no normal packets are dropped mistakenly, as shown by the 0% false positive ratio when the daemon coverage (% of nodes with daemon) equals 0%. It further confirms that the defense system can effectively detect an attack and at the same time avoid false alarm.

The relatively high false negative ratio is probably due to the number of transit-transit links per LDS node in the generated network is low, such that a LDS cannot precisely identify the source interfaces of attack traffic and block them. This point will be further discussed later on.

5.5.3. Packet Survival Ratio

To further evaluate if the infrastructure can effectively stop a DDoS attack, the survival percentage of normal packets are measured and summarized below:

% of nodes with daemon	Without LDSs		With LDSs	
	#normal packets processed by victim	Survival percentage (%)	#normal packets processed by victim	Survival percentage (%)
0%	9096	100	8895	100
2.5%	3604	39.80	3687	40.98
5.0%	1478	16.32	2564	28.51

7.5%	1133	12.60	1812	20.14
10.0%	974	10.80	1352	15.11
12.5%	676	7.55	1142	12.82
15.0%	632	7.04	801	8.91
17.5%	462	5.17	700	7.78
20.0%	437	4.86	473	5.28
25.0%	345	3.84	412	4.63

Table 5.6: Simulation Result – Survival percentage

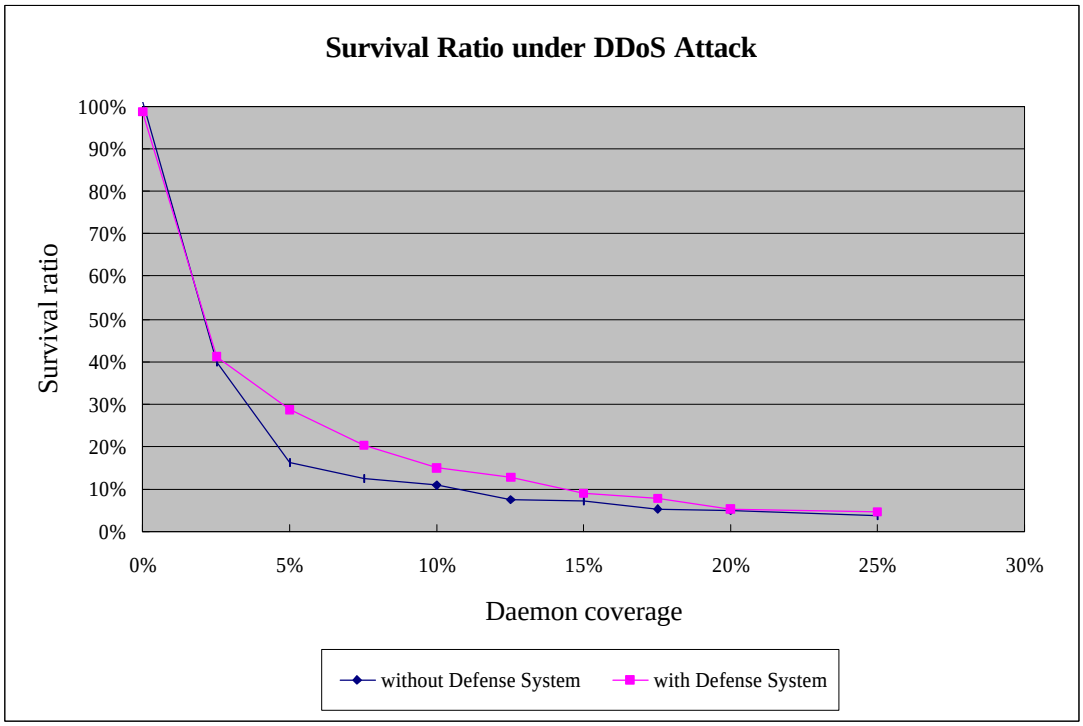


Figure 5.2: Survival Ratio under DDoS Attack

The result shows that the survival ratio with the defense system is much higher than that without the system, when the daemon coverage percentage ranges from 5.0% to 12.5%. It shows that our defense system not only can detect DDoS attack, it can also stop it to a certain extent effectively.

In the simulation, a packet from a source node to a destination node has any one of the following three fates:

1. arrive at and be processed by the destination node;
2. drop at the link connected to the destination because of congestion; and
3. drop by a LDS.

A packet will not be dropped at the backbone because the transit-transit bandwidth is much larger than the transit-stub bandwidth and no congestion will occur at the backbone even if there is a DDoS attack.

Therefore, the simulation results show that when the daemon coverage is between 5% and 12.5% inclusively, the LDSs can drop a large portion of the attack packets before they arrive at the victim. So, the congestion at the link connected to the victim can be relieved and the victim can receive and process more normal packets. As a result, the packet survival percentage is much higher than that without the defense system.

When there is a very small scale DDoS attack, e.g. the daemon coverage is 2.5%, the traffic volume anomalies detected are not very conclusive. The detection parameters used by the LDSs are not sensitive enough to block much attack packets under such circumstance. The packet-level false negative ratio is high (refer to Table 5.5). As a result, the packet survival percentage does not increase significantly when compared with that without the defense system.

When there is a very large scale DDoS attack, e.g. the daemon coverage increases to 20% or above, there is also no significant increase in packet survival percentage with

the defense system. It is because our defense system blocks attack traffic by trying to identify which interfaces the traffic comes from. However, when there are many daemons scattered at different nodes in the network, attack traffic actually come from all directions such that LDS can only block all traffic to the victim with a particular percentage. Therefore, the number of both the normal and attack packets arrive at the victim will decrease proportionally.

5.5.4. Blocking of Attack Packets

Although the survival ratio of normal packets to the victim will be low when there is a large scale DDoS attack, our defense system still can block a large number of attack packets at a early time. Therefore, the attack packets cannot compete for bandwidth with other normal traffic and cause traffic congestion to the backbone. The following table summarizes the number of attack packets arrive at the victim node with and without our defense system installed:

% of nodes with daemon	#attack packets arrive (without LDSs)	#attack packets arrive (with LDSs)
0%	0	0
2.5%	30,041	4,240
5.0%	80,588	7,987
7.5%	95,071	10,619
10.0%	112,896	12,089
12.5%	158,697	27,275
15.0%	166,798	27,259
17.5%	215,035	25,680
20.0%	224,212	35,366
25.0%	278,653	46,856

Table 5.7: Simulation Result – Attack packet arrival percentage

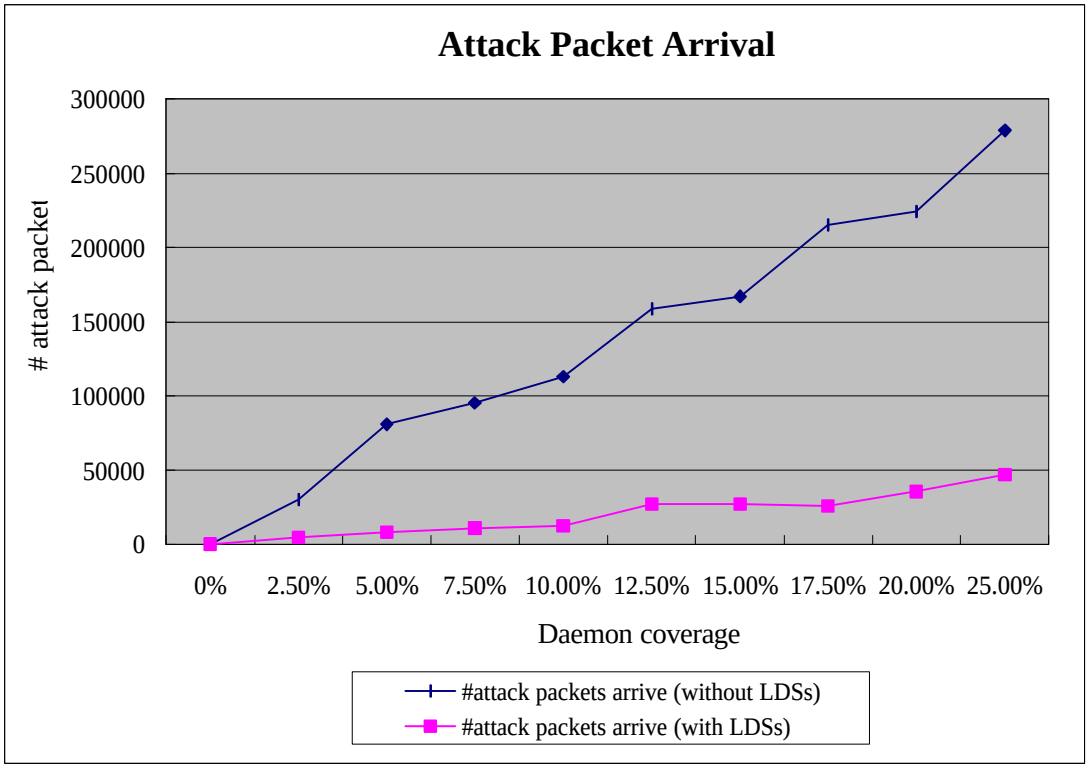


Figure 5.3: Attack packet arrival

The results show that more than 90% of attack packets are blocked well before they arrive at the victim node when the daemon coverage is less than 10%. Even when the daemon coverage increases to 20%, 85% of attack packets are blocked. Therefore the defense system should be able to prevent performance degradation of the overall Internet under large scale DDoS attack.

5.6. Simulation vs Actual Deployment

It should be noted that because of the heterogeneity and dynamic nature of the Internet, it is extremely difficult to simulate it [37]. Therefore this simulation only targets at modeling a simplified Internet with an emphasis on the network topology. Network topology is the fundamental factor which affects where LDSs should be installed and how the attack detection and response algorithm should work. Therefore, this simulation should provide a good indication that whether the proposed infrastructure will succeed or not.

Moreover, it is expected that the effectiveness of the defense system will be higher in the actual Internet environment than in this simulated network. It is because the simulated network is a randomly generated network. Although it has the concept of local ISP and backbone ISP, it does not have the concept of backbone gateway node and backbone POP node. I simply attach the LDSs to the backbone nodes with more interfaces to other nodes. As explained earlier, gateway backbone nodes in the actual Internet have many interfaces to different backbone ISP nodes and local ISP nodes. In this environment, LDS is more able to tell which interfaces the attack traffic come from and then block them. Therefore, our defense system should be able to function more effectively in that environment.

Furthermore, the attack detection and traffic blocking algorithms used in the simulation are simple and not fine tuned. By performing detail data analysis on the actual Internet traffic data and DDoS attack data, better algorithms and algorithm parameters can be found and used in the infrastructure, such that even higher attack

detection and blocking capability can be achieved.

Chapter 6
CONCLUSION AND FUTURE WORKS

6.1. Conclusion

The purpose of this dissertation is to propose an Internet-wide infrastructure to defend against DDoS attack. In Chapter 1, I have laid down six requirements of this infrastructure. They are the followings:

1. can minimize service interruption to DDoS victim;
2. should be cost effective;
3. can detect general DDoS attack;
4. can achieve a low false alarm rate;
5. can trace the actual attacker; and
6. can survive and function even itself is under attack.

In the proposed infrastructure, local detection systems (LDS) are installed at different locations in the Internet. They communicate in peer-to-peer mode to detect for and respond to DDoS attack cooperatively and automatically. This mechanism can block attack traffic within a short time interval once a DDoS attack is started.

Besides, I propose to classify LDS into full-blown LDS (FLDS) and minimal LDS (MLDS). FLDS is responsible for both attack detection and response, which consumes more computing resources, while MLDS is only responsible for attack response, which consumes much less computing resources. FLDSs should be installed at strategic locations in the Internet where most cross-domain traffic passes through, while MLDSs should be installed at other Internet nodes. Analysis of the Internet

topology and *traceroute* result both show that backbone ISP gateway nodes, Internet Exchanges, and Network Access Points are suitable Internet nodes to install FLDSs. Since the number of these locations is relatively small, the proposed system should be a cost-effective solution to the DDoS problem.

I propose attack detection and response algorithm based on traffic volume anomalies when designing LDS. These rules can handle general DDoS attack, rather than attack initiated by specific DDoS attack tools. Simulations show that the proposed detection algorithm is very effective in detecting DDoS attack. Moreover, false alarm rate is very low. Number of normal packets that the victim can receive and process when there is a DDoS attack also increases significantly when the defense system is installed. The defense system can also avoid traffic congestion to the Internet as a whole in case there is a large scale DDoS attack, as the proposed system can block a high proportion of the attack traffic early in the network

I also design an LDS component to detect for control packets of DDoS attack tools by matching IP datagrams with DDoS attack signatures. These control packets can provide clues to trace back the actual attacker and provide evidences for later law enforcement. I also propose a new approach to specify and store attack signature based on XML.

Moreover, I design a simple network protocol such that message from a LDS can reach every other LDS, even the LDS does not know the existence of the other. It simplifies the management of LDS. Furthermore, failure of any LDS will not affect the communication among the other LDSs. Therefore the infrastructure can function

even some LDSs are brought down by attack. I adopt and extend the Intrusion Detection Message Exchange Format (IDMEF) as the communication language among LDSs and among different components in a LDS. IDMEF messages are transmitted using Intrusion Alert Protocol (IAP) as the transport protocol, which provides the authentication and data security features to protect the communication from different kinds of attack.

Other than attack detection, analysis, response, and communication components, other components including load-balancing, database and console components are also designed. The different components compose a complete LDS, while LDSs and the communication protocol used among LDSs compose the infrastructure to defend against DDoS attack. As a conclusion, an infrastructure is designed to serve as a starting point to handle the DDoS attack problem. Moreover, requirements for the infrastructure specified in the project objective are all fulfilled.

6.2. Future Works

The proposed infrastructure is just a starting point to solve the DDoS problem. It identifies the different components that are required in the solution and how they interact. However, it is not a complete solution. Future works are required.

First, I have proposed to detect for and respond to DDoS attack based on traffic volume anomalies. DDoS detection and response algorithms are proposed with this concept in mind. However, the algorithms are intuitive in nature. Since this paper emphasizes on the whole infrastructure, the algorithms are not fine-tuned based on real Internet traffic data. Future works are required to enhance them. Alternatively, new algorithms can be designed and incorporated into the proposed infrastructure.

Second, in the proposed infrastructure, the attack detection algorithm dynamically estimates the normal traffic volume to any particular destination based on the past traffic pattern. This approach works in normal cases. However, there are some Web sites which have very dynamic traffic. For example, online stock brokerage Web site need to accommodate a very sudden increase of requests when the stock market opens. In this case, false alarm of confirmed attack may be triggered. As the LDSs fail to find any confirmed attack interface later on, they will adjust the attack detection rule to be less restrictive, such that the confirmed attack alert will be removed. Even though the alert will be removed eventually, a better alternative is to avoid the false alarm in the first place. It can be achieved by enhancing the infrastructure to allow an administrator to increase the alert threshold for some particular sites. The communication protocol then distribute this piece of information to all other LDSs.

As a result, even traffic to the site increases sharply, it would not trigger any confirmed attack alert as long as the volume is within the pre-determined threshold.

Third, this infrastructure has a limitation as it assumes all daemons will start their attacks against a victim within a short period. However, a very intelligent attacker can plan the attack start time of different daemons in such a way that the attack traffic increases gradually and evenly at different regions of the Internet, so that the attack can escape from the detection mechanism. However, beside a lot of planning is required, the beginning phase of such an attack will also be very long. Therefore, the victim site will probably aware of the attack before its service is interrupted. Therefore, it is possible to enhance the infrastructure to allow a site to communicate with a LDS such that a confirmed attack alert can be generated manually to trigger the attack response mechanism.

Fourth, although the proposed infrastructure has the capability to collect information about the locations of DDoS daemons, masters, and attackers, I have not come up with a detail design of how a LDS can automatically organize the collected information to locate different parties in a confirmed attack. Additional works are required to enhance the infrastructure on this aspect.

Fifth, since IDMEF is still under draft by IETF, its specification may change in near future. Therefore, the communication language of the proposed infrastructure may need to be modified to comply with the new standard.

References

[1] CERT Coordination Center. February 2000. “Internet Denial of Service Attacks and the Federal Response”. http://www.cert.org/congressional_testimony/Fithen_testimony_Feb29.html

[2] CERT Coordination Center. February 1996. “CERT Advisory CA-96.01 UDP Port Denial-of-Service Attack”. http://www.cert.org/advisories/CA-1996-01.html

[3] CERT Coordination Center. September 1996. “CERT Advisory CA-96.21 TCP SYN Flooding and IP Spoofing Attacks”. http://www.cert.org/advisories/CA-1996-21.html

[4] CERT Coordination Center. March 2000. “CERT Advisory CA-98.01 Smurf IP Denial-of-Service Attacks”. http://www.cert.org/advisories/CA-1998-01.html

[5] CERT Coordination Center. November 1999. “Results of the Distributed-Systems Intruder Tools Workshop”. http://www.cert.org/reports/dsit_workshop.pdf

[6] Paul J. Criscuolo, CIAC, US Department of Energy. February 2000. “Distributed Denial of Service – Trin00, Tribe Flood Network, Tribe Flood Network 2000, and Stacheldraht CIAC-2319”. http://ciac.llnl.gov/ciac/documents/CIAC-2319_Distributed_Denial_of_Service.pdf

[7] CERT Coordination Center. December 1999. “Distributed Denial of Service Tools”. http://www.cert.org/incident_notes/IN-99-07.html

[8] CERT Coordination Center. March 2000. “CERT Advisory CA-99-17 Denial-of-Service Tools”. http://www.cert.org/advisories/CA-1999-17.html

[9] Internet Security Systems (ISS). 2000. “Distributed Denial of Service Attack Tools”. http://documents.iss.net/whitepapers/ddos.pdf

[10] Xianjun Geng and Andrew B. Whinston. July 2000. “Defeating Distributed Denial of Service Attacks”. IT Pro July/August 2000.

[11] Phillip A. Porras and Peter G. Neumann. Oct 1997. “EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances”. National Information Systems Security Conference, October, 1997. http://www.sdl.sri.com/papers/emerald-niss97/

[12] G. Vigna and R.A. Kemmerer, University of California Santa Barbara. 1998. “NetSTAT: A Network-based Intrusion Detection Approach”. IEEE 14th Annual Computer Security Applications Conference. 7-11 December, 1998. Scottsdale, Arizona.

[13] M.C. Bernardes, E.S. Moreira. Jun 2000. “Implementation of an Intrusion Detection System Based on Mobile Agents”. Proceedings of the IEEE International Symposium on Software Engineering for Parallel and Distributed Systems (PDSE 2000)”. 10 - 11 June 2000. Limerick, Ireland.

[14] D. Schnackenberg and K. Djahandari. 1999. “Infrastructure for Intrusion Detection and Response”. DARPA Information Survivability Conference & Exposition Volume II of II. 25 - 27 January, 2000. Hilton Head, South Carolina.

[15] M.Y. Huang, R.J. Jasper, and T.M. Wicks. 1999. “A large scale distributed intrusion detection detection framework based on attack strategy analysis”. Computer Networks, Volume 31, Issue 23-24, 14 December 1999.

[16] Peter Mell, Donald Marks, and Mark McLarnon. 2000. “A denial-of-service resistant intrusion detection architecture”. Computer Networks, Volume 34, Issue 4, October 2000.

[17] Stefan Savage, David Wetherall, Anna Karlin, and Tom Anderson. University of Washington. 2000. “Practical Network Support for IP Traceback”. Proceedings of the 2000 ACM SIGCOMM Conference, pp. 295-306, Stockholm, Sweden, August 2000. http://www.cs.washington.edu/homes/savage/traceback.html

[18] Cisco. 1997. “Evolution of the Internet”. http://www.cisco.com/cpress/cc/td/cpress/design/isp/1ispint.htm

[19] The Cooperative Association for Internet Data Analysis (CAIDA). 1999. “Measurements of the Internet Topology in the Asia-Pacific Region”. http://www.caida.org/outreach/papers/asia_paper

[20] Hong Kong Internet Exchange. August 2000. “Presentation on HKIX”. http://www.hkix.net/hkix/hkix.ppt

[21] Chicago NAP. April 1999. “Chicago NAP Technical Information”. http://nap.aads.net/NAP_technical_info.html

[22] AT&T. 2000. “AT&T IP Network Backbone 3Q2000 View”. http://www.ipservices.att.com/backbone/

[23] Randolph C. Nicklas, vBNS. 1998. “Next Generation Backbone Networks”. http://www.vbns.net/presentations/krnet-ngen-backbone/index.html

[24] Cisco. July 2000. “Remote Monitoring Solutions for Switched Internetworks: A Comprehensive Strategy for Monitoring Enterprise Networks”. http://www.cisco.com/warp/public/cc/pd/nemnsw/sips/tech/remon_wp.htm

[25] J. Apisdorf, K. Claffy, K. Thompson, and R. Wilder. “OC3MON: Flexible, Affordable, High Performance Statistics Collection”. INET'97 Conference, June 1997. http://www.vbns.net/papers/inet97.pdf

[26] Julia Allen, Alan Christie, William Fithen, John McHugh, Jed Pickel, and Ed Stoner. Carnegie Mellon University and Software Engineering Institute. January 2000. “State of the Practice of Intrusion Detection Technologies”. CMU/SEI-99-TR-028. http://www.sei.cmu.edu/publications/documents/99.reports/99tr028/99tr028abstract.html

[27] J.S. Balasubramaniyan, J.O. Garcia-Fernandez, D. Isacoff, E. Spafford, and D. Zamboni. COAST Laboratory, Purdue University. June 1998. “An Architecture for Intrusion Detection using Autonomous Agents”. IEEE 14th Annual Computer Security Applications Conference, 7-11 December, 1998, Scottsdale, Arizona.

[28] CIDF Working Group. June 1998. “The Common Intrusion Detection Framework Architecture”. http://www.gidos.org/drafts/architecture.txt

[29] CIDF Working Group. March 2000. “A Common Intrusion Specification Language”. http://www.gidos.org/drafts/language.txt

[30] Intrusion Detection Working Group, IETF. July 2000. “Intrusion Detection Message Exchange Format – Extensible Markup Language (XML) Document Type Definition”. IETF Internet-Draft July 6, 2000. http://www.ietf.org/internet-drafts/draft-ietf-idwg-idmef-xml-01.txt

[31] Intrusion Detection Working Group, IETF. December 2000. “IAP: Intrusion Alert Protocol”. IETF Internet-Draft December 11, 2000. http://www.ietf.org/internet-drafts/draft-ietf-idwg-iap-03.txt

[32] Ulf Lindqvist, Phillip A. Porras. May 1999. “Detecting Computer and Network Misuse Through the Production-Based Expert System Toolset (P-BEST)”. Proceedings of the 1999 IEEE Symposium on Security and Privacy, Oakland, California, May 9–12, 1999. http://www.sdl.sri.com/emerald/pbest-sp99-cr.pdf

[33] Network Flight Recorder, Inc. Oct 1997. “Implementing A Generalized Tool For Network Monitoring”. Proceedings of the 11th Systems Administration Conference (LISA '97), San Diego, California October 26-31, 1997. http://www.nfr.com/forum/publications/LISA-97.htm

[34] W3C. 16 Nov 1999. “XML Path Language (XPath) Version 1.0”. W3C Recommendation November 16, 1999. http://www.w3.org/TR/xpath.html

[35] W3C. 16 Nov 1999. “Extensible Stylesheet Language Transformations (XSLT) Version 1.0”. W3C Recommendation November 16, 1999. http://www.w3.org/TR/xslt.html

[36] The VINT Project. “The Network Simulator - ns-2”. http://www.isi.edu/nsnam/ns/

[37] Vern Paxson, Sally Floyd. 1997. “Why we don’t know how to simulate the Internet”. Proceedings of the 1997 Winter Simulation Conference p.1037.

Appendix

Appendix A. List of Major Network Access Points (NAPs)

1. MAE East
2. Sprint NAP
3. Chicago NAP
4. Pacific Bell NAP
5. CIX Router
6. PAIX
7. MAE West
8. LINX (London Exchange)

Source from: Russ Haynal's ISP Page (http://navigators.com/isp.html)

Appendix B. List of Major Backbone ISPs

1. @Home Network
2. Above Net
3. AGIS (Net99)
4. Alternet (UUNET)
5. AT&T
6. BBN/GTE/genuity
7. Broadwing
8. CAIS
9. Concentric
10. Conxion
11. CRL
12. CWIX (Cable & Wireless)
13. DREN (Defense Research Engineering Network)
14. Digex/Intermedia
15. DRAnet
16. Data Exchange
17. Electric Lightwave
18. Epoch Networks
19. Exodus
20. ESnet (Energy Sciences Network)
21. E.Spire
22. Fiber Network Solutions
23. France Telecom
24. Global Crossing
25. Global One
26. GoodNet
27. IDT
28. Intira
29. Level 3
30. Nacamar
31. Nap.Net
32. NASA Internet
33. Net Access

34. NetRail
35. PSI
36. Savvis
37. Qwest
38. Sprintlink
39. Teleglobe
40. Telia
41. Terabit
42. VBNS
43. Verio (WNA)

Source from: Russ Haynal's ISP Page (http://navigators.com/isp.html)

Appendix C. Traceroute Analysis on Internet Topology

1. Eighteen *traceroute* servers are used in the study. They are:

Region	*traceroute* server
Asia	Hong Kong - Hong Kong - Hongkong Telecom Netplus (AS4637) China - ChinaNet (AS4808) Taiwan - National Chiao Tung University Taiwan (AS1659) Japan - Interlink (AS4698) Australia - Telstra (AS1221)
Europe	United Kingdom - XCIV.ORG (AS3328) France - Easynet France(AS6727) Germany - Cable & Wireless ECRC GmbH (AS1273) Switzerland - Global-IP Switzerland (AS6719) Russia - Novosibirsk State University (AS3335)
North America	Canada - Remote.Net (AS13594) USA - University of Arizona (AS1706) USA - Princeton University (AS88) USA - Iowa State (AS2698) USA - Stanford University (AS3671)
South America and Middle East	Mexico - Mazatlan (AS8151) Argentina - Cooperativa Telefonica Pinamar Ltda (AS14232) Israel - Ilan-Net (AS378)

2. From each *traceroute* server, fourteen traces are run to the following destinations:

Region	*traceroute* server
North America	yahoo.com (216.32.74.50) aol.com (205.188.172.126) infoseek.com (204.162.96.173) altavista.com (128.177.243.17) amazon.com (208.226.122.16)
Europe	iol.it (192.106.7.11) free.fr (212.27.32.114) spray.se (212.78.196.1) gmx.net (194.221.183.51) yahoo.co.uk (194.237.109.72)
South America, Middle East, and South Africa	br.yahoo.com (200.211.225.68) todito.com (200.23.36.21) globes.co.il (199.203.98.205) headlines.co.za (196.3.170.32)

Traceroute result for destinations to North America

Destinations in North America														
From	**To**	**AS Path / number of hops**										**# hops**	**# AS**	**#Back-bone ISP**	**# NAP & IX**
Hong Kong	yahoo.com	hkt.net(4637) 4	PAIX(183) 1	exodus.net(3967) 11								16	3	1	1
	aol.com	hkt.net(4637) 4	PAIX(183) 1	above.net(6461) 4	atdn.net(1668) 4	aol.com(10593) 1						14	5	1	1
	infoseek.com	hkt.net(4637) 5	alter.net(701) 4	infoseek.com(7266) 1								10	3	1	0
	altavista.com	hkt.net(4637) 4	PAIX(183) 1	mibh.net(3557) 3								8	3	0	1
	amazon.com	hkt.net(4637) 4	PAIX(183) 1	gblx.net(3549) 2	pnap.net(6993) 2	amazon.com(7224) 1						10	5	1	1
China	yahoo.com	bta.net.cn(4808) 1	chinanet.cn.net(4134) 3	nw.verio.net 1	verio.net 3	exodus.net(3967) 12						20	5	3	0
	aol.com	bta.net.cn(4808) 1	chinanet.cn.net(4134) 5	teleglobe.net(6453) 4	atdn.net(1668) 4	aol.com(10593) 2						16	5	1	0
	infoseek.com	bta.net.cn(4808) 1	chinanet.cn.net(4134) 3	cm.com.hk 1	alter.net(701) 4	infoseek.com(7266) 1						10	5	1	0
	altavista.com	bta.net.cn(4808) 1	chinanet.cn.net(4134) 3	nw.verio.net 1	verio.net 4	mibh.net(3557) 2						11	5	2	0
	amazon.com	bta.net.cn(4808) 1	chinanet.cn.net(4134) 3	teleglobe.net(6453) 1	sprintlink.net(1790) 7	amazon.com(7224) 1						13	5	2	0
Taiwan	yahoo.com	nctu.edu.tw(1659) 5	att.net(7018) 4	exodus.net(3967) 11								20	3	2	0
	aol.com	nctu.edu.tw(1659) 5	hinet.net(3462) 1	teleglobe.net(6453) 8	atdn.net(1668) 4	aol.com(10593) 2						20	5	1	0
	infoseek.com	nctu.edu.tw(1659) 6	att.net(7018) 4	level3.net 2	infoseek.com(7266) 1							13	4	2	0
	altavista.com	nctu.edu.tw(1659) 6	att.net(7018) 2	verio.net 4	mibh.net(3557) 2							14	4	2	0
	amazon.com	nctu.edu.tw(1659) 6	att.net(7018) 5	pnap.net(6993) 2	amazon.com(7224) 1							14	4	1	0
Japan	yahoo.com	interlink.ad.jp(4698) 1	mex.ad.jp(7514) 2	gblx.net(3549) 3	JPIX(7527) 1	exodus.net(3967) 13						20	5	2	1
	aol.com	interlink.ad.jp(4698) 1	mex.ad.jp(7514) 2	gblx.net(3549) 5	atdn.net(1668) 5	aol.com(10593) 2						15	5	1	0
	infoseek.com	interlink.ad.jp(4698) 1	mex.ad.jp(7514) 2	gblx.net(3549) 5	alter.net(701) 4	infoseek.com(7266) 1						13	5	2	0
	altavista.com	interlink.ad.jp(4698) 1	mex.ad.jp(7514) 2	gblx.net(3549) 5	PAIX(183) 1	mibh.net(3557) 2						11	5	1	1
	amazon.com	interlink.ad.jp(4698) 1	mex.ad.jp(7514) 2	gblx.net(3549) 4	pnap.net(6993) 1	amazon.com(7224) 1						9	5	1	0
Australia	yahoo.com	telstra.net(1221) 7	exodus.net(3967) 12									19	2	1	0
	aol.com	telstra.net(1221)	qwest.net(209)	atdn.net(1668)	aol.com(10593)							21	4	1	0

		7	8	4	2										
	infoseek.com	telstra.net(1221)	attmail.com	concert.net(5727)	prserv.net(2685)	pbi.net(5673)	infoseek.com(7266)					18	6	2	0
		6	1	3	5	2	1								
	altavista.com	telstra.net(1221)	PAIX(183)	verio.net	mibh.net(3557)							12	4	1	1
		7	1	2	2										
	amazon.com	telstra.net(1221)	PAIX(183)	verio.net	nw.verio.net	pnap.net(6993)	amazon.com(7224)					14	6	2	1
		7	1	2	1	2	1								
United Kingdom	yahoo.com	mailbox.net.uk(8401)	netkonect.net(3328)	LINX(5459)	exodus.net(3967)							13	4	1	1
		2	3	1	7										
	aol.com	mailbox.net.uk(8401)	netkonect.net(3328)	LINX(5459)	above.net(6461)	atdn.net(1668)	aol.com(10593)					14	6	1	1
		2	3	1	2	4	2								
	infoseek.com	mailbox.net.uk(8401)	netkonect.net(3328)	LINX(5459)	level3.net-Europe(9057)	level3.net	infoseek.com(7266)					12	6	2	1
		2	3	1	1	4	1								
	altavista.com	mailbox.net.uk(8401)	netkonect.net(3328)	LINX(5459)	above.net(6461)	mibh.net(3557)						11	5	1	1
		2	3	1	3	2									
	amazon.com	mailbox.net.uk(8401)	teleglobe.net - Europe	teleglobe.net(6453)	sprintlink.net(1790)	pnap.net(6993)	amazon.com(7224)					13	6	3	0
		2	3	1	5	1	1								
France	yahoo.com	easynet.fr(6727)	easynet.net(11341)	bbnplanet.net(1)	exodus.net(3967)							22	4	2	0
		3	5	7	7										
	aol.com	easynet.fr(6727)	easynet.net(11341)	teleglobe.net(6453)	atdn.net(1668)	aol.com(10593)						18	5	1	0
		3	5	4	4	2									
	infoseek.com	easynet.fr(6727)	FR-GIX	level3.net - Europe	level3.net	infoseek.com(7266)						11	5	2	1
		3	1	2	4	1									
	altavista.com	easynet.fr(6727)	easynet.net(11341)	teleglobe.net(6453)	mibh.net(3557)							17	4	1	0
		3	5	6	3										
	amazon.com	easynet.fr(6727)	easynet.net(11341)	bbnplanet.net(1)	pnap.net(6993)	amazon.com(7224)						23	5	1	0
		3	5	12	2	1									
Germany	yahoo.com	ecrc.net(1273)	exodus.net(3967)									15	2	2	0
		7	8												
	aol.com	ecrc.net(1273)	atdn.net(1668)	aol.com(10593)								16	3	1	0
		8	6	2											
	infoseek.com	ecrc.net(1273)	DE-CIX(65)	level3.net - Europe	level3.net	infoseek.com(7266)						14	5	3	1
		6	1	2	4	1									
	altavista.com	ecrc.net(1273)	AMS-IX(1200)	above.net(6461)	mibh.net(3557)							16	4	2	1
		7	1	6	2										
	amazon.com	ecrc.net(1273)	ebone.net(1755)	alter.net(701)	alter.net(702)	alter.net(701)	amazon.com(7224)					21	6	4	0
		3	10	5	1	1	1								
Switzer-land	yahoo.com	globalip.ch	gip.net(4000)	Sprint NAP(1673)	exodus.net(3967)							12	4	3	1
		2	3	1	6										
	aol.com	globalip.ch	gip.net(4000)	above.net(6461)	atdn.net(1668)	aol.com(10593)						15	5	3	0
		2	5	2	4	2									
	infoseek.com	globalip.ch	gip.net(4000)	MAE-East (701)	level3.net	infoseek.com(7266)						13	5	3	1
		2	5	1	4	1									
	altavista.com	globalip.ch	gip.net(4000)	above.net(6461)	mibh.net(3557)							12	4	3	0
		2	4	4	2										
	amazon.com	globalip.ch	gip.net(4000)	sprintlink.net(1790)	pnap.net(6993)	amazon.com(7224)						12	5	3	0
		2	2	5	2	1									
Russia	yahoo.com	nsu.ru(3335)	rbnet.ru(5568)	rt.ru(8342)	cw.net(3561)	exodus.net(3967)						14	5	2	0
		3	3	1	2	5									

	aol.com	nsu.ru(3335) 3	rbnet.ru(5568) 3	rt.ru(8342) 1	cw.net(3561) 3	atdn.net(1668) 5	aol.com(10593) 2					17	6	1	0
	infoseek.com	nsu.ru(3335) 3	rbnet.ru(5568) 3	rt.ru(8342) 2	teleglobe.net(6453) 2	alter.net(701) 7	infoseek.com(7266) 1					18	6	2	0
	altavista.com	nsu.ru(3335) 3	rbnet.ru(5568) 3	rt.ru(8342) 2	teleglobe.net(6453) 7	mibh.net(3557) 3						18	5	1	0
	amazon.com	nsu.ru(3335) 3	rbnet.ru(5568) 3	rt.ru(8342) 2	teleglobe.net(6453) 2	sprintlink.net(1790) 6	pnap.net(6993) 1	amazon.com(7224) 1				18	7	2	0
Canada	yahoo.com	remote.net(13594) 1	insinc.net(3602) 2	alter.net(702) 2	alter.net(701) 1	exodus.net(3967) 7						13	5	4	0
	aol.com	remote.net(13594) 1	insinc.net(3602) 3	sprintlink.net(1790) 4	teleglobe.net(6453) 7	atdn.net(1668) 4	aol.com(10593) 2					21	6	3	0
	infoseek.com	remote.net(13594) 1	insinc.net(3602) 2	alter.net(702) 1	alter.net(701) 5	infoseek.com(7266) 1						10	5	3	0
	altavista.com	remote.net(13594) 1	insinc.net(3602) 3	sprintlink.net(1790) 2	verio.net 4	mibh.net(3557) 2						12	5	3	0
	amazon.com	remote.net(13594) 1	insinc.net(3602) 3	sprintlink.net(1790) 2	pnap.net(6993) 2	amazon.com(7224) 1						9	5	2	0
USA1	yahoo.com	arizona.edu(1706) 4	qwest.net(209) 2	bbnplanet.net(1) 7	exodus.net(3967) 9							22	4	3	0
	aol.com	arizona.edu(1706) 3	qwest.net(209) 2	bbnplanet.net(1) 8	atdn.net(1668) 1	aol.com(10593) 2						16	5	2	0
	infoseek.com	arizona.edu(1706) 4	qwest.net(209) 3	cw.net(3561) 5	level3.net 4	infoseek.com(7266) 1						17	5	3	0
	altavista.com	arizona.edu(1706) 4	qwest.net(209) 2	bbnplanet.net(1) 4	verio.net 5	mibh.net(3557) 2						17	5	3	0
	amazon.com	arizona.edu(1706) 3	qwest.net(209) 3	cw.net(3561) 3	pnap.net(6993) 2	amazon.com(7224) 1						12	5	2	0
USA2	yahoo.com	princeton.edu(88) 2	verio.net 3	exodus.net(3967) 7								12	3	2	0
	aol.com	princeton.edu(88) 2	verio.net 3	above.net(6461) 2	atdn.net(1668) 4	aol.com(10593) 2						13	5	2	0
	infoseek.com	princeton.edu(88) 2	verio.net 4	alter.net(702) 1	alter.net(701) 5	infoseek.com(7266) 1						13	5	3	0
	altavista.com	princeton.edu(88) 2	verio.net 7	mibh.net(3557) 2								11	3	1	0
	amazon.com	princeton.edu(88) 2	cerf.net(1740) 3	sprintlink.net(1790) 6	pnap.net(6993) 2	amazon.com(7224) 1						14	5	1	0
USA3	yahoo.com	iastate.edu(2698) 6	att.net(7018) 7	exodus.net(3967) 9								22	3	2	0
	aol.com	iastate.edu(2698) 6	att.net(7018) 6	atdn.net(1668) 1	aol.com(10593) 2							15	4	1	0
	infoseek.com	iastate.edu(2698) 6	att.net(7018) 7	alter.net(701) 7	infoseek.com(7266) 1							21	4	2	0
	altavista.com	iastate.edu(2698) 6	att.net(7018) 9	above.net(6461) 4	mibh.net(3557) 2							21	4	2	0
	amazon.com	iastate.edu(2698) 6	att.net(7018) 9	pnap.net(6993) 2	amazon.com(7224) 1							18	4	1	0
USA4	yahoo.com	stanford.edu(3671) 1	slac.stanford.edu(32) 1	es.net(293) 1	MAE-East (701) 1	exodus.net(3967) 7						11	5	2	1

	aol.com	stanford.edu(3671) 2	slac.stanford.edu(32) 1	MAE-WEST (701) 1	above.net(6461) 3	atdn.net(1668) 4	aol.com(10593) 2					13	6	1	1
	infoseek.com	stanford.edu(3671) 1	slac.stanford.edu(32) 1	es.net(293) 1	alter.net(701) 2	alter.net(702) 1	alter.net(701) 2	infoseek.com(7266) 1				9	7	4	0
	altavista.com	stanford.edu(3671) 1	slac.stanford.edu(32) 1	es.net(293) 1	MAE-WEST (701) 1	verio.net 4	mibh.net(3557) 2					10	6	2	1
	amazon.com	stanford.edu(3671) 2	slac.stanford.edu(32) 1	nasa.gov(372) 1	icp.net(1239) 4	sprintlink.net(1790) 5	pnap.net(6993) 1	amazon.com(7224) 1				15	7	3	0
Mexico	yahoo.com	uninet.net.mx(6332) 3	gip.net(4000) 1	sprintlink.net(1790) 6	exodus.net(3967) 5							15	4	3	0
	aol.com	uninet.net.mx(6332) 3	gip.net(4000) 1	sprintlink.net(1790) 1	qwest.net(209) 5	atdn.net(1668) 4	aol.com(10593) 2					16	6	3	0
	infoseek.com	uninet.net.mx(6332) 3	gip.net(4000) 1	sprintlink.net(1790) 2	level3.net 2	infoseek.com(7266) 1						9	5	3	0
	altavista.com	uninet.net.mx(6332) 3	gip.net(4000) 1	sprintlink.net(1790) 4	verio.net 5	mibh.net(3557) 2						15	5	3	0
	amazon.com	uninet.net.mx(6332) 3	gip.net(4000) 1	sprintlink.net(1790) 6	pnap.net(6993) 3	amazon.com(7224) 1						14	5	2	0
Argentina	yahoo.com	tecoint.net(7303) 3	opentransit.net(5511) 6	MAE-East (701) 1	exodus.net(3967) 7							17	4	2	1
	aol.com	tecoint.net(7303) 3	gip.net(4000) 1	sprintlink.net(1790) 3	teleglobe.net(6453) 4	atdn.net(1668) 4	aol.com(10593) 2					17	6	3	0
	infoseek.com	tecoint.net(7303) 3	opentransit.net(5511) 2	level3.net 4	infoseek.com(7266) 1							10	4	2	0
	altavista.com	tecoint.net(7303) 3	opentransit.net(5511) 3	above.net(6461) 4	mibh.net(3557) 2							12	4	2	0
	amazon.com	tecoint.net(7303) 3	alter.net(701) 1	alter.net(702) 1	alter.net(701) 3	alter.net(702) 1	alter.net(701) 1	amazon.com(7224) 1				11	7	5	0
Israel	yahoo.com	ilan.net.il(378) 5	exodus.net(3967) 7									12	2	1	0
	aol.com	ilan.net.il(378) 2	dante-us(9010) 1	alter.net(701) 6	atdn.net(1668) 2	aol.com(10593) 2						13	5	1	0
	infoseek.com	ilan.net.il(378) 2	dante-us(9010) 1	alter.net(701) 7	infoseek.com(7266) 1							11	4	1	0
	altavista.com	ilan.net.il(378) 2	dante-us(9010) 1	alter.net(701) 2	alter.net(702) 1	alter.net(701) 1	above.net(6461) 4	mibh.net(3557) 2				13	7	4	0
	amazon.com	ilan.net.il(378) 2	dante-us(9010) 1	alter.net(701) 5	alter.net(702) 1	alter.net(701) 1	amazon.com(7224) 1					11	6	3	0

Number of traces	**Avg# hops**	**Avg# AS**	**Avg# Backbone**	**Avg# NAP&IX**
90	14.54	4.76	2.00	0.23

Traceroute result for destinations to Europe

Destinations in Europe														#Back-	# NAP
From	To	AS Path / number of hops										# hops	# AS	bone ISP	& IX
Hong Kong	iol.it	hkt.net(4637) 5	alter.net(701) 5	bbnplanet.net(1) 9	arcor-ip.net(3211) 5	iunet.it(1267) 3						27	5	2	0
	free.fr	hkt.net(4637) 5	alter.net(701) 6	teleglobe.net(6453) 8	teleglobe.net - Europe 6	proxad.net(12322) 3						28	5	3	0
	spray.se	hkt.net(4637) 5	alter.net(701) 6	alter.net(702) 6	utfors.net(8434) 1	spray.se(12383) 1						18	5	2	0
	gmx.net	hkt.net(4637) 5	alter.net(701) 5	alter.net(702) 5	de.alter.net(1270) 3	alter.net(701) 2	gmx.net(12722) 1					21	6	4	0
	yahoo.co.uk	hkt.net(4637) 5	alter.net(701) 2	alter.net(702) 2	alter.net(701) 1	us.telia.net(1833) 2	telia.net(3301) 4					16	6	5	0
China	iol.it	bta.net.cn(4808) 1	chinanet.cn.net(4134) 4	attmail.com 1	concert.net(5727) 1	bbnplanet.net(1) 10	arcor-ip.net(3211) 5	iunet.it(1267) 3				25	7	2	0
	free.fr	bta.net.cn(4808) 1	chinanet.cn.net(4134) 5	cm.com.hk 1	alter.net(701) 6	teleglobe.net(6453) 8	teleglobe.net - Europe 6	proxad.net(12322) 3				30	7	3	0
	spray.se	bta.net.cn(4808) 1	chinanet.cn.net(4134) 5	cm.com.hk 1	alter.net(701) 5	alter.net(702) 6	utfors.net(8434) 1	spray.se(12383) 1				20	7	2	0
	gmx.net	bta.net.cn(4808) 1	chinanet.cn.net(4134) 5	cm.com.hk 1	alter.net(701) 5	alter.net(702) 5	de.alter.net(1270) 3	alter.net(701) 2	gmx.net(12722) 1			23	8	4	0
	yahoo.co.uk	bta.net.cn(4808) 1	chinanet.cn.net(4134) 5	nw.verio.net 1	verio.net 2	PAIX(183) 1	us.telia.net(1833) 3	telia.net(3301) 4				17	7	4	1
Taiwan	iol.it	nctu.edu.tw(1659) 6	att.net(7018) 4	us.telia.net(1833) 4	arcor-ip.net(3211) 4	iunet.it(1267) 2						20	5	2	0
	free.fr	nctu.edu.tw(1659) 6	att.net(7018) 4	sprintlink.net(1790) 6	teleglobe.net(6453) 5	teleglobe.net - Europe 5	proxad.net(12322) 3					29	6	4	0
	spray.se	nctu.edu.tw(1659) 6	att.net(7018) 3	us.telia.net(1833) 2	telia.net(3301) 3	spray.se(12383) 1						15	5	3	0
	gmx.net	nctu.edu.tw(1659) 6	att.net(7018) 7	eqip.net(3300) 3	ecrc.net(1273) 2	gmx.net(12722) 1						19	5	2	0
	yahoo.co.uk	nctu.edu.tw(1659) 6	att.net(7018) 4	us.telia.net(1833) 2	telia.net(3301) 4							16	4	3	0
Japan	iol.it	interlink.ad.jp(4698) 1	mex.ad.jp(7514) 2	gblx.net(3549) 5	us.telia.net(1833) 4	arcor-ip.net(3211) 5	iunet.it(1267) 3					20	6	2	0
	free.fr	interlink.ad.jp(4698) 1	mex.ad.jp(7514) 2	gblx.net(3549) 6	teleglobe.net(6453) 3	teleglobe.net - Europe 6	proxad.net(12322) 3					21	6	3	0
	spray.se	interlink.ad.jp(4698) 1	mex.ad.jp(7514) 2	gblx.net(3549) 5	us.telia.net(1833) 2	telia.net(3301) 3	spray.se(12383) 1					14	6	3	0
	gmx.net	interlink.ad.jp(4698) 1	mex.ad.jp(7514) 2	gblx.net(3549) 5	alter.net(701) 4	alter.net(702) 7	de.alter.net(1270) 3	alter.net(701) 2	gmx.net(12722) 1			25	8	5	0
	yahoo.co.uk	interlink.ad.jp(4698) 1	mex.ad.jp(7514) 2	gblx.net(3549) 5	us.telia.net(1833) 2	telia.net(3301) 4						14	5	3	0
Australia	iol.it	telstra.net(1221) 7	PAIX(183) 1	us.telia.net(1833) 5	arcor-ip.net(3211) 5	iunet.it(1267) 3						21	5	1	1

	free.fr	telstra.net(1221) 7	bbnplanet.net(1) 4	teleglobe.net(6453) 3	teleglobe.net - Europe 6	proxad.net(12322) 3						23	5	3	0
	spray.se	telstra.net(1221) 7	PAIX(183) 1	us.telia.net(1833) 3	telia.net(3301) 3	spray.se(12383) 1						15	5	2	1
	gmx.net	telstra.net(1221) 7	bbnplanet.net(1) 8	ebone.net(1755) 10	ecrc.net(1273) 3	gmx.net(12722) 1						29	5	2	0
	yahoo.co.uk	telstra.net(1221) 7	PAIX(183) 1	us.telia.net(1833) 3	telia.net(3301) 4							15	4	2	1
United Kingdom	iol.it	mailbox.net.uk(8401) 2	teleglobe.net - Europe 4	iunet.it(1267) 3								9	3	1	0
	free.fr	mailbox.net.uk(8401) 2	teleglobe.net - Europe 5	proxad.net(12322) 3								10	3	1	0
	spray.se	mailbox.net.uk(8401) 2	teleglobe.net - Europe 1	LINX(5459) 1	telia.net(3301) 4	spray.se(12383) 1						9	5	2	1
	gmx.net	mailbox.net.uk(8401) 2	netkonect.net(3328) 2	LINX(5459) 1	ecrc.net(1273) 3	gmx.net(12722) 1						9	5	1	1
	yahoo.co.uk	mailbox.net.uk(8401) 2	teleglobe.net - Europe 1	LINX(5459) 1	telia.net(3301) 5							9	4	2	1
France	iol.it	easynet.fr(6727) 3	easynet.net(11341) 2	easynet.de(6659) 3	ecrc.net(1273) 1	arcor-ip.net(3211) 4	iunet.it(1267) 3					16	6	1	0
	free.fr	easynet.fr(6727) 3	proxad.net(12322) 4									**7**	**2**	**0**	**0**
	spray.se	easynet.fr(6727) 3	telia.net(3301) 6	spray.se(12383) 1								10	3	1	0
	gmx.net	easynet.fr(6727) 3	telia.net(3301) 4	DE-CIX(65) 1	ecrc.net(1273) 3	gmx.net(12722) 1						12	5	2	1
	yahoo.co.uk	easynet.fr(6727) 4	telia.net(3301) 7									11	2	1	0
Germany	iol.it	ecrc.net(1273) 4	arcor-ip.net(3211) 4	iunet.it(1267) 3								11	3	1	0
	free.fr	ecrc.net(1273) 6	DE-CIX(65) 1	telia.net(3301) 4	proxad.net(12322) 3							14	4	2	1
	spray.se	ecrc.net(1273) 6	DE-CIX(65) 1	telia.net(3301) 4	spray.se(12383) 1							12	4	2	1
	gmx.net	ecrc.net(1273) 5	gmx.net(12722) 1									6	2	1	0
	yahoo.co.uk	ecrc.net(1273) 6	DE-CIX(65) 1	telia.net(3301) 5								12	3	2	1
Switzerland	iol.it	globalip.ch 2	gip.net(4000) 5	MAE-East (701) 1	us.telia.net(1833) 4	arcor-ip.net(3211) 5	iunet.it(1267) 3					20	6	3	1
	free.fr	globalip.ch 2	gip.net(4000) 2	opentransit.net(5511) 4	proxad.net(12322) 3							11	4	3	0
	spray.se	globalip.ch 2	gip.net(4000) 3	SE-GIX(SE-GIX) 1	telia.net(3301) 3	spray.se(12383) 1						10	5	3	1
	gmx.net	globalip.ch 2	gip.net(4000) 3	eqip.net(3300) 7	ecrc.net(1273) 2	gmx.net(12722) 1						15	5	3	0
	yahoo.co.uk	globalip.ch 2	gip.net(4000) 3	SE-GIX(SE-GIX) 1	telia.net(3301) 4							10	4	3	1
Russia	iol.it	nsu.ru(3335)	rbnet.ru(5568)	rt.ru(8342)	cw.net(3561)	us.telia.net(1833)	arcor-ip.net(3211)	iunet.it(1267)				23	7	2	0

		3	3	1	5	3	5	3							
	free.fr	nsu.ru(3335)	rbnet.ru(5568)	rt.ru(8342)	telia.net(3301)	proxad.net(12322)						18	5	1	0
		3	3	2	7	3									
	spray.se	nsu.ru(3335)	rbnet.ru(5568)	rt.ru(8342)	telia.net(3301)	spray.se(12383)						14	5	1	0
		3	3	2	5	1									
	gmx.net	nsu.ru(3335)	rbnet.ru(5568)	rt.ru(8342)	cw.net(3561)	ecrc.net(1273)	gmx.net(12722)					16	6	2	0
		3	3	1	5	3	1								
	yahoo.co.uk	nsu.ru(3335)	rbnet.ru(5568)	rt.ru(8342)	telia.net(3301)							14	4	1	0
		3	3	2	6										
Canada	iol.it	remote.net(13594)	insinc.net(3602)	sprintlink.net(1790)	bbnplanet.net(1)	arcor-ip.net(3211)	iunet.it(1267)					25	6	3	0
		1	3	5	8	5	3								
	free.fr	remote.net(13594)	insinc.net(3602)	sprintlink.net(1790)	teleglobe.net(6453)	teleglobe.net - Europe	proxad.net(12322)					19	6	4	0
		1	3	4	3	5	3								
	spray.se	remote.net(13594)	insinc.net(3602)	sprintlink.net(1790)	alter.net(701)	alter.net(702)	utfors.net(8434)	spray.se(12383)				21	7	4	0
		1	3	4	5	6	1	1							
	gmx.net	remote.net(13594)	insinc.net(3602)	sprintlink.net(1790)	ebone.net(1755)	ecrc.net(1273)	gmx.net(12722)					24	6	3	0
		1	3	6	10	3	1								
	yahoo.co.uk	remote.net(13594)	insinc.net(3602)	sprintlink.net(1790)	us.telia.net(1833)	telia.net(3301)						16	5	4	0
		1	3	7	1	4									
USA1	iol.it	arizona.edu(1706)	qwest.net(209)	bbnplanet.net(1)	arcor-ip.net(3211)	iunet.it(1267)						21	5	2	0
		3	2	10	3	3									
	free.fr	arizona.edu(1706)	qwest.net(209)	bbnplanet.net(1)	teleglobe.net(6453)	teleglobe.net - Europe	proxad.net(12322)					26	6	4	0
		4	2	8	3	6	3								
	spray.se	arizona.edu(1706)	qwest.net(209)	bbnplanet.net(1)	us.telia.net(1833)	telia.net(3301)	spray.se(12383)					20	6	4	0
		4	2	7	3	3	1								
	gmx.net	arizona.edu(1706)	qwest.net(209)	cw.net(3561)	ecrc.net(1273)	gmx.net(12722)						18	5	3	0
		4	3	7	3	1									
	yahoo.co.uk	arizona.edu(1706)	qwest.net(209)	bbnplanet.net(1)	us.telia.net(1833)	telia.net(3301)						19	5	4	0
		3	2	7	3	4									
USA2	iol.it	princeton.edu(88)	cerf.net(1740)	att.net(7018)	us.telia.net(1833)	arcor-ip.net(3211)	iunet.it(1267)					20	6	2	0
		2	4	2	4	5	3								
	free.fr	princeton.edu(88)	verio.net	teleglobe.net(6453)	teleglobe.net - Europe	proxad.net(12322)						16	5	3	0
		2	3	2	6	3									
	spray.se	princeton.edu(88)	cerf.net(1740)	att.net(7018)	us.telia.net(1833)	telia.net(3301)	spray.se(12383)					14	6	3	0
		2	4	2	2	3	1								
	gmx.net	princeton.edu(88)	cerf.net(1740)	alter.net(702)	alter.net(701)	alter.net(702)	de.alter.net(1270)	alter.net(701)	gmx.net(12722)			21	8	5	0
		2	4	1	3	5	3	2	1						
	yahoo.co.uk	princeton.edu(88)	cerf.net(1740)	att.net(7018)	us.telia.net(1833)	telia.net(3301)						14	5	3	0
		2	4	2	2	4									
USA3	iol.it	iastate.edu(2698)	att.net(7018)	bbnplanet.net(1)	arcor-ip.net(3211)	iunet.it(1267)						33	5	2	0
		6	9	10	5	3									
	free.fr	iastate.edu(2698)	att.net(7018)	sprintlink.net(1790)	teleglobe.net(6453)	teleglobe.net - Europe	proxad.net(12322)					25	6	4	0
		6	5	5	1	5	3								
	spray.se	iastate.edu(2698)	att.net(7018)	us.telia.net(1833)	telia.net(3301)	spray.se(12383)						19	5	3	0
		6	7	2	3	1									
	gmx.net	iastate.edu(2698)	att.net(7018)	eqip.net(3300)	ecrc.net(1273)	gmx.net(12722)						21	5	2	0
		6	8	4	2	1									
	yahoo.co.uk	iastate.edu(2698)	att.net(7018)	us.telia.net(1833)	telia.net(3301)							19	4	3	0
		6	7	2	4										
USA4	iol.it	stanford.edu(3671)	slac.stanford.edu(32)	es.net(293)	bbnplanet.net(1)	arcor-ip.net(3211)	iunet.it(1267)					19	6	2	0

		1	1	1	8	5	3								
	free.fr	stanford.edu(3671)	slac.stanford.edu(32)	es.net(293)	Sprint NAP(1673)	teleglobe.net(6453)	teleglobe.net - Europe	proxad.net(12322)				14	7	3	1
		1	1	1	1	2	5	3							
	spray.se	stanford.edu(3671)	slac.stanford.edu(32)	MAE-East (701)	alter.net(702)	alter.net(701)	alter.net(702)	utfors.net(8434)	spray.se(12383)			17	8	3	1
		2	1	1	1	2	8	1	1						
	gmx.net	stanford.edu(3671)	slac.stanford.edu(32)	nasa.gov(372)	icp.net(1239)	ebone.net(1755)	ecrc.net(1273)	gmx.net(12722)				25	7	3	0
		2	1	1	6	11	3	1							
	yahoo.co.uk	stanford.edu(3671)	slac.stanford.edu(32)	es.net(293)	alter.net(701)	alter.net(702)	alter.net(701)	us.telia.net(1833)	telia.net(3301)			13	8	6	0
		1	1	1	2	1	1	2	4						
Mexico	iol.it	uninet.net.mx(6332)	gip.net(4000)	sprintlink.net(1790)	bbnplanet.net(1)	arcor-ip.net(3211)	iunet.it(1267)					22	6	3	0
		3	1	3	9	5	1								
	free.fr	uninet.net.mx(6332)	gip.net(4000)	sprintlink.net(1790)	teleglobe.net(6453)	teleglobe.net - Europe	proxad.net(12322)					18	6	4	0
		3	1	5	1	5	3								
	spray.se	uninet.net.mx(6332)	gip.net(4000)	sprintlink.net(1790)	alter.net(701)	alter.net(702)	alter.net(701)	alter.net(702)	utfors.net(8434)	spray.se(12383)		23	9	6	0
		3	1	3	1	2	3	8	1	1					
	gmx.net	uninet.net.mx(6332)	gip.net(4000)	sprintlink.net(1790)	ebone.net(1755)	ecrc.net(1273)	gmx.net(12722)					24	6	3	0
		3	1	6	9	4	1								
	yahoo.co.uk	uninet.net.mx(6332)	gip.net(4000)	sprintlink.net(1790)	us.telia.net(1833)	telia.net(3301)						14	5	4	0
		3	1	7	1	2									
Argentina	iol.it	tecoint.net(7303)	opentransit.net(5511)	sprintlink.net(1790)	bbnplanet.net(1)	arcor-ip.net(3211)	iunet.it(1267)					25	6	3	0
		3	5	3	6	5	3								
	free.fr	tecoint.net(7303)	gip.net(4000)	sprintlink.net(1790)	teleglobe.net(6453)	teleglobe.net - Europe	proxad.net(12322)					16	6	4	0
		3	1	3	1	5	3								
	spray.se	tecoint.net(7303)	opentransit.net(5511)	LINX(5459)	telia.net(3301)	spray.se(12383)						12	5	2	1
		3	4	1	3	1									
	gmx.net	tecoint.net(7303)	opentransit.net(5511)	AMS-IX(1200)	ecrc.net(1273)	gmx.net(12722)						11	5	2	1
		3	2	1	4	1									
	yahoo.co.uk	tecoint.net(7303)	opentransit.net(5511)	LINX(5459)	telia.net(3301)							12	4	2	1
		3	3	1	5										
Israel	iol.it	ilan.net.il(378)	dante-us(9010)	alter.net(701)	bbnplanet.net(1)	arcor-ip.net(3211)	iunet.it(1267)					24	6	2	0
		2	1	4	5	9	3								
	free.fr	ilan.net.il(378)	dante-us(9010)	alter.net(701)	teleglobe.net(6453)	teleglobe.net - Europe	proxad.net(12322)					16	6	3	0
		2	1	4	1	5	3								
	spray.se	ilan.net.il(378)	dante-us(9010)	alter.net(701)	alter.net(702)	utfors.net(8434)	spray.se(12383)					15	6	2	0
		2	1	5	5	1	1								
	gmx.net	ilan.net.il(378)	dante-us(9010)	alter.net(701)	alter.net(702)	de.alter.net(1270)	alter.net(701)	gmx.net(12722)				19	7	4	0
		2	1	5	5	3	2	1							
	yahoo.co.uk	ilan.net.il(378)	dante-us(9010)	alter.net(701)	us.telia.net(1833)	telia.net(3301)						11	5	3	0
		2	1	3	1	4									

Number of traces	Avg# hops	Avg# AS	Avg# Backbone	Avg# NAP&IX
90	17.67	5.38	2.69	0.21

Traceroute result for destinations to Middle East and Africa

Destinations in Middle East and Africa															
From	To	AS Path / number of hops										# hops	# AS	#Back-bone ISP	# NAP & IX
Hong Kong	br.yahoo.com	hkt.net(4637) 5	alter.net(701) 5	alter.net(702) 1	alter.net(701) 2	embratel.net.br(4230) 4						17	5	3	0
	todito.com	hkt.net(4637) 2	cw.net(3561) 5	avantel.net.mx(6503) 3	dataflux.com.mx 1							11	4	1	0
	globes.co.il	hkt.net(4637) 5	alter.net(701) 5	alter.net(702) 1	alter.net(701) 1	netvision.net.il(1680) 4						16	5	3	0
	headlines.co.za	hkt.net(4637) 1	cais.net(3491) 1	cwci.net(5551) 5	SAIX 4	gia.net.za(5710) 3						14	5	2	1
China	br.yahoo.com	bta.net.cn(4808) 1	chinanet.cn.net(4134) 4	cm.com.hk 1	alter.net(701) 5	alter.net(702) 1	alter.net(701) 2	embratel.net.br(4230) 4				18	7	3	0
	todito.com	bta.net.cn(4808) 1	chinanet.cn.net(4134) 4	cm.com.hk 1	alter.net(701) 5	alter.net(702) 1	alter.net(701) 1	avantel.net.mx(6503) 2	dataflux.com.mx 1			16	8	3	0
	globes.co.il	bta.net.cn(4808) 1	chinanet.cn.net(4134) 6	teleglobe.net(6453) 4	netvision.net.il(1680) 4							15	4	1	0
	headlines.co.za	bta.net.cn(4808) 1	chinanet.cn.net(4134) 4	cm.com.hk 1	alter.net(701) 2	alter.net(702) 2	alter.net(701) 1	SAIX 2	gia.net.za(5710) 3			16	8	3	1
Taiwan	br.yahoo.com	nctu.edu.tw(1659) 6	att.net(7018) 6	alter.net(701) 4	alter.net(702) 1	alter.net(701) 2	embratel.net.br(4230) 4					23	6	4	0
	todito.com	nctu.edu.tw(1659) 6	att.net(7018) 4	cw.net(3561) 4	avantel.net.mx(6503) 2	dataflux.com.mx 1						17	5	2	0
	globes.co.il	nctu.edu.tw(1659) 6	att.net(7018) 6	alter.net(701) 4	alter.net(702) 1	alter.net(701) 1	netvision.net.il(1680) 4					22	6	4	0
	headlines.co.za	nctu.edu.tw(1659) 6	att.net(7018) 6	alter.net(701) 2	alter.net(702) 1	alter.net(701) 1	SAIX 2	gia.net.za(5710) 3				21	7	4	1
Japan	br.yahoo.com	interlink.ad.jp(4698) 1	mex.ad.jp(7514) 2	gblx.net(3549) 5	alter.net(702) 1	alter.net(701) 3	alter.net(702) 1	alter.net(701) 2	embratel.net.br(4230) 4			19	8	5	0
	todito.com	interlink.ad.jp(4698) 1	mex.ad.jp(7514) 2	gblx.net(3549) 6	alter.net(702) 1	alter.net(701) 3	alter.net(702) 1	alter.net(701) 1	avantel.net.mx(6503) 2	dataflux.com.mx 1		18	9	5	0
	globes.co.il	interlink.ad.jp(4698) 1	mex.ad.jp(7514) 2	gblx.net(3549) 6	teleglobe.net(6453) 5	netvision.net.il(1680) 4						18	5	2	0
	headlines.co.za	interlink.ad.jp(4698) 1	mex.ad.jp(7514) 2	gblx.net(3549) 5	alter.net(702) 1	alter.net(701) 3	alter.net(702) 1	alter.net(701) 1	SAIX 2	gia.net.za(5710) 3		19	9	5	1
Australia	br.yahoo.com	telstra.net(1221) 6	attmail.com 1	concert.net(5727) 1	alter.net(701) 5	alter.net(702) 1	alter.net(701) 2	embratel.net.br(4230) 4				20	7	4	0
	todito.com	telstra.net(1221) 6	attmail.com 1	concert.net(5727) 1	alter.net(701) 5	alter.net(702) 1	alter.net(701) 1	avantel.net.mx(6503) 2	dataflux.com.mx 1			18	8	4	0
	globes.co.il	telstra.net(1221) 6	attmail.com 1	concert.net(5727) 1	alter.net(701) 5	alter.net(702) 1	alter.net(701) 1	netvision.net.il(1680) 4				19	7	4	0
	headlines.co.za	telstra.net(1221) 7	bbnplanet.net(1) 10	carrier1.net(8918) 5	interpacket.net(5097) 3	gia.net.za(5710) 3						28	5	1	0
United	br.yahoo.com	mailbox.net.uk(8401)	netkonect.net(3328)	alter.net(701)	alter.net(702)	alter.net(701)	embratel.net.br(4230)					15	6	3	0

Kingdom		2	3	1	3	2	4								
	todito.com	mailbox.net.uk(8401) 2	netkonect.net(3328) 3	alter.net(701) 1	alter.net(702) 5	alter.net(701) 1	avantel.net.mx(6503) 2	dataflux.com.mx 1				15	7	3	0
	globes.co.il	mailbox.net.uk(8401) 2	teleglobe.net - Europe 3	teleglobe.net(6453) 2	netvision.net.il(1680) 4							11	4	2	0
	headlines.co.za	mailbox.net.uk(8401) 2	netkonect.net(3328) 3	LINX(5459) 1	cwci.net(5551) 5	SAIX 4	gia.net.za(5710) 3					18	6	1	2
France	br.yahoo.com	easynet.fr(6727) 3	easynet.net(11341) 5	bbnplanet.net(1) 6	alter.net(701) 2	alter.net(702) 1	alter.net(701) 2	embratel.net.br(4230) 4				23	7	4	0
	todito.com	easynet.fr(6727) 3	easynet.net(11341) 5	bbnplanet.net(1) 6	cw.net(3561) 4	avantel.net.mx(6503) 3	dataflux.com.mx 1					22	6	2	0
	globes.co.il	easynet.fr(6727) 3	easynet.net(11341) 5	teleglobe.net(6453) 3	netvision.net.il(1680) 4							15	4	1	0
	headlines.co.za	easynet.fr(6727) 3	FR-GIX 1	carrier1.net(8918) 6	interpacket.net(5097) 3	gia.net.za(5710) 3						**16**	**5**	**0**	**1**
Germany	br.yahoo.com	ecrc.net(1273) 3	ebone.net(1755) 10	alter.net(701) 3	alter.net(702) 1	alter.net(701) 2	embratel.net.br(4230) 4					23	6	4	0
	todito.com	ecrc.net(1273) 5	cw.net(3561) 6	avantel.net.mx(6503) 3	dataflux.com.mx 1							15	4	2	0
	globes.co.il	ecrc.net(1273) 3	ebone.net(1755) 10	alter.net(701) 3	alter.net(702) 1	alter.net(701) 1	netvision.net.il(1680) 4					22	6	4	0
	headlines.co.za	ecrc.net(1273) 6	LINX(5459) 1	cwci.net(5551) 4	SAIX 4	gia.net.za(5710) 3						18	5	2	2
Switzerland	br.yahoo.com	globalip.ch 2	gip.net(4000) 2	sprintlink.net(1790) 4	alter.net(701) 3	alter.net(702) 1	alter.net(701) 2	embratel.net.br(4230) 4				18	7	6	0
	todito.com	globalip.ch 2	gip.net(4000) 2	sprintlink.net(1790) 3	cw.net(3561) 5	avantel.net.mx(6503) 3	dataflux.com.mx 1					16	6	4	0
	globes.co.il	globalip.ch 2	gip.net(4000) 2	sprintlink.net(1790) 4	teleglobe.net(6453) 2	netvision.net.il(1680) 4						14	5	4	0
	headlines.co.za	globalip.ch 2	gip.net(4000) 3	LINX(5459) 1	cwci.net(5551) 5	SAIX 4	gia.net.za(5710) 3					18	6	3	2
Russia	br.yahoo.com	nsu.ru(3335) 3	rbnet.ru(5568) 3	rt.ru(8342) 1	cw.net(3561) 3	concert.net(5727) 1	alter.net(702) 1	alter.net(701) 1	alter.net(702) 3	alter.net(701) 2	embratel.net.br(4230) 4	22	10	5	0
	todito.com	nsu.ru(3335) 3	rbnet.ru(5568) 3	rt.ru(8342) 2	teleglobe.net(6453) 5	alter.net(702) 1	alter.net(701) 3	alter.net(702) 1	alter.net(701) 1	avantel.net.mx(6503) 2	dataflux.com.mx 1	22	10	5	0
	globes.co.il	nsu.ru(3335) 3	rbnet.ru(5568) 3	rt.ru(8342) 1	teleglobe.net(6453) 3	netvision.net.il(1680) 4						14	5	1	0
	headlines.co.za	nsu.ru(3335) 3	rbnet.ru(5568) 2	rt.ru(8342) 2	teleglobe.net(6453) 4	alter.net(701) 1	alter.net(702) 4	alter.net(701) 2	interpacket.net(5097) 2	mcast.net(mcast.net) 1	gia.net.za(5710) 2	23	10	4	0
Canada	br.yahoo.com	remote.net(13594) 1	insinc.net(3602) 2	alter.net(702) 5	alter.net(701) 2	embratel.net.br(4230) 4						14	5	3	0
	todito.com	remote.net(13594) 1	insinc.net(3602) 2	alter.net(702) 1	alter.net(701) 3	alter.net(702) 1	alter.net(701) 1	avantel.net.mx(6503) 2	dataflux.com.mx 1			12	8	5	0
	globes.co.il	remote.net(13594) 1	insinc.net(3602) 2	alter.net(702) 5	alter.net(701) 1	netvision.net.il(1680) 4						13	5	3	0
	headlines.co.za	remote.net(13594) 1	insinc.net(3602) 3	sprintlink.net(1790) 7	carrier1.net(8918) 5	interpacket.net(5097) 3	gia.net.za(5710) 3					22	6	2	0
USA1	br.yahoo.com	arizona.edu(1706)	qwest.net(209)	bbnplanet.net(1)	alter.net(701)	alter.net(702)	alter.net(701)	embratel.net.br(4230)				20	7	5	0

		3	2	3	5	1	2	4							
	todito.com	arizona.edu(1706)	qwest.net(209)	cw.net(3561)	avantel.net.mx(6503)	dataflux.com.mx						16	5	2	0
		3	3	6	3	1									
	globes.co.il	arizona.edu(1706)	qwest.net(209)	bbnplanet.net(1)	alter.net(701)	alter.net(702)	alter.net(701)	netvision.net.il(1680)				20	7	5	0
		4	2	3	5	1	1	4							
	headlines.co.za	arizona.edu(1706)	qwest.net(209)	bbnplanet.net(1)	carrier1.net(8918)	interpacket.net(5097)	mcast.net	gia.net.za(5710)				27	7	2	0
		3	2	10	5	3	1	3							
USA2	br.yahoo.com	princeton.edu(88)	cerf.net(1740)	alter.net(702)	alter.net(701)	embratel.net.br(4230)						14	5	2	0
		2	4	2	2	4									
	todito.com	princeton.edu(88)	cerf.net(1740)	alter.net(702)	alter.net(701)	alter.net(702)	alter.net(701)	avantel.net.mx(6503)	dataflux.com.mx			15	8	4	0
		2	4	1	3	1	1	2	1						
	globes.co.il	princeton.edu(88)	cerf.net(1740)	alter.net(702)	alter.net(701)	netvision.net.il(1680)						14	5	2	0
		2	4	3	1	4									
	headlines.co.za	princeton.edu(88)	cerf.net(1740)	alter.net(702)	alter.net(701)	interpacket.net(5097)	mcast.net(mcast.net)	gia.net.za(5710)				16	7	2	0
		2	4	2	2	2	1	3							
USA3	br.yahoo.com	iastate.edu(2698)	att.net(7018)	alter.net(701)	alter.net(702)	alter.net(701)	embratel.net.br(4230)					25	6	4	0
		6	7	3	3	2	4								
	todito.com	iastate.edu(2698)	att.net(7018)	cw.net(3561)	avantel.net.mx(6503)	dataflux.com.mx						23	5	2	0
		6	9	4	3	1									
	globes.co.il	iastate.edu(2698)	att.net(7018)	alter.net(701)	alter.net(702)	alter.net(701)	netvision.net.il(1680)					24	6	4	0
		6	7	3	3	1	4								
	headlines.co.za	iastate.edu(2698)	att.net(7018)	alter.net(701)	alter.net(702)	alter.net(701)	interpacket.net(5097)	gia.net.za(5710)				25	7	4	0
		6	7	3	2	2	2	3							
USA4	br.yahoo.com	stanford.edu(3671)	slac.stanford.edu(32)	MAE-East (701)	alter.net(702)	alter.net(701)	embratel.net.br(4230)					15	6	2	1
		2	1	1	5	2	4								
	todito.com	stanford.edu(3671)	slac.stanford.edu(32)	es.net(293)	MAE-East (701)	alter.net(702)	alter.net(701)	alter.net(702)	alter.net(701)	avantel.net.mx(6503)	dataflux.com.mx	13	10	5	1
		1	1	1	1	1	3	1	1	2	1				
	globes.co.il	stanford.edu(3671)	slac.stanford.edu(32)	Sprint NAP(1673)	alter.net(701)	alter.net(702)	alter.net(701)	netvision.net.il(1680)				13	7	3	1
		2	1	1	1	3	1	4							
	headlines.co.za	stanford.edu(3671)	slac.stanford.edu(32)	alter.net(701)	alter.net(702)	alter.net(701)	SAIX	gia.net.za(5710)				13	7	3	1
		2	1	2	2	1	2	3							
Mexico	br.yahoo.com	uninet.net.mx(6332)	gip.net(4000)	sprintlink.net(1790)	alter.net(701)	alter.net(702)	alter.net(701)	embratel.net.br(4230)				19	7	5	0
		3	1	3	1	5	2	4							
	todito.com	uninet.net.mx(6332)	gip.net(4000)	sprintlink.net(1790)	cw.net(3561)	avantel.net.mx(6503)	dataflux.com.mx					13	6	3	0
		3	1	3	2	3	1								
	globes.co.il	uninet.net.mx(6332)	gip.net(4000)	sprintlink.net(1790)	alter.net(701)	alter.net(702)	alter.net(701)	netvision.net.il(1680)				18	7	5	0
		3	1	3	1	5	1	4							
	headlines.co.za	uninet.net.mx(6332)	gip.net(4000)	sprintlink.net(1790)	carrier1.net(8918)	interpacket.net(5097)	gia.net.za(5710)					22	6	2	0
		3	1	7	5	3	3								
Argentina	br.yahoo.com	tecoint.net(7303)	embratel.net.br(4230)									**10**	**2**	**0**	**0**
		3	7												
	todito.com	tecoint.net(7303)	seabone.net(6762)	alter.net(701)	alter.net(702)	alter.net(701)	alter.net(702)	alter.net(701)	avantel.net.mx(6503)	dataflux.com.mx		15	9	5	0
		3	2	1	1	3	1	1	2	1					
	globes.co.il	tecoint.net(7303)	gip.net(4000)	sprintlink.net(1790)	teleglobe.net(6453)	netvision.net.il(1680)						13	5	3	0
		3	1	3	2	4									
	headlines.co.za	tecoint.net(7303)	agis.net(4200)	alter.net(701)	alter.net(702)	alter.net(701)	interpacket.net(5097)	gia.net.za(5710)				17	7	4	0
		3	2	1	1	5	2	3							
Israel	br.yahoo.com	ilan.net.il(378)	dante-us(9010)	alter.net(701)	alter.net(702)	alter.net(701)	embratel.net.br(4230)					13	6	3	0
		2	1	2	2	2	4								

	todito.com	ilan.net.il(378) 2	dante-us(9010) 1	alter.net(701) 5	alter.net(702) 1	alter.net(701) 1	avantel.net.mx(6503) 2	dataflux.com.mx 1				13	7	3	0
	globes.co.il	ilan.net.il(378) 5	netvision.net.il(1680) 2									**7**	**2**	**0**	**0**
	headlines.co.za	ilan.net.il(378) 4	nap.net(5646) 1	bbnplanet.net(1) 11	carrier1.net(8918) 6	interpacket.net(5097) 3	gia.net.za(5710) 3					28	6	2	0

Number of traces	Avg# hops	Avg# AS	Avg# Backbone	Avg# NAP&IX
72	17.60	6.28	3.08	0.21

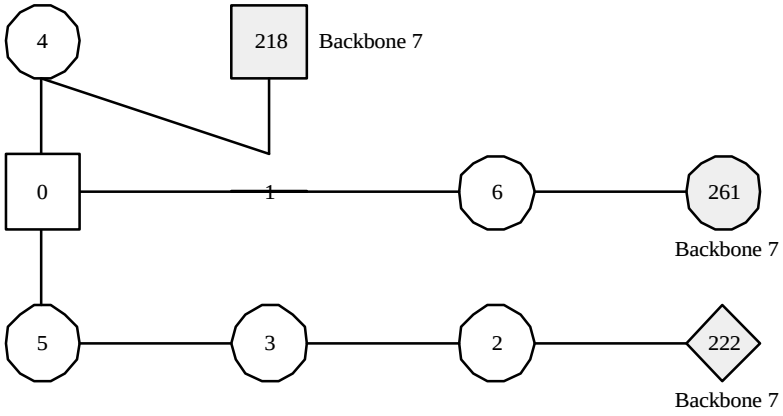
Overall Traceroute Result

Number of traces	Avg# of hops	Avg# of AS	Avg# Backbone	Avg# NAP & IX
252	16.53	5.41	2.56	0.22

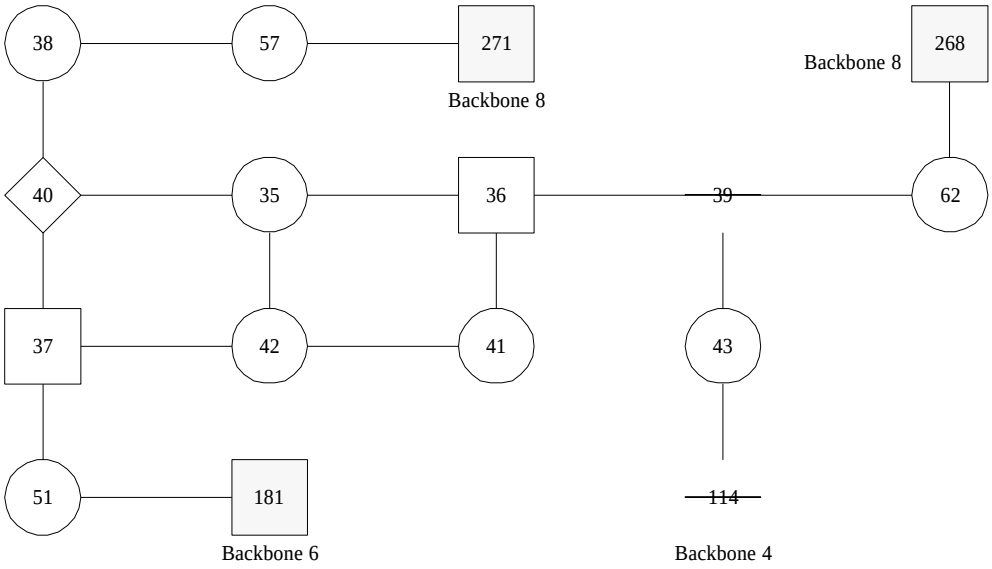
Appendix D. Topology of the Simulated Network

The simulated network consists of 320 network nodes. There are 8 transit domains, which represent the backbone ISPs, connecting the stub domains together. The structures of the 8 transit domains (backbone) are as follows:

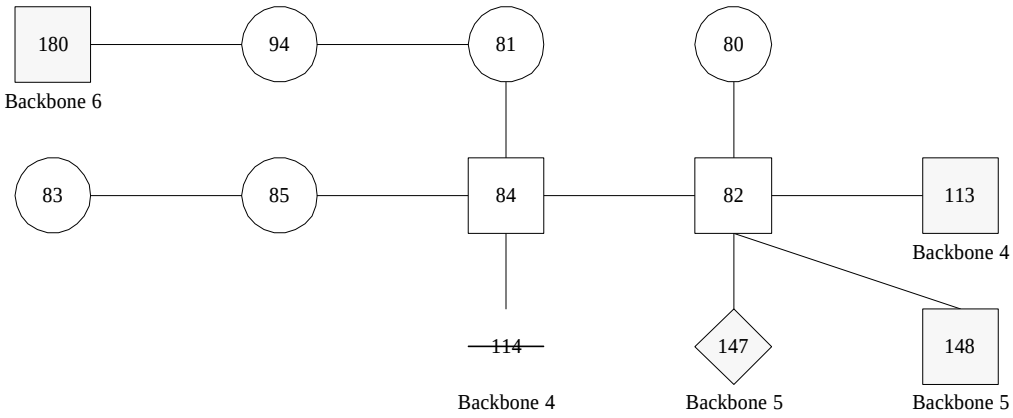
Backbone 1:



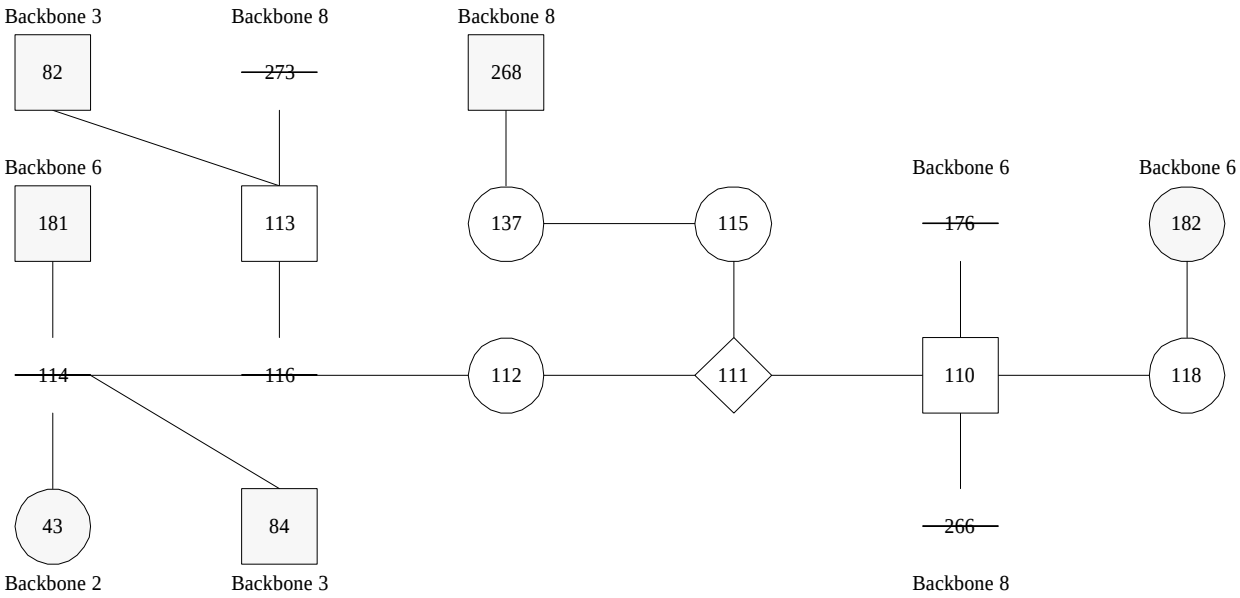
Backbone 2:



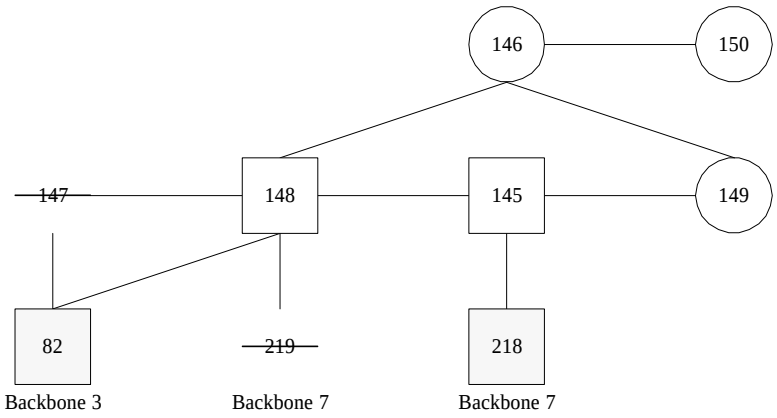
Backbone 3:



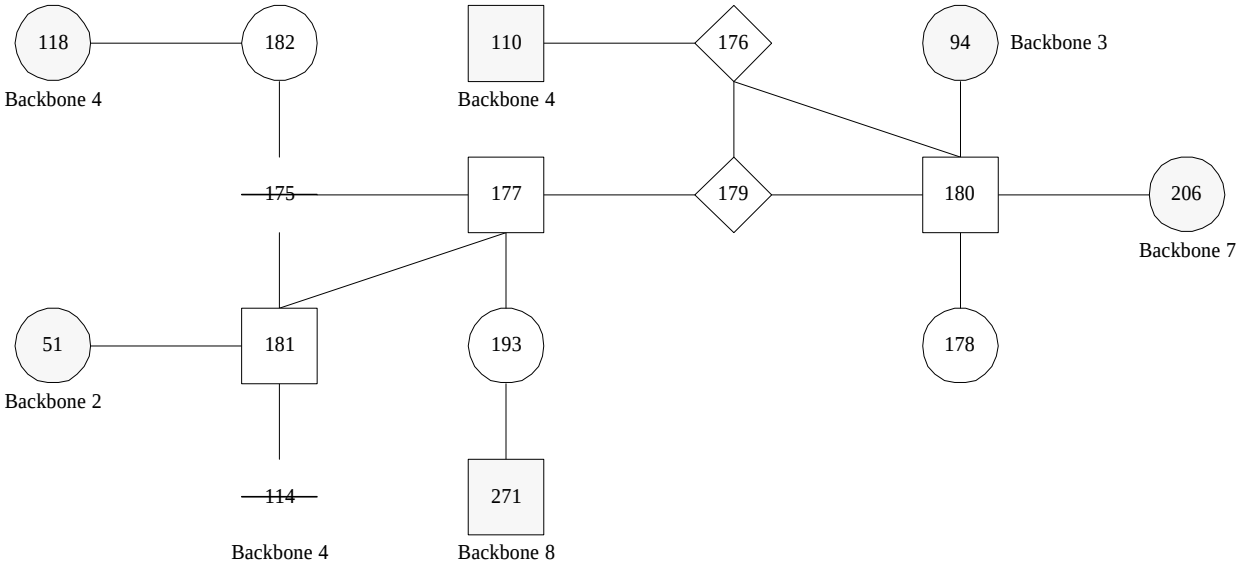
Backbone 4:



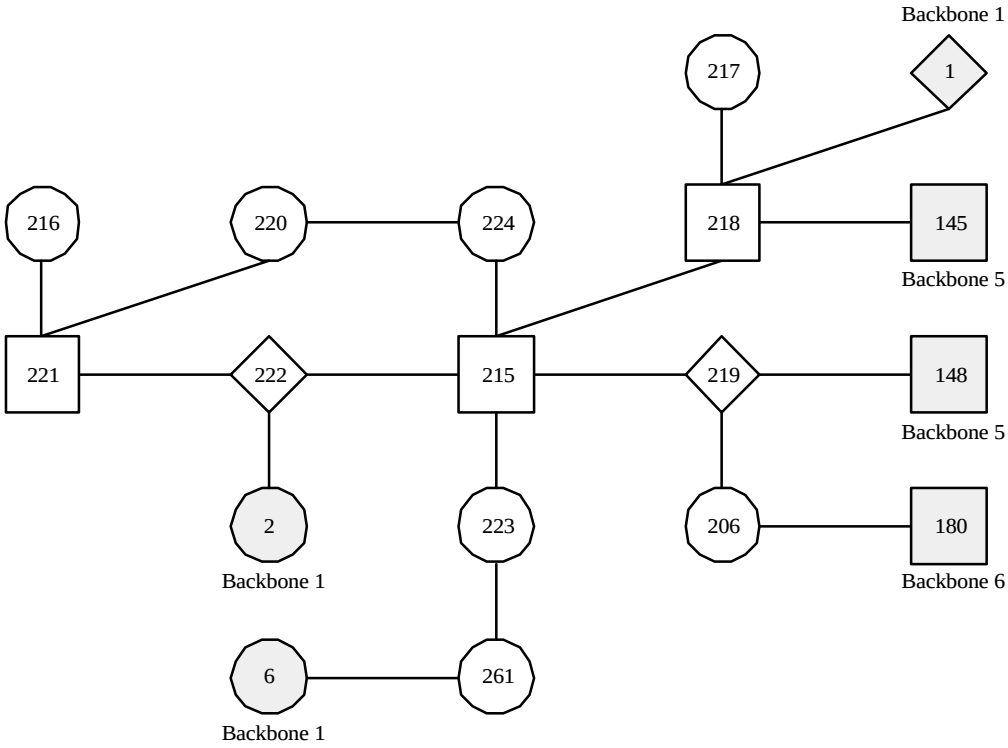
Backbone 5:



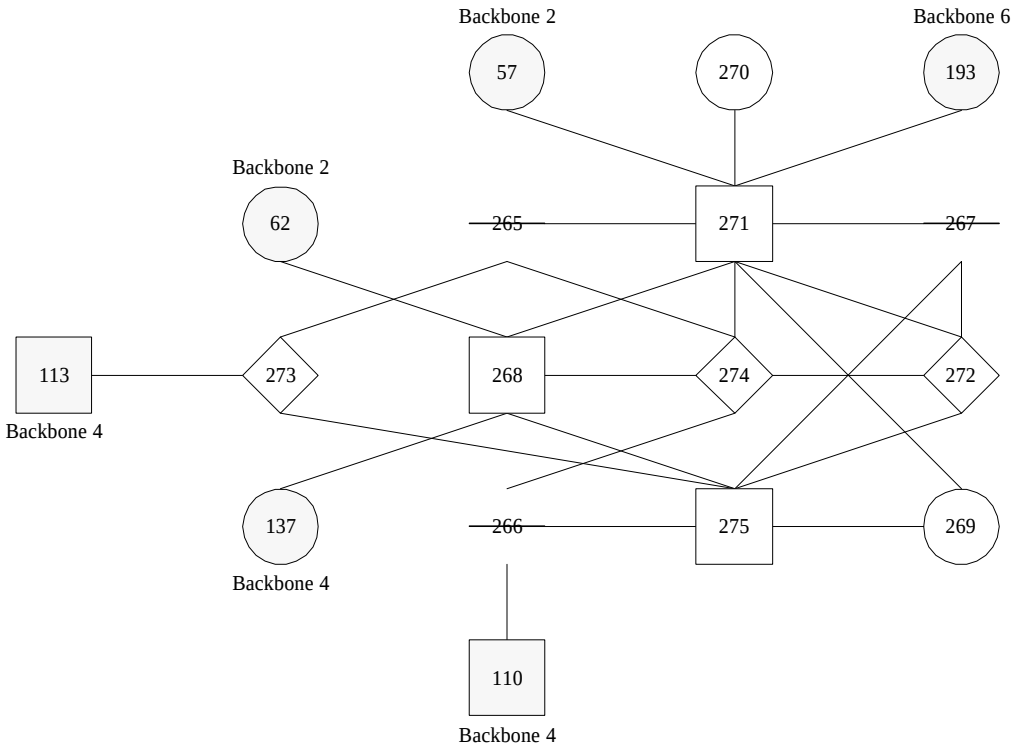
Backbone 6:



Backbone 7:



Backbone 8:



The stub nodes (Local ISP) are connected to the backbone nodes as follows:

Backbone #	Backbone node	Local ISP node connected	Backbone node connected
1	0	7, 8, 9, 10, 211	1, 4, 5
	1	11, 12, 13, 14, ***15***, 16, 90	4, 6, **218**
	2	17, 302	3, **222**
	3	18, 19, 20, 89	2, 5
	4	21, 22, 23	0, 1
	5	24, 25, 26	0, 3
	6	27, 28, 29, 30, 31, 32, 33, 34	1, **261**
2	35	44, 45, 166	36, 40, 42
	36	46, 47, 48, 49, 50, 115	35, 39, 41
	37	52, 53, 54, ***55***, 171	40, 42, 51
	38	56, 58, 59, 60, 61	40, 57
	39	63, 64	36, 43, 62
	40	65, 66, 67, 68	35, 37, 38
	41	69, 70, 71, 72	36, 42
	42	73, 74, 75	35, 37, 41
	43	76, 77, 78, 79	39, **114**
	51		37, **181**
	57		38, **271**
	62		39, **268**
3	80	86, 87	82
	81	88, 89, 90, 91, 92, 93, ***95***	84, 94
	82	96, 97, 98, 99, 100	80, 84, **113, 147, 148**
	83	101, 102, 103, 104, 259, 290	85
	84	105, 126	81, 82, 85, **114**
	85	106, 107, 108, 109, 314	83, 84
	94		**180**
4	110	117	111, 118, **176, 266**
	111	119, 120, 121, 122, 123	110, 112, 115
	112	124, 125, 126, 127, 128, 129	111, 116
	113	130, 131	**82**, 116, **273**
	114	132, 133	**43, 84**, 116, **181**
	115	134, ***135***, 136, 138, 139, 140	111, 137
	116	141, 142, 143, 144	112, 113, 114
	118		110, **182**
	137		115, **268**

5	145	151	148, 149, **218**
	146	152, 153, 154, 155, 156, 277	148, 149, 150
	147	157, 158, 159, 160	**82**, 148
	148	161, 162, 163	**82**, 145, 146, 147, **219**
	149	164, 165, 168, 184	145, 146
	150	166, 167, 168, 169, 170, 171, 172, 173, 174	146
6	***175***	183, 184, 185	177, 181, 182
	176	186, 312	**110**, 179, 180
	177	187, 188, 189, 190, 191, 192, 194, 195	175, 179, 181, 193
	178	196, 197, 198, 199	180
	179	200, 201, 202, 203	176, 177, 180
	180	204, 205, 207, 208, 209, 210	**94**, 176, 178, 179, **206**
	181	211, 212	**51, 114**, 175, 177
	182	213, 214	**118**, 175
	193		177, **271**
7	206		**180**, 219
	215	225, 226,227, 228, 229	218, 219, 222, 223, 224
	216	230, 231, 232	221, 223
	217	233, 234, 235, 236, 237, 238, 239, 280	218
	218	240, 241	**1**, **145**, 215, 217
	219	242, 243, 244, 245	**148,** 206, 215
	220	246, 247, 248, 249, 250	221, 224
	221	251, 252	216, 220, 222
	222	253, 254, ***255***, 256, 257, 258, 259	**2**, 215, 221
	223	260, 261	215, 216, 261
	224	262, 263, 264	215, 220
	261		**6**, 223
8	265	276, 277, 278, 279	271, 273, 274
	266	276, 280, 281, 282, 283, 284, 285, 286	**110**, 274, 275
	267	287	271, 272, 275
	268	288, 289, 290, 291, 292, 293, 294	**62, 137**, 271, 274, 275
	269	***295***, 296	271, 275
	270	297, 298	271
	271	299, 300, 301, 302, 303	**57, 193**, 265, 267, 268, 269, 270, 272, 274

	272	304, 305, 306	267, 271, 274, 275
	273	307, 308, 309, 310, 311, 312	**113**, 265, 275
	274	313, 314	265, 266, 268, 271, 272
	275	315, 316, 317, 318, 319	266, 267, 268, 269, 272, 273

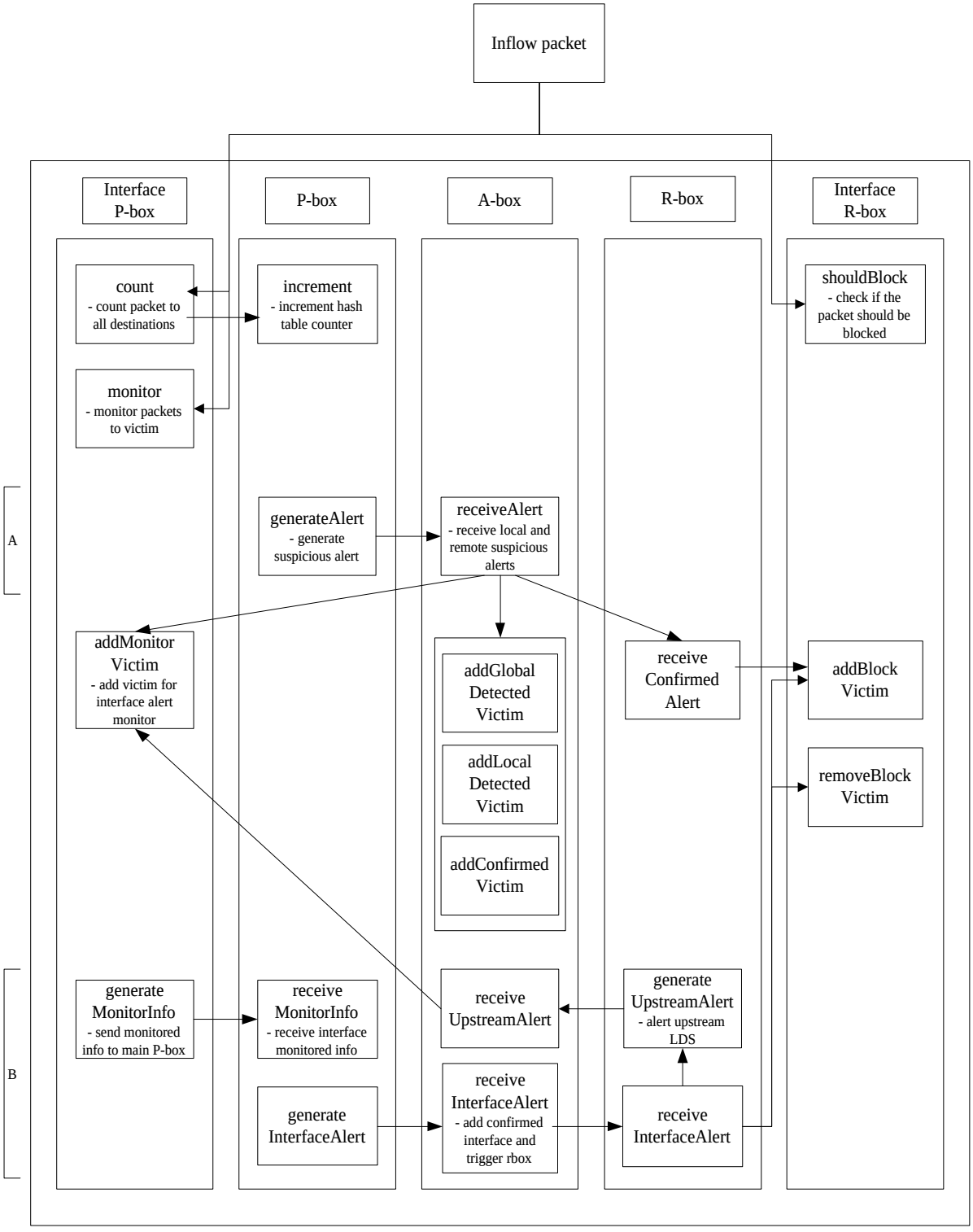
Remark:

nn – Backbone node in foreign domain

nn – Popular node, i.e. node with higher volume of traffic

<u>nn</u> – Victim node

Appendix E. Design Diagram of FLDS Simulation Module



A: Trigger on every normal sample period (2 sec)
B: Trigger on every response period (0.2sec)

Appendix F. Detail Simulation Results

Survival Ratio (without Global Defense System)

Daemon coverage	#normal packet arrived before sec10	#normal packet dropped before sec10	#normal packet arrived before sec20	#normal packet dropped before sec20	#normal packets arrived	#normal packets dropped	#normal packets process	Survival ratio	#total attack packets arrived	#total attack packets dropped
Victim = Node 55										
0.0%	5856	0	14538	0	8682	0	8682	101.70%	0	0
2.5%	6003	1011	14493	6948	8490	5937	2553	29.91%	40,014	25,133
5.0%	6232	2289	15118	10057	8886	7768	1118	13.10%	109,588	93,724
7.5%	6165	2224	14103	9043	7938	6819	1119	13.11%	92,879	76,412
10.0%	5672	2069	13225	8771	7553	6702	851	9.97%	118,241	101,924
12.5%	5983	2563	13557	9387	7574	6824	750	8.79%	139,077	122,433
15.0%	5414	2449	13206	9611	7792	7162	630	7.38%	157,381	140,146
17.5%	5733	2266	13370	9340	7637	7074	563	6.60%	171,383	154,317
20.0%	5269	2207	12205	8727	6936	6520	416	4.87%	232,481	215,176
25.0%	5343	2083	12585	8951	7242	6868	374	4.38%	245,792	229,263
Victim = Node 95										
0.0%	6587	0	15860	0	9273	0	9273	101.52%	0	0
2.5%	5885	536	14466	5270	8581	4734	3847	42.12%	26,113	12,689
5.0%	5797	1918	14720	9565	8923	7647	1276	13.97%	90,214	74,000
7.5%	6643	2162	15250	9750	8607	7588	1019	11.16%	113,175	96,828
10.0%	6002	2114	14529	9756	8527	7642	885	9.69%	134,287	118,017
12.5%	5911	2049	14873	10394	8962	8345	617	6.75%	189,333	172,828
15.0%	6312	2357	14827	10254	8515	7897	618	6.77%	176,164	159,449
17.5%	6092	2935	14243	10744	8151	7809	342	3.74%	286,212	269,036
20.0%	6454	2531	15542	11163	9088	8632	456	4.99%	251,819	234,751
25.0%	5669	2438	13667	10168	7998	7730	268	2.93%	335,315	317,783
Victim = Node 255										
0.0%	6527	0	15861	0	9334	0	9,334	100.18%	0	0
2.5%	5337	802	13845	4897	8508	4095	4,413	47.37%	23,996	8,954
5.0%	6378	790	13857	6230	7479	5440	2,039	21.88%	41,961	27,258
7.5%	5134	958	13567	8129	8433	7171	1,262	13.55%	79,158	63,972
10.0%	6233	1984	14068	8632	7835	6648	1,187	12.74%	86,160	70,088
12.5%	5840	2484	13218	9201	7378	6717	661	7.09%	147,682	130,543
15.0%	5853	2503	13842	9843	7989	7340	649	6.97%	166,848	149,582
17.5%	5795	2364	13483	9570	7688	7206	482	5.17%	187,510	170,573
20.0%	5606	1925	12243	8123	6637	6198	439	4.71%	188,336	171,476
25.0%	6022	2917	13715	10217	7693	7300	393	4.22%	254,851	237,354
Average										
0.0%							9096	101.14%	0	0
2.5%							3604	39.80%	30,041	15,592
5.0%							1478	16.32%	80,588	64,994
7.5%							1133	12.60%	95,071	79,071

10.0%							974	10.80%	112,896	96,676
12.5%							676	7.55%	158,697	141,935
15.0%							632	7.04%	166,798	149,726
17.5%							462	5.17%	215,035	197,975
20.0%							437	4.86%	224,212	207,134
25.0%							345	3.84%	278,653	261,467

Survival Ratio (with Global Defense System)

Daemon coverage	#normal packet arrived before sec10	#normal packet dropped before sec10	#normal packet arrived before sec20	#normal packet dropped before sec20	#normal packets arrived	#normal packets dropped	#normal packets processed	Survival ratio	#total attack packets arrived	#total attack packets dropped
Victim = Node 55										
0.0%	6,508	0	14,899	0	8,391	0	8,391	98.30%	0	0
2.5%	4,638	93	8,147	158	3,509	65	3,444	40.34%	7,126	346
5.0%	4,790	363	7,750	642	2,960	279	2,681	31.41%	12,828	1,632
7.5%	4,024	105	6,055	352	2,031	247	1,784	20.90%	14,212	1,585
10.0%	3,959	72	5,586	127	1,627	55	1,572	18.42%	10,202	500
12.5%	5,667	1,434	10,766	4,986	5,099	3,552	1,547	18.12%	45,366	29,937
15.0%	4,482	1,398	6,174	2,299	1,692	901	791	9.27%	38,998	22,504
17.5%	3,592	712	4,816	1,275	1,224	563	661	7.74%	32,090	15,725
20.0%	3,592	538	4,611	1,031	1,019	493	526	6.16%	32,755	16,222
25.0%	3,387	568	4,596	1,212	1,209	644	565	6.62%	37,210	20,563
Victim = Node 95										
0.0%	5,783	0	14,778	0	8,995	0	8,995	98.48%		
2.5%	6,143	268	11,638	1,858	5,495	1,590	3,905	42.75%	17,848	4,902
5.0%	4,635	1,214	7,548	2,584	2,913	1,370	1,543	16.89%	32,302	15,687
7.5%	5,763	2,012	9,416	4,111	3,653	2,099	1,554	17.01%	38,062	21,246
10.0%	4,306	821	7,053	2,394	2,747	1,573	1,174	12.85%	37,841	21,613
12.5%	3,887	1,077	6,322	2,783	2,435	1,706	729	7.98%	49,799	32,706
15.0%	5,198	1,618	7,460	3,054	2,262	1,436	826	9.04%	49,784	32,528
17.5%	5,728	2,070	7,621	3,169	1,893	1,099	794	8.69%	48,582	31,768
20.0%	5,307	1,844	7,080	3,174	1,773	1,330	443	4.85%	69,338	52,142
25.0%	4,614	1,322	6,595	2,945	1,981	1,623	358	3.92%	90,468	72,835
Victim = Node 255										
0.0%	5,955	0	15,255	0	9,300	0	9,300	99.82%		
2.5%	5,588	1,205	11,150	3,055	5,562	1,850	3,712	39.84%	21,374	7,472
5.0%	4,948	900	9,333	1,817	4,385	917	3,468	37.22%	21,829	6,642
7.5%	4,970	1,294	7,837	2,064	2,867	770	2,097	22.51%	24,264	9,026
10.0%	4,866	1,163	7,438	2,426	2,572	1,263	1,309	14.05%	30,577	14,153
12.5%	4,676	1,199	7,230	2,602	2,554	1,403	1,151	12.35%	35,407	19,183
15.0%	4,090	1,081	6,406	2,612	2,316	1,531	785	8.43%	44,176	26,745
17.5%	4,751	1,421	6,654	2,680	1,903	1,259	644	6.91%	46,329	29,546
20.0%	3,548	1,011	5,043	2,056	1,495	1,045	450	4.83%	55,544	37,735
25.0%	4,281	1,438	5,483	2,327	1,202	889	313	3.36%	64,913	47,170
Average										

0.0%							8,895	98.86%	0	0
2.5%							3,687	40.98%	15,449	4,240
5.0%							2,564	28.51%	22,320	7,987
7.5%							1,812	20.14%	25,513	10,619
10.0%							1,352	15.11%	26,207	12,089
12.5%							1,142	12.82%	43,524	27,275
15.0%							801	8.91%	44,319	27,259
17.5%							700	7.78%	42,334	25,680
20.0%							473	5.28%	52,546	35,366
25.0%							412	4.63%	64,197	46,856

Packet-level False Positive & False Negative Ratios

Daemon coverage	#true positive	#false positive	#false negative	#normal packet	#attack packet	False positive ratio	False negative ratio
Victim = Node 55							
0.0%	0	0	0	1,560,589	0	0.00%	0.00%
2.5%	27,412	7,307	68,231	1,568,856	95,643	0.47%	71.34%
5.0%	50,223	8,539	121,047	1,559,177	171,270	0.55%	70.68%
7.5%	88,170	8,196	165,481	1,554,809	253,651	0.53%	65.24%
10.0%	121,231	10,189	154,480	1,546,069	275,711	0.66%	56.03%
12.5%	91,308	7,273	254,117	1,538,208	345,425	0.47%	73.57%
15.0%	148,653	12,999	302,975	1,528,704	451,628	0.85%	67.09%
17.5%	252,503	11,134	261,146	1,564,325	513,649	0.71%	50.84%
20.0%	292,792	10,140	344,806	1,532,025	637,598	0.66%	54.08%
25.0%	295,983	9,694	352,074	1,530,011	648,057	0.63%	54.33%
Victim = Node 95							
0.0%	0	0	0	1,588,725	0	0.00%	0.00%
2.5%	14,867	3,817	91,705	1,576,802	106,572	0.24%	86.05%
5.0%	63,174	7,838	135,692	1,570,139	198,866	0.50%	68.23%
7.5%	69,980	5,793	137,578	1,580,989	207,558	0.37%	66.28%
10.0%	170,151	8,407	260,695	1,537,196	430,846	0.55%	60.51%
12.5%	169,091	8,920	230,819	1,546,861	399,910	0.58%	57.72%
15.0%	180,034	10,566	271,445	1,557,490	451,479	0.68%	60.12%
17.5%	192,319	8,687	311,767	1,551,167	504,086	0.56%	61.85%
20.0%	266,571	11,251	387,356	1,545,831	653,927	0.73%	59.24%
25.0%	291,391	8,618	449,845	1,560,935	741,236	0.55%	60.69%
Victim = Node 255							
0.0%	0	0	0	1,575,507	0	0.00%	0.00%
2.5%	15,542	3,824	103,205	1,574,247	118,747	0.24%	86.91%
5.0%	75,344	12,510	164,958	1,528,676	240,302	0.82%	68.65%
7.5%	95,122	10,146	170,655	1,585,401	265,777	0.64%	64.21%
10.0%	81,745	7,565	200,686	1,551,745	282,431	0.49%	71.06%
12.5%	140,152	7,925	349,696	1,544,171	489,848	0.51%	71.39%

15.0%	189,266	8,843	273,976	1,556,588	463,242	0.57%	59.14%
17.5%	218,095	8,971	368,951	1,510,365	587,046	0.59%	62.85%
20.0%	266,473	13,740	352,757	1,503,973	619,230	0.91%	56.97%
25.0%	293,813	9,458	409,018	1,539,163	702,831	0.61%	58.20%
Average							
0.0%	0	0	0	1,574,940	0	0.00%	0.00%
2.5%	19,274	4,983	87,714	1,573,302	106,987	0.32%	81.43%
5.0%	62,914	9,629	140,566	1,552,664	203,479	0.62%	69.19%
7.5%	84,424	8,045	157,905	1,573,733	242,329	0.51%	65.24%
10.0%	124,376	8,720	205,287	1,545,003	329,663	0.56%	62.53%
12.5%	133,517	8,039	278,211	1,543,080	411,728	0.52%	67.56%
15.0%	172,651	10,803	282,799	1,547,594	455,450	0.70%	62.12%
17.5%	220,972	9,597	313,955	1,541,952	534,927	0.62%	58.51%
20.0%	275,279	11,710	361,640	1,527,276	636,918	0.77%	56.76%
25.0%	293,729	9,257	403,646	1,543,370	697,375	0.60%	57.74%