University of Colorado at Colorado Springs

On the Proxy Server Based Multipath Connections (PSMC)

Yu Cai

A dissertation proposal submitted

in partial fulfillment of the requirement for the degree

Doctor of Philosophy

Thesis Committee:

Dr. Terry Boult

Dr. Edward. Chow

Dr. Charlie Shub

Dr. Xiaobo Zhou

Dr. Rodger Ziemer

Table of Contents

Abstract

Multipath connections provide potential multiple paths between network nodes. The traffic from a source can be spread over multiple paths and transmitted in parallel through the network. Multipath connections offer applications with the ability to improve network security, reliability and performance.

In this dissertation, we propose to study proxy server based multipath connections (PSMC). The key idea of PSMC is as follows. 1) By using a set of connection relay proxy servers, we could set up indirect routes via the proxy servers, and transport packets over the network through the indirect routes. 2) By enhancing existing TCP/IP protocols, we could efficiently distribute and reassemble packets among multiple paths at two end nodes, and increase end-to-end TCP throughput.

PSMC utilizes existing network protocols and infrastructure with several protocol enhancements. This ensures the compatibility with current Internet. PSMC can be more conveniently and adaptively deployed in various network environments. Therefore, a large number of applications could benefit from utilizing PSMC.

We plan to systematically study PSMC as follows. 1) We plan to develop proxy server(s) selection / placement algorithms for PSMC. 2) We plan to design and implement protocols to distribute, transport and reassemble packets for PSMC. 3) We plan to develop several PSMC applications, like Secure Collective Defense (SCOLD) network, providing additional bandwidth based on operational requirements, and other applications. 4) Security. We plan to investigate security issues related to PSMC, like the potential attacks, the misuse, and the collective defense mechanisms.

The research result and insight obtained from PSMC could have broader impact on the protocols and security in today's Internet.

Chapter 1

Introduction

1.1 Overview

The key challenges in today's Internet are to improve network security, reliability and performance for the clients. The network connection between two network nodes is mostly over a single path connection. The single path connection itself is vulnerable to potential attacks, link breakage or even traffic congestion. The single path connection model may also under-utilize network resources and suffer from performance problems. Therefore it may not always provide a good and reliable network connection.

Multipath connections provide potential multiple paths between network nodes. The traffic from a source can be spread over multiple paths and transmitted in parallel through the network. Multipath connections can utilize the network resources more efficiently; thereby increasing the network security, reliability and performance. By utilizing the aggregate bandwidth of network nodes, it can provide better quality-of-service, and the ability to cope up with network congestion, link breakage and potential attacks.

The IBM Systems Network Architecture (SNA) network in 1974 [66] is probably the first wide area network which provides multiple paths connections between nodes. N. F. Maxemchuk studied how to disperse the traffic over multiple paths in 1975. He called it dispersity routing in his paper [12]. Since then, multipath connections have been studied in various settings. One example of multipath connections is link aggregation [3], which is a data link layer protocol. In IP layer, multipath connections have been studied extensively in the name of mulitpath routing. Various table-driving multipath routing algorithms (link state or distance vector) [2, 8, 15, 16, 17, 18, 19, 20, 23, 24, 26] and source routing algorithms [5, 6] were proposed. Linux has multipath connections implemented in TCP layer [9, 64], which is suited for multi-homing users. For the detailed taxonomy on multipath connections, please see Chapter 2.

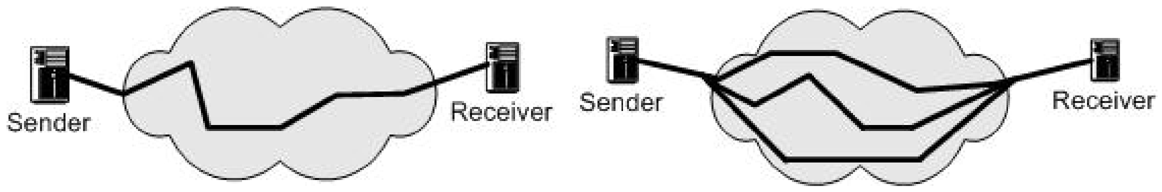


Figure 1.1: Single path connection vs. multipath connections

In this dissertation, we propose to study proxy server based multipath connections (PSMC).

The key idea of PSMC is as follows.

a) By using a set of connection relay proxy servers, we could set up indirect routes via the proxy servers, and transport packets over the network through the indirect routes.

b) By enhancing existing TCP/IP protocols, we could efficiently distribute and reassemble packets among multiple paths at two end nodes, and increase end-to-end TCP throughput.

This approach offers applications with the ability to improve network security, reliability and performance.

Figure 1.2 is the diagram that illustrates the proxy server based multipath connections (PSMC). There are three basic components in a PSMC network. The **multipath sender**, or distributor, is responsible for efficiently and adaptively distributing packets over the selected multiple paths. Some of the packets will go through the normal direct route, other packets will go through the alternate indirect routes via the proxy servers. The intermediate connection relay **proxy servers**, or forwarders, examine the incoming packets and forward them to the destinations through the selected path. The **multipath receiver**, or collector, collects the packets arrived from multiple paths, reassembles them in order and delivers them to the user.

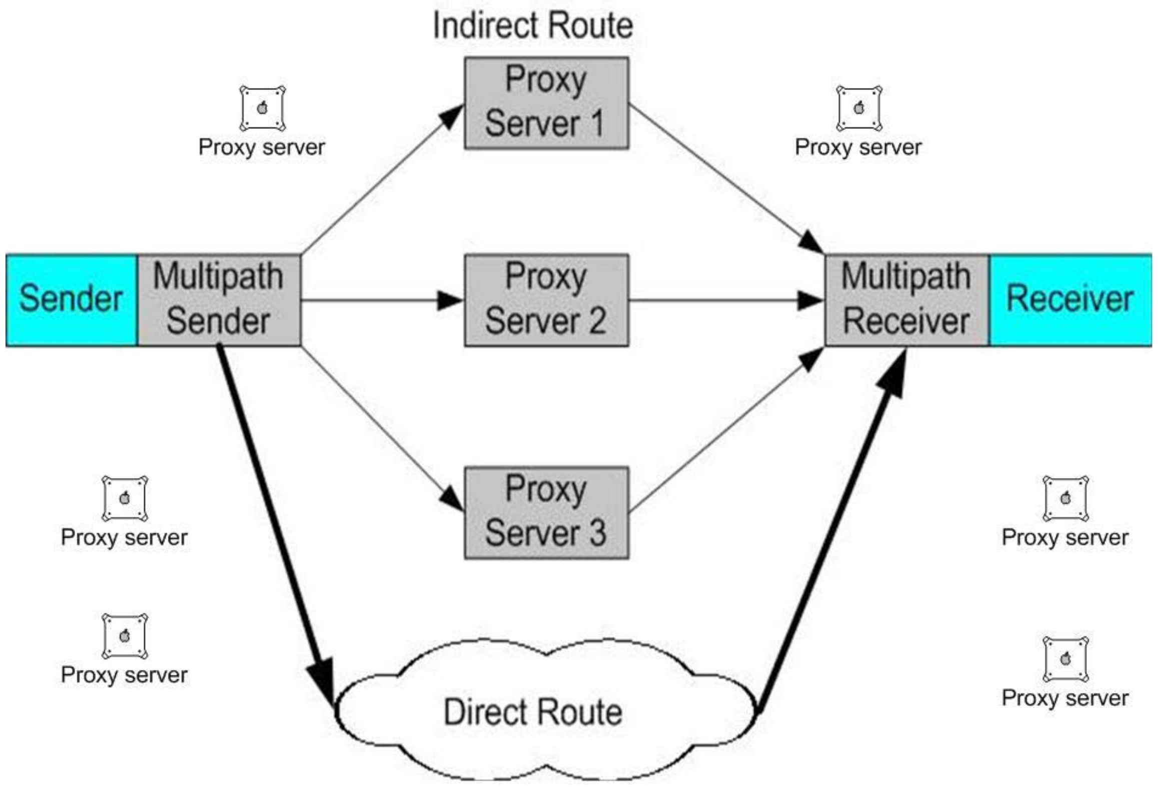


Figure 1.2: Proxy server based multipath connections (PSMC)

For convenience, from now on, we refer to our approach of "proxy server based multipath connections" as "PSMC". And we use the phase "direct route" to refer to the network route whereby a packet normally takes when it travels through the network. The phase "indirect route" is used to refer to the network route which utilizes the connection relay proxy server. The phase "proxy server" is used specifically for the connection relay proxy servers in a PSMC network, unless otherwise specified.

Compared with other types of multipath connections approaches, PSMC has general benefits like other multipath connections approaches, plus the following unique advantages:

a) **Compatibility**: PSMC utilizes and enhances existing TCP/IP protocols and network infrastructure to distribute, transport and reassemble packets. Unlike some multipath connection approaches (ie. link aggregation, multipath routing), which require changes on physical network infrastructure, PSMC only requires some feasible changes on network software and protocols. This ensures the compatibility with current Internet.

b) **Flexibility**: PSMC can be more conveniently and adaptively deployed in various network environments. PSMC gives the end users more control and flexibility on how to set up multipath connections.

c) **Usability**: A large number of applications in various categories could benefit from utilizing PSMC. For example, secure collective defense network (SCOLD), which provides alternate routes for DDoS attack intrusion tolerance. PSMC can also be utilized to provide additional bandwidth based on operational requirements in an enterprise network. PSMC can be utilized to provide QoS for various applications, like video streaming. See Chapter 4 for details.

1.2 Thesis outline

We propose to systematically study the PSMC technique as follows.

1) Algorithms for PSMC.

To set up multipath connections in PSMC, the first step is to select the desired proxy servers. Different proxy server selections may result in significantly different performance [10].

We have developed heuristic algorithms to choose the best mirror sites for parallel download from multiple mirror sites [61]. The preliminary performance results on simulated network and real network are very promising.

We plan to develop algorithms to solve the following problems in PSMC.

a) **Server Selection Problem**. Given the target server location and a set of proxy servers, choose the best proxy server(s) for a client or for a group of clients, to achieve best performance, in terms of aggregate bandwidth.

When there are hundreds of clients and proxy servers, the selected paths might interfere with each other, especially when the paths have join nodes. Therefore, the server selection algorithm needs to find the disjoint paths to achieve the best performance.

However, finding the disjoint paths is not an easy task. One of the solutions is **Server Partition.** Assuming we have enough proxy servers, we can partition the clients and the proxy servers into several zones. A client will only use proxy servers assigned to its partition. Better partition algorithms with consideration of network distance and network location need to be studied.

b) **Server Placement Problem**. Given the target server location and a set of nodes, choose the best node(s) to place the proxy servers, for certain connection requirements, like maximize the aggregate network bandwidth.

The above problems are likely to be NP problems, therefore a heuristic algorithm might be a good solution. The other research direction is to loosen the optimal constrains and simplify the problem to make the problem solvable in P-time.

2) Protocols for PSMC

To enjoy the benefit of PSMC, protocols need to be designed to distribute, transport and reassemble packets for PSMC.

We plan to design and implement a thin layer between TCP and IP [59], to deal with packets distribution and reassembling. We will modify the Linux kernel to support it. The benefits of putting a thin layer between TCP and IP are as follows:

a) We can utilize existing TCP/IP protocols, particularly the packets re-sequencing mechanism.

b) We can hide the complexity of multipath connections from upper layer users.

c) We can still maintain the high end-to-end TCP throughput.

We plan to implement various scheduling algorithms for packets distribution and reassembling, like round robin scheduling and least usage scheduling [62].

For the packets transmission, after investigating various approaches, like SOCKS proxy server [54], Zebedee [53], we proposed to use IP Tunnel[55] or IPSec[58] to enable indirect routes via proxy servers. IP Tunnel is a widely used protocol and available in various operating systems. We will utilize and enhance existing tunneling protocols for PSMC. In particular, we will study issues like tunnel setup, host authentication, security mechanism, fail-over mechanism and sticky connection.

3) PSMC prototype and applications

As a feasibility study of PSMC, we proposed a Secure Collective Defense (SCOLD) [60] network which is used for DDoS intrusion tolerance. SCOLD tolerates the DDoS attacks through indirect routes via proxy servers, and improves network performance by spreading packets through multiple indirect routes. SCOLD incorporates various cyber security techniques, like secure DNS update [60], Autonomous Anti-DDoS network [39], IDIP protocol [41]. We have finished the prototype of SCOLD system. We plan to enhance SCOLD for better scalability, reliability, performance and security.

PSMC can be utilized to provide additional bandwidth based on operational requirements in an enterprise network. PSMC can also be utilized to provide QoS for various applications, like video streaming. Other applications include content switch cluster server load balancing, wireless ad hoc network, and parallel download from multiple mirror sites.

We will evaluate the overhead of multipath connections, including tunneling overhead, handshake overhead, and distribution/reassembling overhead. We will evaluate the performance of multipath connections in terms of bandwidth, response time and throughput.

We also plan to systematically compare PSMC with other multipath connections approaches, like source routing, or Linux multipath connections.

4) Security issues related to PSMC.

We plan to investigate security issues raised by misuse of PSMC, like how to control “aggressive” users, and how to prevent PSMC from being used to launch DDoS attacks. We will study the potential attacks against PSMC, like potential “Tunneling to death”(similar to ping to death). We need to study how to detect and handle the compromised proxy servers.

Last, we will study the collective defense mechanism to tie different organizations with better cooperation and collaboration.

The rest of the proposal is organized as follows. Chapter 2 presents related work of multipath connections. Chapter 3 presents several PSMC related Algorithms. Chapter 4 presents protocols to support PSMC. Chapter 5 presents several PSMC applications, including SCOLD, wireless ad hoc network, providing additional bandwidth upon request, QoS for video streaming, content switch with cluster servers, and parallel download from multiple mirror sites. Chapter 6 studies security issues of PSMC. Chapter 7 is the conclusion.

Chapter 2

Related Work

This chapter surveys the related work of PSMC.

The technique of multipath connections appears under many different labels, like multiple path routing, alternate path routing and traffic dispersion. And often the same label is used in literature to refer to different things. We try to survey and clarify the different concepts of multipath connections in this chapter.

The rest of the chapter is organized as follows. Section 1 provides the taxonomy of multipath connections. Section 2 introduces a related problem of parallel download from multiple mirror sites. Section 3 presents related work on server selection algorithm for PSMC. Section 4 presents related work on TCP/IP protocols in PSMC.

2.1 Multipath connections

The IBM Systems Network Architecture (SNA) network in 1974 [66] is probably the first wide area network which provides multiple paths connections between nodes. However, in the SNA network, only one path is used at a time, and the purpose of multiple paths is to provide fault-tolerance mechanisms. Also, SNA multiple paths are predefined and pre-computed.

N. F. Maxemchuk [12] in 1975 used channel sharing to provide multipath connections and reduce queuing delay in store and forward network. He called the technique "dispersity routing". The research was extended to virtual circuit networks [13] and ATM network [27] to deal with busty traffic data, where both redundant and non-redundant dispersity routing techniques were described. A literature survey on traffic dispersion was presented in [28]. The author illustrated various strategies, such as packet-by-packet or string mode, to give dispersion in different network configurations.

According to the Open System Interconnection (OSI) Network Reference Model [67], we try to differentiate multipath connections between physical layer, data link layer, network layer, transport layer and application layer. This is only a rough classification. Some approaches might be multiple layers implementation. Figure 2.1 is a diagram illustrated the classification of multipath connections.

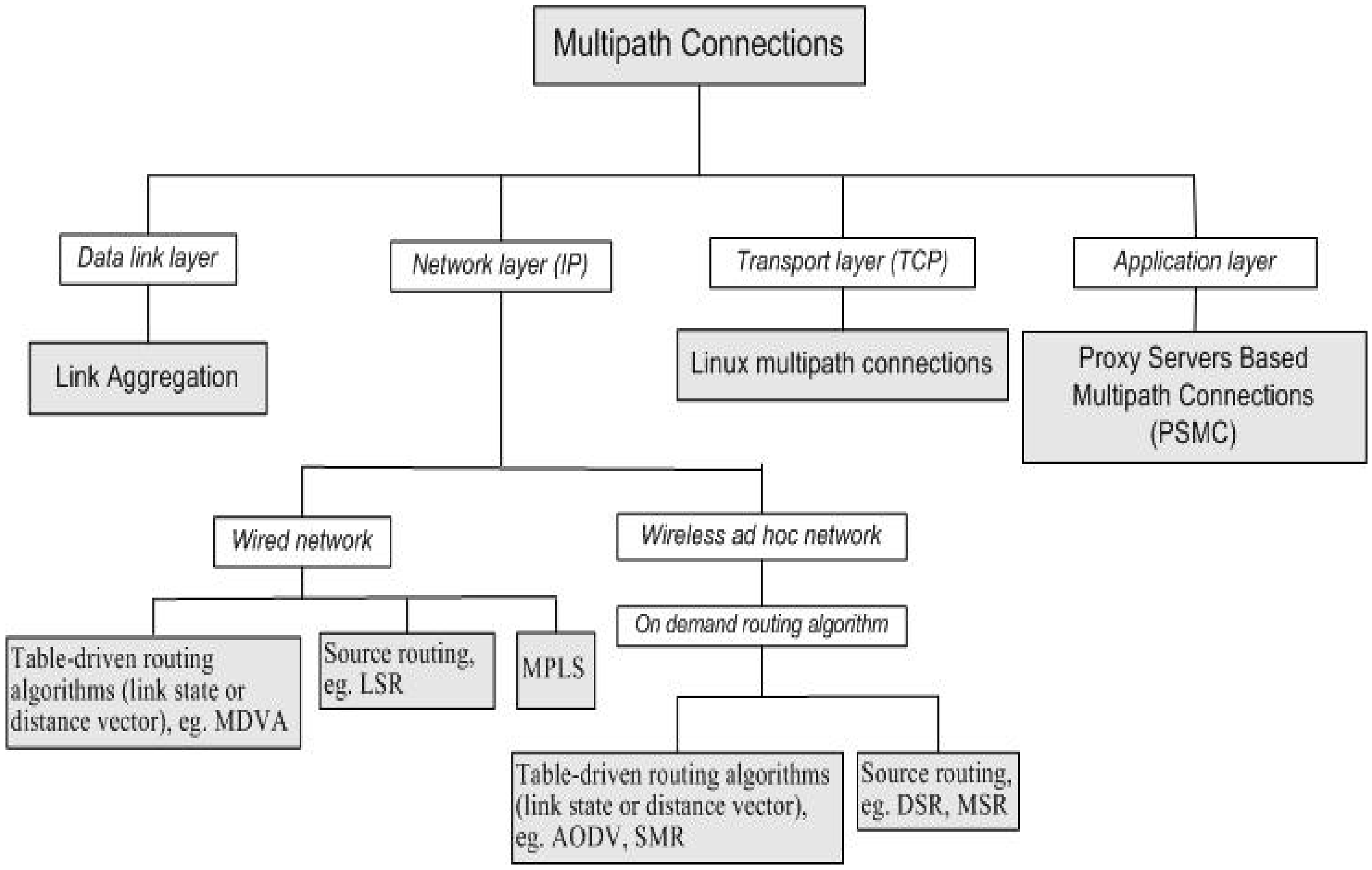

Multipath Connections
Data link layer
Network layer (IP)
Transport layer (TCP)
Application layer
Link Aggregation
Linux multipath connections
Proxy Servers Based Multipath Connections (PSMC)
Wired network
Wireless ad hoc network
Table-driven routing algorithms (link state or distance vector), eg. MDVA
Source routing, eg. LSR
MPLS
On demand routing algorithm
Table-driven routing algorithms (link state or distance vector), eg. AODV, SMR
Source routing, eg. DSR, MSR

Figure 2.1: diagram of multipath connections.

2.1.1 Physical layer

Multipath connections in physical layer are not always something that we want. For example, sometimes FM radio sounds staticy and bad because of "*multipath interference*"

[68]. Multipath interference happens when FM signals bounce around between the buildings in a city, or other large obstructions, this bounce causes a reflection and your FM radio tries to lock onto the original signal as well as the reflection! Other usages of multipath connections in physical layer, like antenna array [71], are beyond the scope of this proposal.

2.1.2 Data link layer

Multipath connections in data link layer have been implemented as Link Aggregation or Trunking, defined in IEEE 802.3ad [3]. It is a method of combining multiple physical network links between two devices into a single logical link for increased bandwidth. The upper layer applications or protocols, such as a MAC client, can treat the link aggregation group as if it were a single link. Link Aggregation requires special network hardware / software support [14]. Therefore, it is only suited for high-end users.

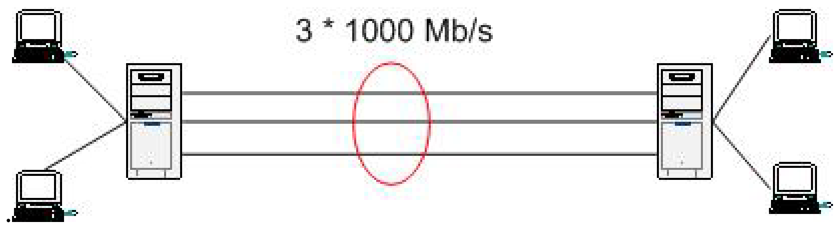


Figure 2.2: Two servers interconnected by an aggregation of three 1000 Mb/s links

2.1.3 Network layer

In network layer, Multipath connections have been studied extensively in the name of multipath routing. Various protocols have been designed for wired network and wireless ad hoc network.

a) Wired network

Based on the routing mechanism, we differentiate between Table-driven algorithms (link state or distance vector) and Source Routing.

Table-driven algorithms (link state or distance vector)

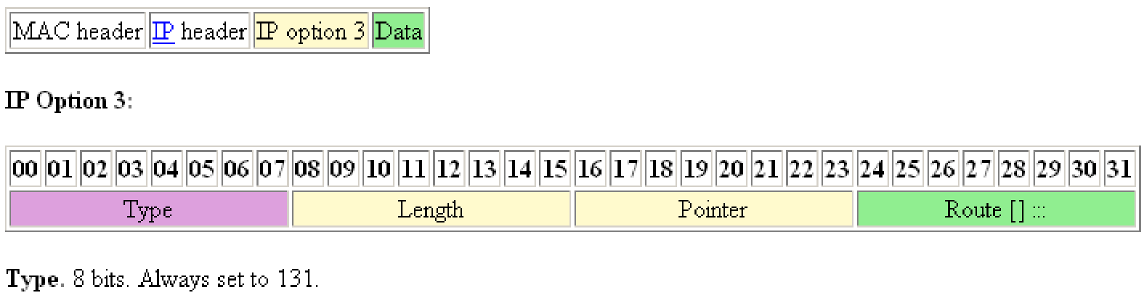
S. Vutukury et al. [2] proposed a multipath distance vector routing algorithm named MDVA. It uses a set of loop-free invariants to prevent the count-to-infinity problem. The computed multipaths are loop-free at every instant.

Johnny Chen, in his Ph.D. dissertation at Rice University [3], proposed a complete multipath network model that includes the following three components: routing algorithms that compute multiple paths, a multipath forwarding method to ensure that data travel their specified paths, and an end-host protocol that effectively use multiple paths.

Other works in similar area include [15], [16], [17], [18], [19], [20]. These protocols use table-driven algorithms (link state or distance vector) to compute multiple routes. These protocols require fundamental changes on Internet routers and routing protocols, therefore, the usage and deployment of these algorithms and protocols are limited.

Source Routing

Source Routing [4] is a technique whereby the sender of a packet can specify the route that the packet should take when the packet travels through the network. In today's Internet, when a packet travels through the network, each router will examine the "destination IP address" and choose the next hop to forward the packet to. In source routing, the sender makes some or all of these decisions. If the sender makes only some of these decisions, it is called Loose Source Routing. Source routing could be used to implement multipath routing. But, because of security concerns of source routing, most routers in today's Internet have disabled the source routing. J. Saltzer et al. [21] implemented source routing in campus-wide network environment.



MAC header	IP header	IP option 3	Data

IP Option 3:

00-07	08-15	16-23	24-31
Type	Length	Pointer	Route [] :::

Type. 8 bits. Always set to 131.

Figure 2.3: Datagram format for Loose Source Routing [22].

MultiProtocol Label Switching (MPLS)

Multiprotocol Label Switching (MPLS) [70] provides a mechanism for engineering network traffic patterns that is independent of routing tables. MPLS assigns short labels to network packets that describe how to forward them through the network. MPLS is independent of any routing protocol.

In the traditional Level 3 forwarding paradigm, as a packet travels from one router to the next, an independent forwarding decision is made at each hop. The IP network layer header is analyzed, and the next-hop is chosen based on this analysis and on the information in the routing table. In an MPLS environment, the analysis of the packet header is performed just once, when a packet enters the MPLS cloud. The packet then is assigned to a stream, which is identified by a label, which is a short (20-bit), fixed-length value at the front of the packet. Labels are used as lookup indexes into the label forwarding table. For each label, this table stores forwarding information. You can associate additional information with a label—such as class-of-service (CoS) values—that can be used to prioritize packet forwarding. MPLS could be used to set up multipath connections for traffic engineering and quality of service [69, 70].

b) Wireless ad hoc network

Multipath routing in ad hoc wireless network is a topic gaining interest, and much work has recently been done in this field. An ad hoc wireless network is a collection of wireless mobile hosts forming an instant deployable network without the aid of any base station, other infrastructure or centralized administration. The most popular routing approach in ad hoc network is on-demand routing because of its effectiveness and efficiency. Routing protocols used in wired network, which periodically exchanging route messages to maintain route table, are not well suited for ad hoc network, due to the considerable overhead produced by route update and their slow convergence to topological changes. On-demand routing protocols build routes only when a node needs to send data packets to a destination. Each node operates as a specialized router, and routes are obtained on-demand with no reliance on periodic advertisements.

Based on the routing mechanism, we differentiate between Table-driven algorithms (link state or distance vector) and Source Routing.

Table-driven algorithms (link state or distance vector)

C. Perkins et al. [7] proposed a novel algorithm for the operation of ad-hoc networks, named Ad-hoc On Demand Distance Vector Routing (AODV). The routing algorithm is quite suitable for a dynamic self-starting network, as required by users wishing to utilize ad-hoc networks.

Multipath routing protocols in ad hoc network proposed in [23], [24], [25], [26] are really backup route protocols, in the sense that even though these protocols build multiple paths on demand, but the traffic is not distributed into multiple paths. Only one route is primarily used and the secondary path is used when the primary route is broken.

S. Lee et al. [8] propose an on-demand multipath routing scheme for ad hoc wireless network, called Split Multipath Routing (SMR), that establishes and utilizes multiple routes of maximally disjoint paths. The proposed protocol uses a per-packet allocation scheme to distribute data packets into multiple paths of active sessions.

Source Routing

Dynamic Source Routing (DSR) proposed by D. Johnson et al. [5] is an enhanced source routing designed specially for wireless ad hoc network. The protocol is composed of two main mechanisms of “Route Discovery” and “Route Maintenance”, which together allow ad hoc nodes to discover and maintain routes to any destinations in the ad hoc network. This protocol allows multipath routing and allows sender to select the route(s) to use.

L. Wang et al. [6] proposed a Multipath Source Routing (MSR) protocol for ad hoc wireless networks based on Dynamic Source Routing. MSR extends DSR’s route discovery and route maintenance mechanism to deal with multipath routing. The proposed scheme distributes load balance between multiple paths based on the measurement of RTT.

2.1.4 Transport layer

Linux has its own implementation of multipath connections [9, 64]. For convenience, we refer to it as “Linux multipath connections”. It is a solution for using multiple ISP connections (multi-homing) at the same time. Linux kernel needs to be patched to support “Advance Router” and “Multiple Path Routing” options. The Linux kernel distributes packets between multiple network connections in TCP layer. The solution’s configuration is complicated, and it fails to provide fail-over mechanism in case of failure of a connection. Also, it requires the host machine to have multiple network interfaces with multiple ISP connections.

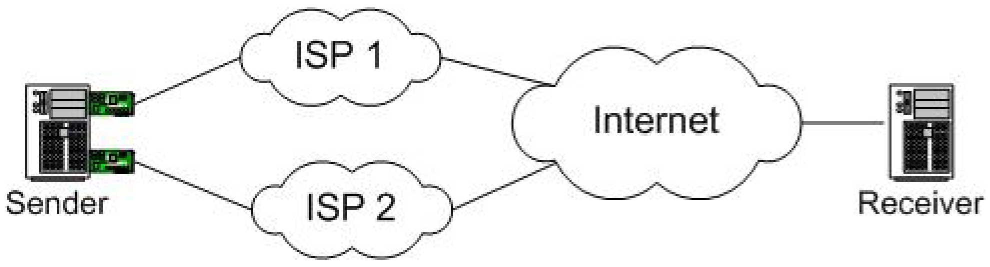


Figure 2.4: Linux multipath connections for multiple ISP connections

2.2 Parallel download from multiple mirror sites

A related problem to multipath connections is the problem of parallel download from multiple mirror sites. Rodriguez et al. [10] studied how to use the existing HTTP 1.1 byte range header protocol to retrieve documents from multiple mirror sites in parallel to reduce the download time and to improve the reliability. J. Byers [11] proposed a feedback-free protocol to access documents from multiple mirror sites in parallel. The protocol is based on erasure codes (Tornado codes). It can deliver dramatic speedups at the expense of transmitting a moderate number of additional packets into the network.

The network topology of parallel download from multiple mirror sites could be viewed as half part of the PSMC. In the topology graph of PSMC, if we draw a line through the middle of the proxy servers, the right hand side network topology is exactly that of a multiple mirror sites download problem. Therefore, the result we got from PSMC can be easily extended to the multiple mirror sites download problem, and vice verse.

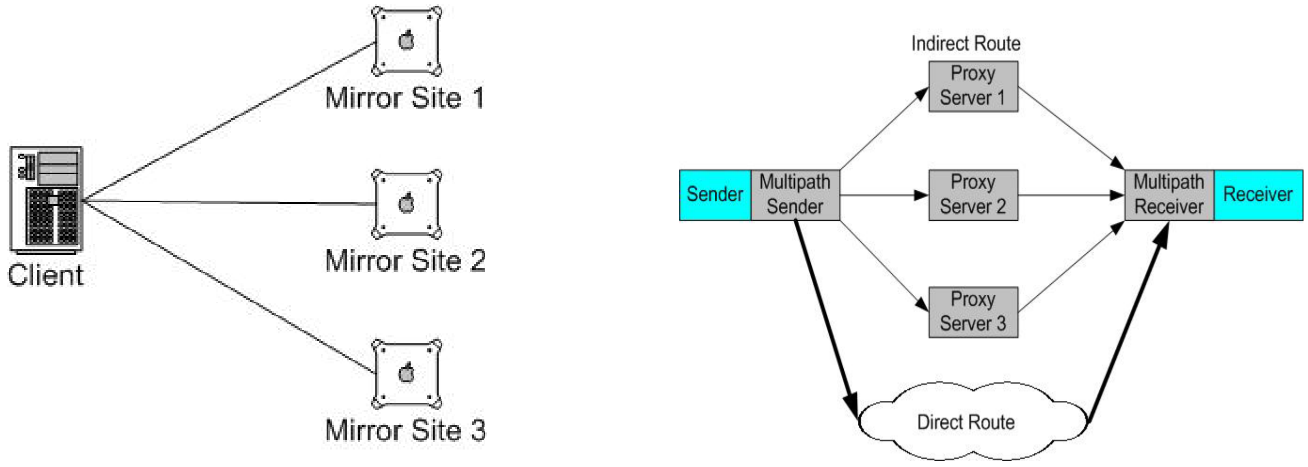


Figure 2.5: Parallel download from multiple mirror sites vs. PSMC

2.3 Algorithm for server selection problem.

Proxy server selection and placement is a critical decision in PSMC. Similar problems, like mirror server and cache server placement and selection problems, are topics gaining interests recent years [72, 73, 74, 75, 76]. Both mirror server and cache server are used to replicate web content to improve the user-perceived performance and reduce the over-all network traffic.

There are two types of mirror servers, gateway redirecting and automatic redirecting. For the gateway redirecting, there is a gateway web page with a list of available servers, and the client can choose which one to go. For automatic redirecting, the mirror server passes the client's request automatically to the real server, the real server reply back to client directly [72].

Web cache server keeps a copy of web content. When a client sends a request to the cache server, the cache server will check if it has an up-to-date local copy. If yes, it will reply to the client directly; if not, it will send the request to the real server.

According to paper [72], there are basically two types of approaches for server selection problem.

1) Formal approach.

It abstracts the network topology to a formal graphic model, and use graphic theory to study the problem. The algorithms are usually based on the following common assumptions:

a) The network topology is pre-known and static.

b) The cost associated with each path is pre-known and static.

c) The network connection between two end nodes is single path connection, and static connection.

These assumptions are reasonable for simplifying the network topology, but they are only approximation to the real Internet environment. Vern Paxson has studied extensively the end-to-end Internet dynamics [29].

K-center problem is one of the well known optimal server placement problems. For k replicas, we want to find a set of nodes K of size k that allows us to minimize the maximum distance between a node and its closet replica. K-center problem is NP-complete [73].

The existing formal algorithms include the followings.

a) Random algorithm: randomly selecting servers, without consideration of other constrains [73].

b) Greedy algorithm: selecting servers in a greedy fashion and local optimal way [73].

c) Tree-based algorithm: some authors propose solutions by further simplifying the network model from a mesh model to a tree-based model [76]. However, studies [73] show that this simplification does not always yield the optimal solution.

d) K-min algorithm: by loosing the condition to tolerate the maximum distance between a node and its closest center up to twice the distance of the maximum node-closest center distance, it can be solved in O (N|E|) time [73, 74].

e) Hot Spot algorithm: place replicas near the clients generating the greatest load [73].

2) Practical approach.

In real world situation, the network topology and connection costs information might not be pre-known or difficult to obtain. Therefore, the formal approach might not be feasible. There are several practical server selection approaches for real work situation without assumption of pre-known network information. It includes IDMap [74] and Client clustering [77]

IDMap is an architecture designed for global Internet host distance estimation service. It provides a map with Internet distance instead of geographic distance. IDMap utilize a set of Tracers to measure the distance between themselves and Address Prefixes regions of the Internet. Client of IDMap can collect the advertised traces and use them to create distance map [74].

Client Clustering is the approach to cluster the clients and place the web replicas close to the largest concentration of the clients [77].

Sever selection problem is an extremely difficult problem, and no prevailing approach proposed by far. It seems that the simplest algorithms (like the random selection [72] and greedy algorithm [73]) can provide pretty good results in practice.

2.4 Protocols

Protocols dealing with packets distribution, transmission and reassembling is an essential part of PSMC.

J. Liu et al. proposed an enhanced TCP/IP protocol called "ATCP", by adding a thin layer between TCP and IP layer, to deal with high bit error rates problem in ad hoc network.

For the packets transmission, we have investigated various approaches, like SOCKS proxy server [54], Zebedee [53], IP Tunnel[55] and IPSec[58, 78].

Proxies are mostly used to control, monitor or outbound traffic. There are two types of proxy servers: cache proxies, which cache the requested data, and SOCKS proxies, which is like an old switch board and can cross wires your connection through the system to another outside connection [54].

SOCKS proxy servers have several drawbacks. First, it didn't support UDP, only TCP. Second, it didn't support certain applications, like FTP. Third, it runs slow [54].

Zebedee [53] is a simple program to establish an encrypted, compressed "tunnel" for TCP/IP or UDP data transfer between two systems.

IP tunnel [55] (also called IP encapsulation) is a technique to encapsulate IP datagram within IP datagrams. This allows datagrams destined for one IP address to be wrapped and redirected to another IP address. The IP tunnel can be set up from Linux to Linux, windows to windows, or between Linux and windows (windows must be Windows 2000 server and above).

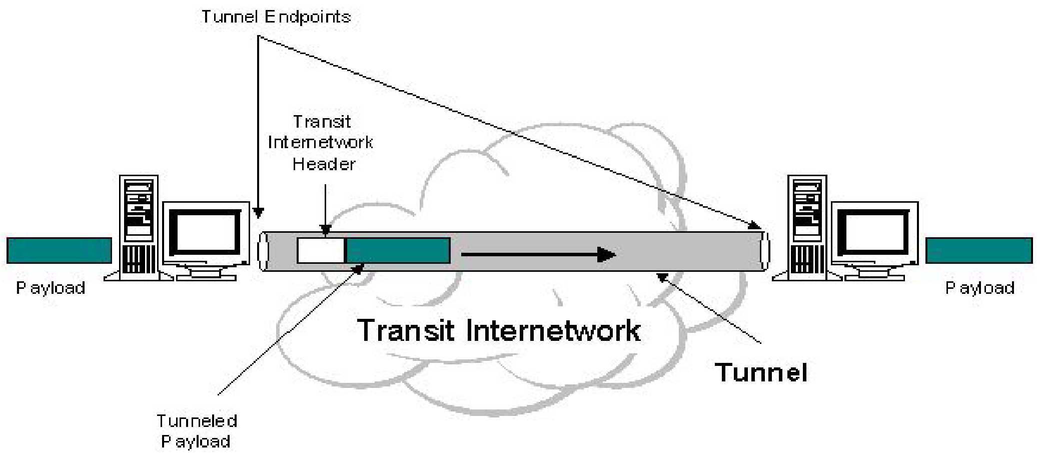


Figure 2.6: IP over IP tunneling [55]

The advantages of using IP tunnel is as follows. It is a layer two / three protocol; therefore all the upper layer protocols and applications can use it transparently. Second, IP Tunnel is essentially a device descriptor, and it consumes limited system resources on the server.

IP Tunnel brings overhead by an extra set of IP headers. Typically it is 20 bytes per packet. So if the normal packet size (MTU) on a network is 1500 bytes, a packet that is sent through a tunnel can only be 1480 bytes big, therefore the payload size is reduced. This also causes fragmentation and reassembly overhead. But these overheads can be reduced or avoided by setting smaller MTU at the client side.

IPSec [58] is an extension to the IP protocol which provides security to the IP and the upper-layer protocols. The IPsec architecture is described in the RFC2401. IPsec uses two different protocols – Authentication Header (AH) and Encapsulating Security Payload (ESP) - to ensure the authentication, integrity and confidentiality of the communication. It can protect either the entire IP datagram or only the upper-layer protocols. The appropiate modes are called tunnel mode and transport mode. In tunnel mode the IP datagram is fully encapsulated by a new IP datagram using the IPsec protocol. In transport mode only the payload of the IP datagram is handled by the IPsec protocol inserting the IPsec header between the IP header and the upper-layer protocol header [78].

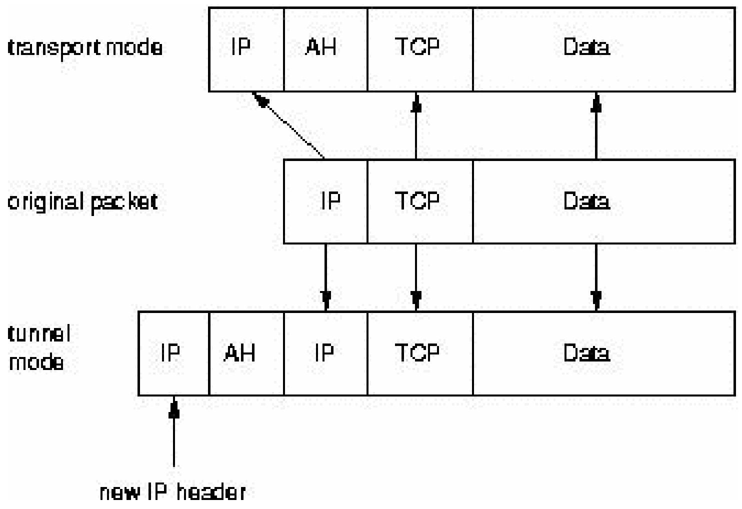


Figure 2.7: IPsec tunnel and transport mode [58]

IPSec and IP tunnel has been used widely in Virtual Private Network (VPN) [79]. A VPN is a private network that uses the Internet to securely connect remote sites or users together. Instead of using a dedicated, real-world connection such as a leased line, a VPN uses a “virtual” connection routed through the Internet. From the user’s perspective, a VPN operates transparently. The tunneling handshake and packets transmission mechanism in VPN is a good reference for PSMC packets transmission.

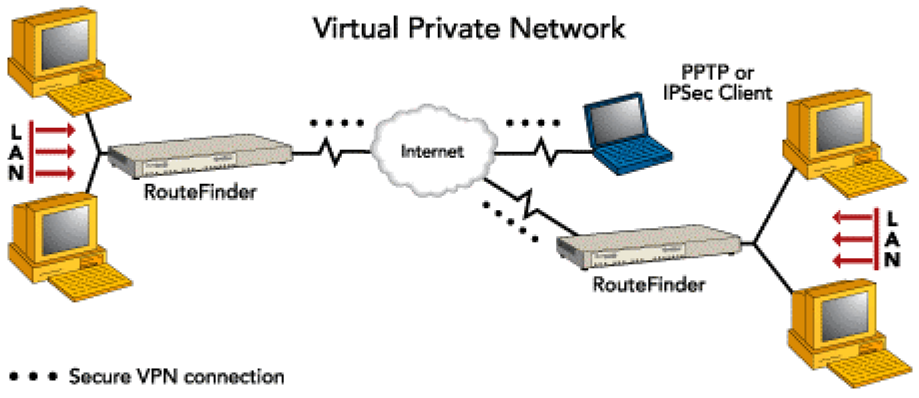


Figure 2.8: VPN [79]

Linux Virtual Server (LVS) [80] is a highly scalable and highly available server built on a cluster of real servers. The architecture of the cluster is transparent to end users, and the users see only a single virtual server. The packets distribution and re-assembly mechanism in LVS on load balancer is a good reference for PSMC packets distribution and re-assembly.

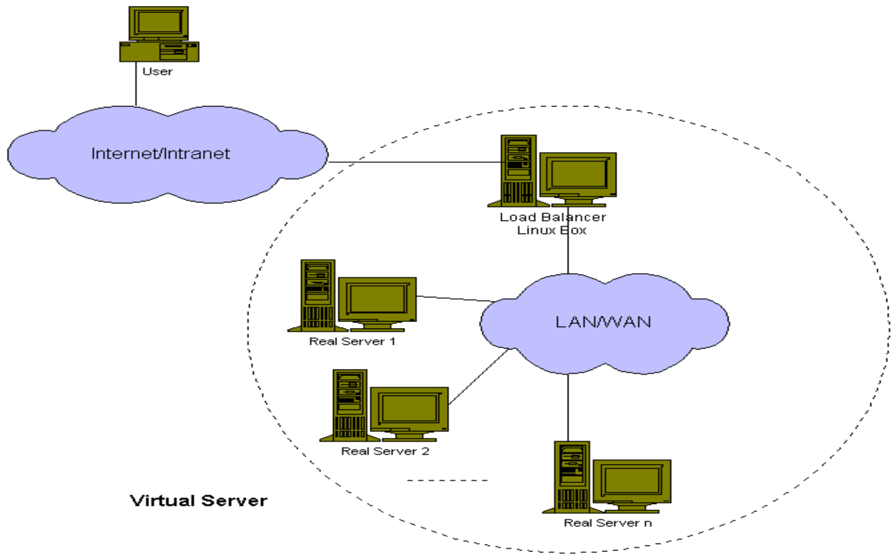


Figure 2.9: Linux Virtual Server [80]

Chapter 3

Algorithms for PSMC

This chapter presents several algorithms for proxy server selection in PSMC.

To set up multipath connections in PSMC, the very first step is to select the desired proxy servers. Then we can set up indirect routes via the selected proxy servers and spread packets over the multiple indirect routes. Different server selections result in significantly different performance [10]. Therefore, proxy servers selection is a critical decision in PSMC.

The chapter is organized as follows. Section 3.1 presents brutal force algorithms and genetic algorithms for parallel download from multiple mirror sites. Section 3.2 proposes the algorithms for proxy server selection in PSMC.

3.1 Algorithms for parallel download from multiple mirror sites.

As described in Chapter 2, the network topology of parallel download from multiple mirror sites could be viewed as half part of the PSMC. Therefore the algorithms for parallel download can be extended for PSMC.

We have developed heuristic algorithms to choose best mirror sites for parallel download from multiple mirror sites [61]. For detailed information, please refer to [61] or Appendix A.

In that paper, we present a mathematical model that simulates the problem of parallel download from multiple mirror sites. Based on the model, we present algorithms for selecting the best subset of mirror sites for parallel download. The versions of brutal force algorithms and genetic algorithms are implemented. Performance of these algorithms on the simulated network topology as well as a real-world network topology is presented.

Figure 3.1 is the test result of algorithm execution time vs. simulated network size in the paper. The red line is the brutal force algorithm, the execution time increases dramatically when network size increases. The green line is the genetic algorithm, the execution time keep linear even when the network size reach 1200 nodes.

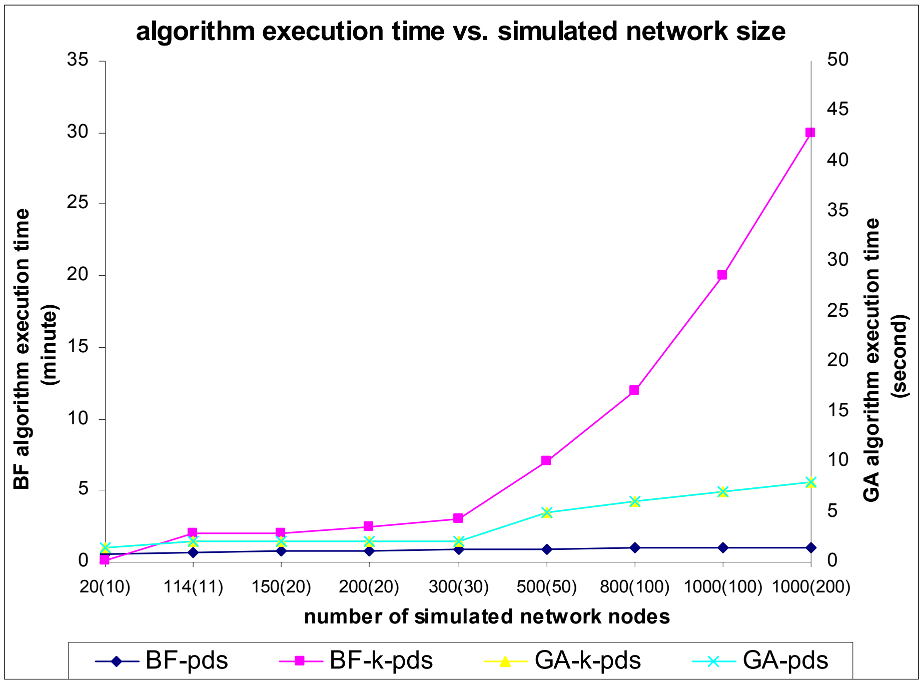


Figure 3.1: Algorithm execution time vs. simulated network size.

The performance result is quite encouraging. The heuristic algorithms converge fast enough and generate good enough result for small-scale network as well as large-scale network with hundreds of nodes.

3.2 Algorithms for selecting proxy servers for PSMC

In PSMC, proxy servers selection is a critical decision. Different server selections result in different performance, and a bad choice might be 5-10 times slower than the best choice [10] in some scenarios.

We plan to develop algorithms to solve the following problems in PSMC.

a) **Server Selection Problem**. Given the target server location and a set of proxy servers, choose the best proxy server(s) for a client or for a group of client, to achieve best performance, in terms of bandwidth, latency or throughput. For example, in Figure 3.2, we have a client, a target server and a set of proxy servers, we need to choose the best proxy server(s) to achieve the max aggregate network bandwidth, or best user-perceived performance.

When there are hundreds of clients and proxy servers, the selected paths might interfere with each other, especially when the paths have join nodes. Therefore, the server selection algorithm needs find the disjoint paths to achieve the best performance.

However, finding the disjoint paths is not an easy task. One of the solutions is **Proxy Server Partition** to select geographically diverse proxy servers. Assuming we have enough proxy servers, we can partition the clients and the proxy servers into several zones. A client will only use proxy servers assigned to its partition. By proper server partition, we can minimize the possibility of joint paths.

Figure 3.3 is an example to partition the proxy servers based on geographical location. Better partition algorithms with consideration of network distance and location need to be studied.

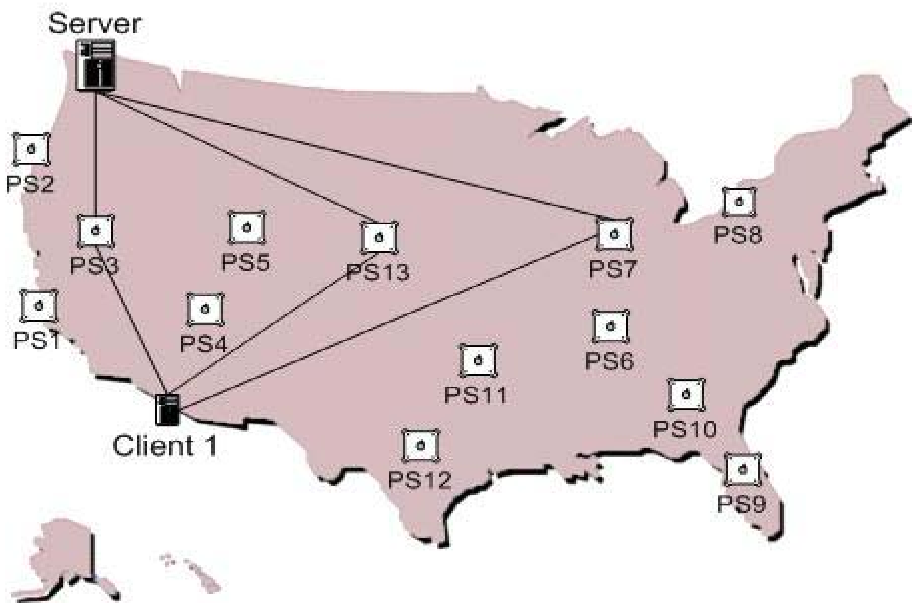


Figure 3.2 Server selection problem

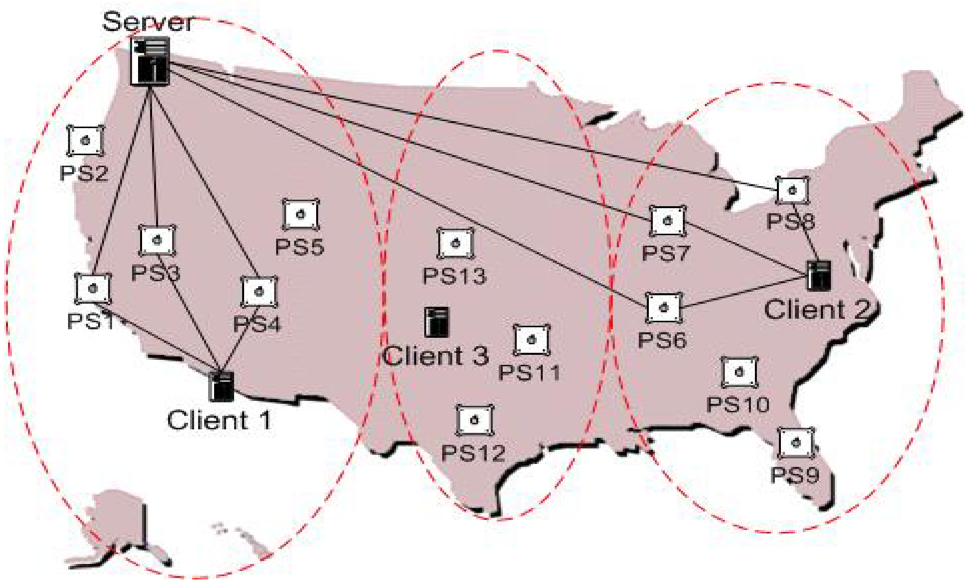


Figure 3.3: Server partition problem

b) **Server Placement Problem**. Given the target server location and a set of nodes, choose the best node(s) to place the proxy servers, for certain connection requirements, like maximize the aggregate network bandwidth. In Figure 3.4, assuming we have a server and a set of potential nodes, if we want to add more proxy servers, which nodes will be the best candidates?

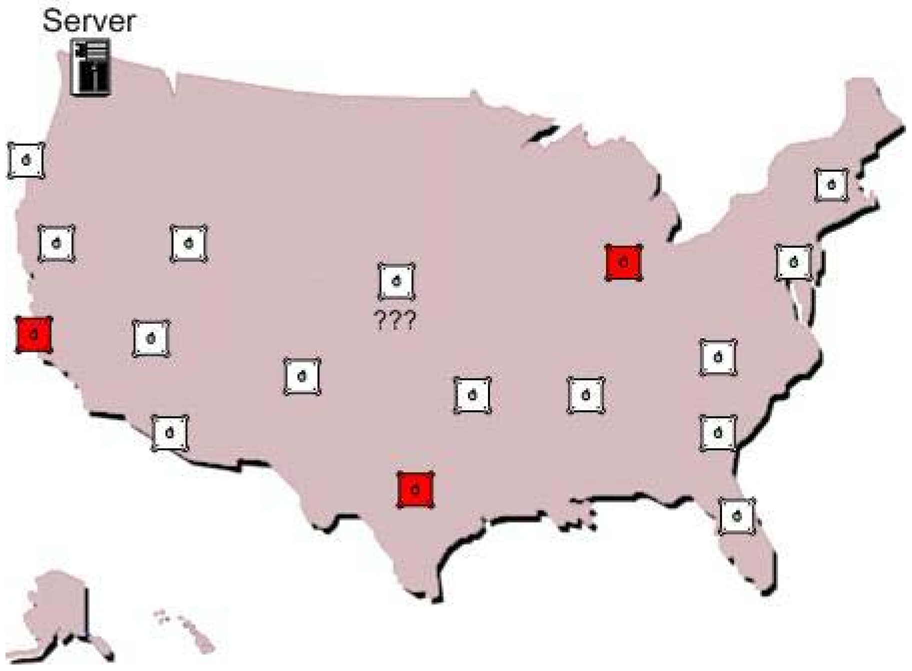


Figure 3.4 Server placement problem.

Sever selection/placement problem are extremely difficult problems, and no prevailing approach proposed by far. It seems that the simplest algorithms (like the random selection [72] and greedy algorithm [73]) can provide pretty good results in practice.

The Sever selection/placement problem seems to be NP problem. We plan to extend the heuristic algorithms in Section 3.1 to solve the above the problems. Another direction is to loosen the optimal constrains to simplify the problem to make it solvable in P-time.

Chapter 4

PSMC Protocols

This chapter presents protocols designed to distribute, transport and reassemble packets for PSMC.

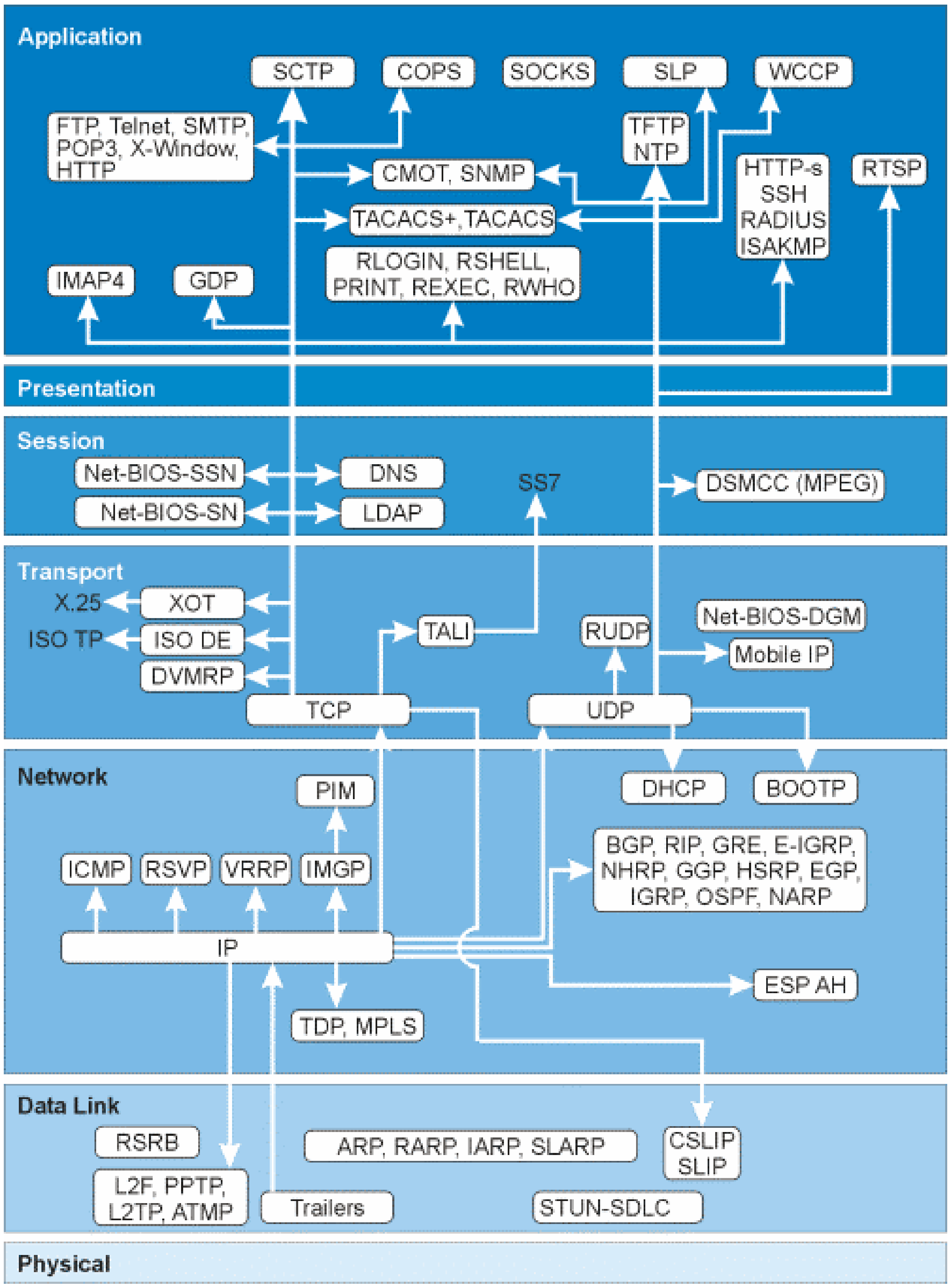
There are three essential parts which make a PSMC network viable. The PSMC sender, who distributes packets among the selected multiple paths; the proxy servers, who forward packets through the network; the PSMC receiver, who reassembles packets from multiple paths, and forward the result to the end user.

TCP/IP is the predominant transport protocol in today's Internet. Studies have shown that the sending, receiving and transmitting mechanisms in TCP/IP protocol have significant impact on network performance and end-host performance [29]. As we can foresee, in multipath connections, those mechanisms would have even more dramatic impact on performance.

The rest of the chapter is organized as follows. Section 4.1 presents protocols for packets distribution and reassembling. Section 4.2 presents IP tunnel enhancement for packets transmission.

4.1 Protocols for packets distribution and reassembling

We propose to add a thin layer between TCP/UDP and IP layer [59, 9, 64] to deal with packets distribution and reassembling. We plan to implement it in Redhat Linux platform. Therefore we need to modify Linux kernel and related software packages to support the scheme.



Firgure 4.1: TCP/IP map [64]

The benefits of putting a thin layer between TCP and IP is as follows:

a) We can utilize existing TCP/IP protocols, particularly the packets re-sequencing mechanism.

b) We can hide the complexity of multipath connections from upper layer users

c) We can still maintain the high end-to-end TCP throughput.

We will further investigate how to resend and re-sequence packets when needed. For example, a packet gets lost for some reason during transmission, the receiver needs to notify the sender, and the sender will resend the packet through another indirect route. This will be a very challenging problem in PSMC protocols.

The packets distribution and re-assembly mechanism in LVS at load balancer is a good reference for PSMC packets distribution and re-assembly.

We plan to implement various scheduling algorithms for packets distribution and reassembling, like round robin scheduling and least usage scheduling [62].

4.2 Protocols for packets transmission

For the packets transmission, after investigate various approaches, like SOCKS proxy server [54], Zebedee [53], we proposed to use IP Tunnel[55] or IPSec[58] to enable indirect routes via proxy servers.

IP tunnel [55] (also called IP encapsulation) is a technique to encapsulate IP datagram within IP datagrams. This allows datagrams destined for one IP address to be wrapped and redirected to another IP address. The IP tunnel can be set up from Linux to Linux, windows to windows, or between Linux and windows (windows must be Windows 2000 server and above).

IPSec [58] is an extension to the IP protocol which provides security to the IP and the upper-layer protocols. The IPsec architecture is described in the RFC2401. IPsec uses two different protocols – Authentication Header (AH) and Encapsulating Security

Payload (ESP) - to ensure the authentication, integrity and confidentiality of the communication. It can protect either the entire IP datagram or only the upper-layer protocols. The appropiate modes are called tunnel mode and transport mode. In tunnel mode the IP datagram is fully encapsulated by a new IP datagram using the IPsec protocol. In transport mode only the payload of the IP datagram is handled by the IPsec protocol inserting the IPsec header between the IP header and the upper-layer protocol header [78].

The advantages of using IP tunnel / IPSec is as follows. It is a layer two / three protocol; therefore all the upper layer protocols and applications can use it transparently. Second, IP Tunnel and IPSec are widely used protocols and available in various operating systems. Third, IP Tunnel and IPSec are essentially device descriptors, and it consumes limited system resources on the server. The transmission efficiency and performance of IP Tunnel and IPSec are also acceptable.

We will utilize and enhance existing tunneling protocols for PSMC. In particular, we will study issues like tunnel handshake, host authentication, security mechanism and fail-over mechanism when packets are lost or links are broken.

We will evaluate the performance, overhead and security on IP tunnel and IPSec, and choose one to be the primary implementation in PSMC.

The tunneling handshake and packets transmission mechanism in VPN is a good reference for PSMC packets transmission.

One special issue in PSMC is sticky-connection. Certain protocols, like http keep alive, require connection from same IP address. In PSMC, we need to explore the possibility of sending packets through a specific route upon user request.

A related issue of PSMC is as follows. In enterprise environment where we have more control over the network infrastructure and more trust relationship between groups, we have the following additional options to set up multiple indirect routes. We can modify the routing table of routers dynamically and directly, or use source routing, or use MPLS to enable multipath connections. We will do some feasibility study on this issue.

Chapter 5

PSMC Prototype and Applications

This chapter presents several PSMC prototype and applications.

Compared with other multipath connections approaches, PSMC requires feasible changes on existing network, and can be more conveniently and adaptively deployed in various network environments. At the same time, PSMC maintains a high end-to-end TCP/UDP throughput, ensures the network efficiency and performance. Therefore, a vast number of applications in various environments can benefit from utilizing PSMC.

A typical application of PSMC is Secure Collective Defense (SCOLD) [60] network defending against Distributed Denial of Service (DDoS) attacks. PSMC can also be utilized to provide additional bandwidth based on operational requirement in enterprise network. PSMC can be utilized to provide QoS for various applications, like video streaming. Other applications includes content switch cluster server load balancing, wireless ad hoc network, parallel download from multiple mirror sites.

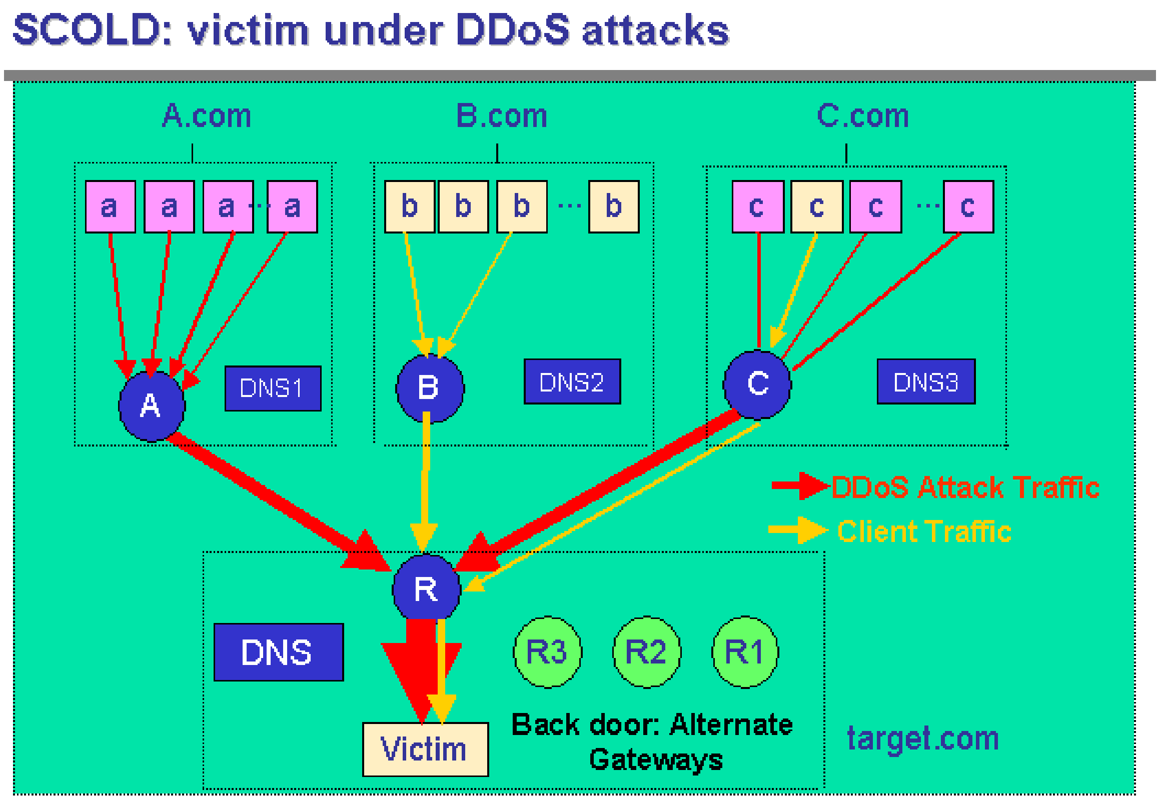
The rest of the chapter is organized as follows. Section 5.1 presents Secure Collective Defense (SCOLD) network. Section 5.2 presents PSMC in wireless ad hoc network. Section 5.3 presents other PSMC applications, like using PSMC to provide additional bandwidth based on operational requirement, QoS for video streaming, parallel download, server load balancing. Section 5.4 compares PSMC and other multipath connections approaches.

5.1 Secure Collective Defense (SCOLD) network

We have finished the prototype of Secure Collective Defense (SCOLD) [60] network. We plan to enhance SCOLD for better security, functionality, scalability, reliability and performance. For detailed information, please refer to [60] or Appendix B.

SCOLD tolerates the DDoS attacks through indirect routes via proxy servers, and improves network performance by spreading packets among multiple indirect routes. SCOLD incorporates various cyber security techniques, like secure DNS update [60], Autonomous Anti-DDoS network [39], IDIP protocols [41].

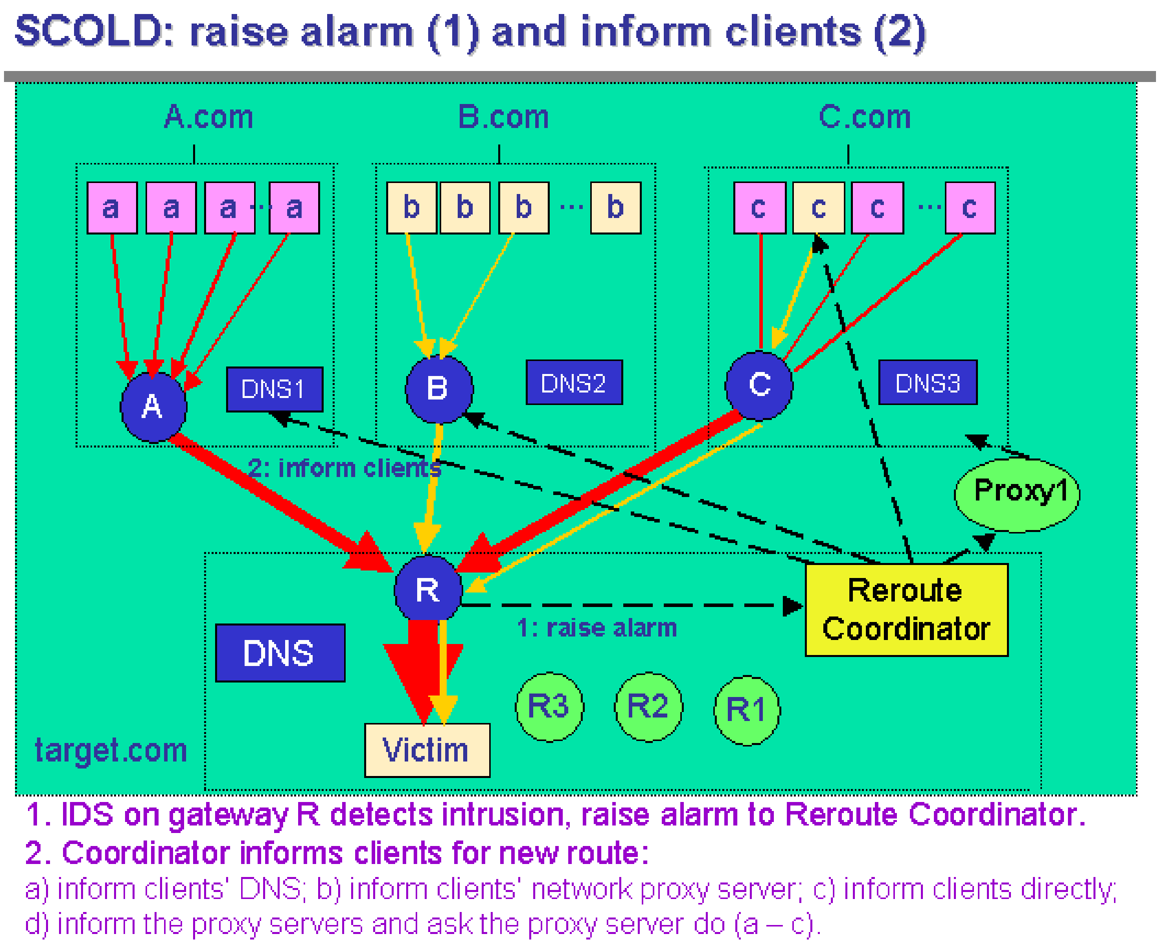
Below are viewgraphs illustrating the concept of SCOLD.



Main gateway R under attacks, we want to inform Clients to go through the "back door" - alternate gateways R1- R3. We needs to hide IPs of R1-R3, otherwise they are subject to potential attacks too.
how to inform Clients? how to hide IPs of R1-R3?

Figure 5.1: Victim server under DDoS attacks

Figure 5.1 shows the motivation of SCOLD. A lot of organizations have multiple gateways or multi-homing. When the main gateway is under serious DDoS attacks, we want to inform the clients to go through the "back door" – alternate gateway. But we needs to hide the IP addresses of those alternate gateways, otherwise they are subject to potential attacks too. Therefore, the two key questions are: how to inform the clients about the new route information? how to hide the IP addresses of the alternate gateways?


SCOLD: raise alarm (1) and inform clients (2)

1. IDS on gateway R detects intrusion, raise alarm to Reroute Coordinator.
2. Coordinator informs clients for new route:
a) inform clients' DNS; b) inform clients' network proxy server; c) inform clients directly;
d) inform the proxy servers and ask the proxy server do (a – c).

Figure 5.2: Raise alarm (step 1) and inform clients (step 2)

Figure 5.2 shows the design of SCOLD. We recruit some proxy servers as the intermediate connection relay servers. In this way, we can hide the IP addresses of alternate gateways. Actually the proxy servers can provide more benefits than this. The steps of SCOLD activation are as follows. First, IDS on main gateway detects the intrusion and raise alarm to notify the reroute coordinator. Second, we want to inform the clients about the reroute information. There are several options.

a) The coordinator informs the clients' DNS, the clients get reroute information by querying DNS. This requires the modification of DNS.

b) The coordinator informs the clients directly. The clients need to have daemon servers listening to a certain port. This may not be able to go through firewall.

c) The coordinator informs the clients' proxy servers (like the proxy servers in most enterprise network). The clients' traffic is re-directed by the proxy servers.

d) The coordinator notifies a selected proxy server, and let the proxy server inform the clients about the reroute information, as described in a-c. Because the coordinator is inside the target server network, and is subject to potential DDoS attacks, therefore, we also want to hide the IP address of the coordinator, and ask proxy servers to interact with the clients.

Figure 5.3 shows the step 3 of SCOLD. The clients get the reroute information, and set up indirect route through proxy servers. The proxy servers will check and redirect the clients' traffic to alternate gateway. The proxy servers are equipped with IDS, so the attacking traffic can be blocked at the proxy servers.

As we can see, the proxy servers play a important role in SCOLD. First, it can hide the IP addresses of alternate gateways and coordinator. Second, it can block the attacking traffic. Third, it provide the alternate route and potential multiple routes.

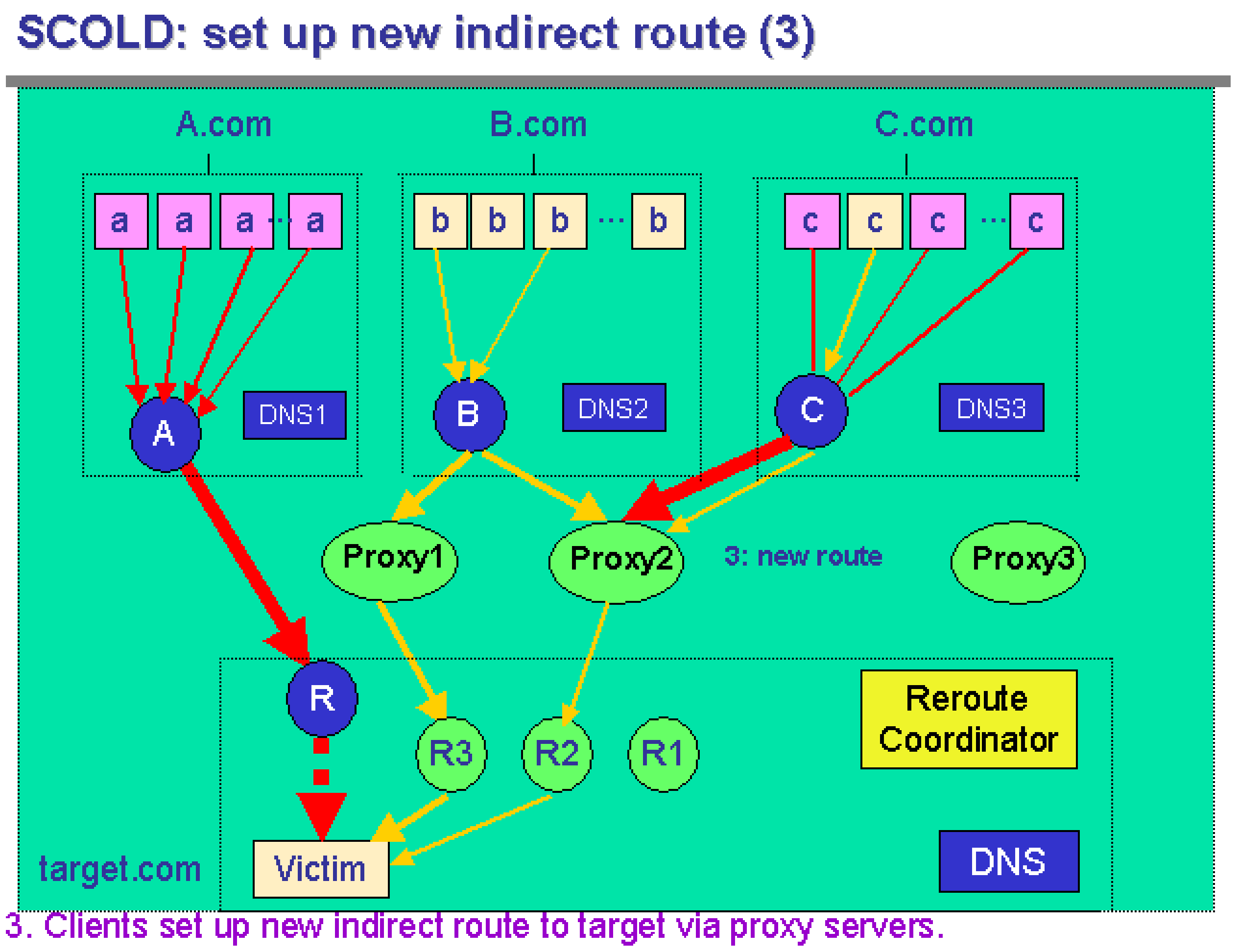


Figure 5.3: Set up new indirect route (step 3)

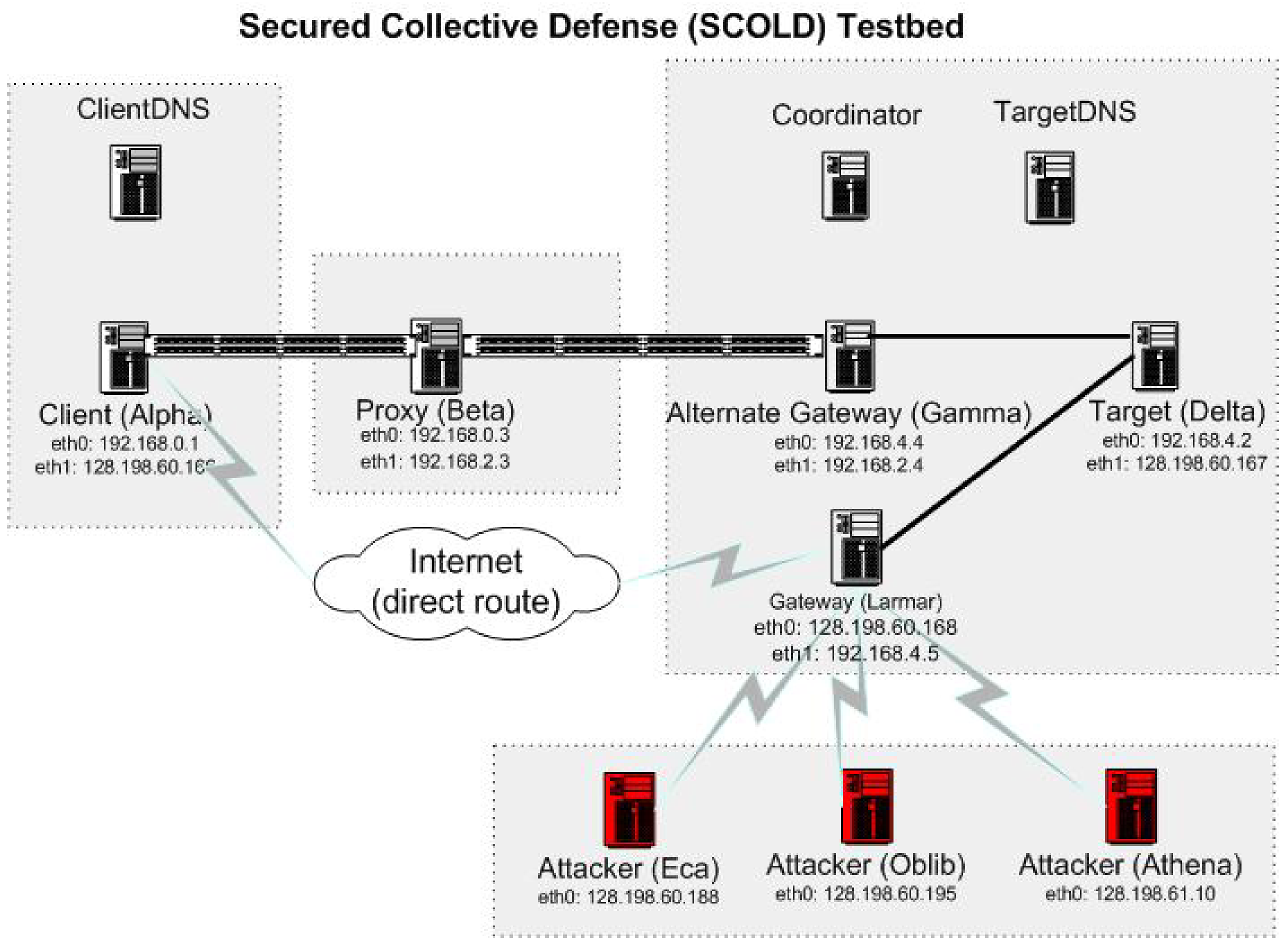
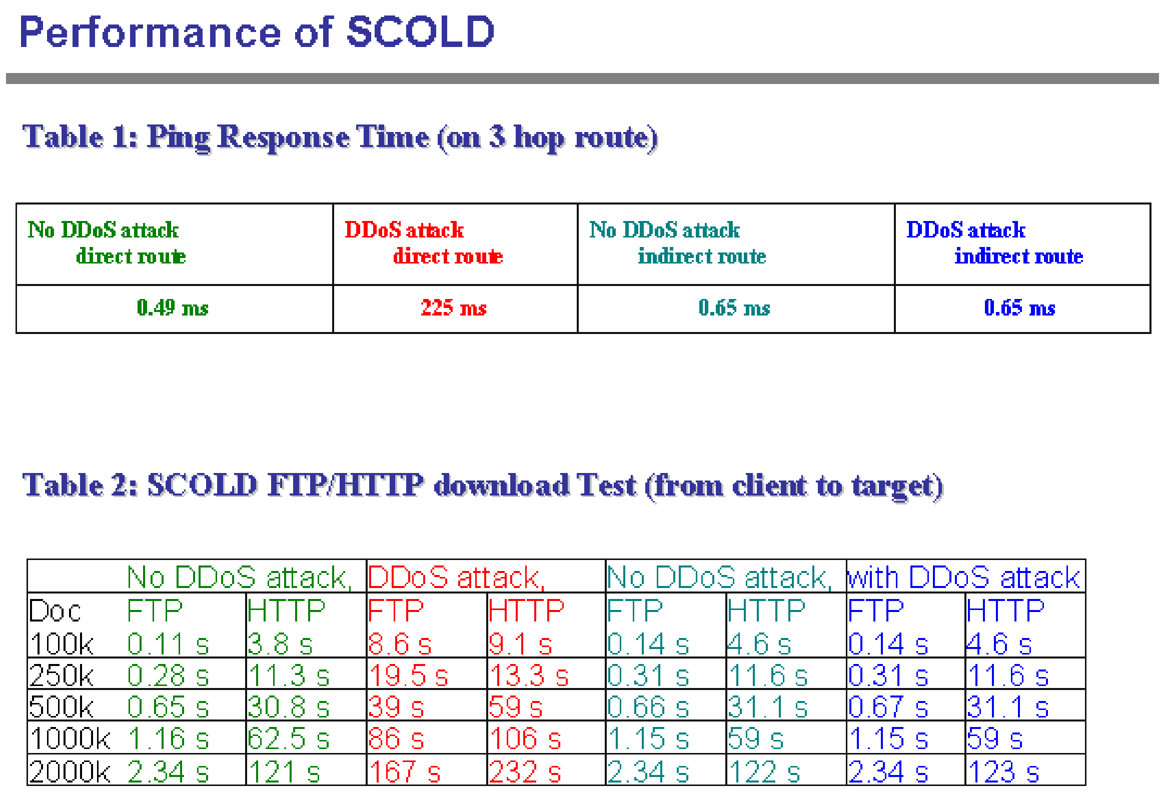


Figure 5.4: SCOLD test bed viewgraph

We get some preliminary test result on PSMC performance, which shows that the overhead of PSMC is in an acceptable range, compared with the possible delay caused by DDoS attacks.



Performance of SCOLD

Table 1: Ping Response Time (on 3 hop route)

No DDoS attack direct route	DDoS attack direct route	No DDoS attack indirect route	DDoS attack indirect route
0.49 ms	225 ms	0.65 ms	0.65 ms

Table 2: SCOLD FTP/HTTP download Test (from client to target)

	No DDoS attack,		DDoS attack,		No DDoS attack,		with DDoS attack	
Doc	FTP	HTTP	FTP	HTTP	FTP	HTTP	FTP	HTTP
100k	0.11 s	3.8 s	8.6 s	9.1 s	0.14 s	4.6 s	0.14 s	4.6 s
250k	0.28 s	11.3 s	19.5 s	13.3 s	0.31 s	11.6 s	0.31 s	11.6 s
500k	0.65 s	30.8 s	39 s	59 s	0.66 s	31.1 s	0.67 s	31.1 s
1000k	1.16 s	62.5 s	86 s	106 s	1.15 s	59 s	1.15 s	59 s
2000k	2.34 s	121 s	167 s	232 s	2.34 s	122 s	2.34 s	123 s

Figure 5.5: preliminary result of SCOLD

We will further evaluate the overhead of PSMC, including tunneling overhead, handshake overhead, distribution/reassembling overhead. We will evaluate the performance of multipath connections in terms of response time, throughput and bandwidth.

The research results and insights obtained from SCOLD can improve the security of the networks and have a broader impact on the network architecture and the client side network software interface.

5.2 PSMC in Wireless ad hoc network

An ad hoc wireless network is a collection of wireless mobile hosts forming an instant deployable network without the aid of any base station, other infrastructure or centralized administration. The most popular routing approach in ad hoc network is on-demand routing because of its effectiveness and efficiency. On-demand routing protocols build routes only when a node needs to send data packets to a destination. Each node operates as a specialized router, and routes are obtained on-demand with no reliance on periodic advertisements.

In ad hoc environment, the network topology keep changing, which means the multiple path connections between two nodes may change frequently. This brings a challenging problem of resending and re-sequencing packets. How to effectively distribute and reassemble packets in ad hoc network is another interesting research issue. PSMC could be used in wireless ad hoc network to transmit heavy traffic betweens ad hoc nodes, for example, transferring large data file or video conferencing among ad hoc nodes.

We plan to set up a wireless ad hoc network test bed and implement PSMC. We will evaluate the testing result, and compare it with wired network.

5.3 PSMC other applications

a) PSMC for QoS of video streaming

PSMC can be used for QoS of video streaming. The potential multiple paths can be used to transmit different portion of video stream [81].

b) PSMC for additional bandwidth upon operational needs

In corporate intranet, based on operational needs, PSMC can be utilized to provide additional bandwidth dynamically.

c) Parallel download from multiple mirror sites

We will implement an application to do parallel download from multiple mirror sites, utilizing PSMC server selection algorithms.

d) Content switch cluster servers load balancing

We will study how to implement PSMC for content switch cluster servers load balancing. If the connections between the load balancer and the actual processing servers are through the Internet, then the connections could be a possible bottleneck, despite of the high performance on balancer and actual servers. PSMC could be used to increase effective bandwidth between load balancer and the actual servers.

5.4 Comparison with other multipath connections approaches

We plan to set up a small-scale source routing network in campus network, and compare the result with PSMC, in terms of overhead, performance, bandwidth and throughput.

We will systematically evaluate the pro and cons of PSMC, and compare the result with other multipath connections approaches.

Chapter 6

Security Issues on PSMC

This chapter discusses security issues on PSMC.

The increasing network attacks reveal one of the fundamental security problems of today's Internet. Many Internet services, such as DNS and routing protocols, were not originally designed with security as one of the basic requirements. It is difficult to modify the existing protocols or network architecture without significant work. At the same time, it offers an opportunity to create new, secure and reliable network protocols, and packet delivery systems.

The rest of the chapter is organized as follows. Section 6.1 discusses the potential attacks or misuse of PSMC. Section 6.2 discusses the collective defense mechanism.

6.1 Potential attacks or misuse on PSMC

We plan to investigate security issues raised by misusing of PSMC, or potential attacks against PSMC.

Malicious clients in Internet can misuse the abundant bandwidth provided by PSMC for delivery of numerous attacking messages to the victim. Therefore, the proxy servers in PSMC must be equipped with IDS, firewall and further security mechanisms. We will investigate how to put admission control mechanisms and resources management mechanisms in the proxy servers.

We envision several potential attacks on PSMC system.

Malicious clients can issue tunnel-to-death attacks similar to ping-to-death.

Malicious clients may hack into a proxy server and redirect the clients' traffic to other places. How to detect and handle the compromised proxy servers is an important security issue in PSMC.

6.2 Collective defense mechanism

Today's Internet is managed in a distributed manner with highly independence; therefore it is hard to enforce collective security policies or defense mechanisms among its participants. On the other hand, malicious clients in Internet can take advantage of this and recruit thousands of Internet nodes to build a collective attack network. The unbalance of defense and attack brings enormous opportunities for DDoS attacks.
We plan to discuss the collective defense mechanism so that different organizations in Internet could be tied with better cooperation and tighten collaboration.

Chapter 7

Conclusion

In the dissertation, we propose to study proxy server based multipath connections (PSMC).

The key idea of PSMC is as follows.

a) By using a set of connection relay proxy servers, we could set up indirect routes via the proxy servers, and transport packets over the network through the indirect routes.

b) By enhancing existing TCP/IP protocols, we could efficiently distribute and reassemble packets among multiple paths at two end nodes, and increase end-to-end TCP throughput.

This approach offers applications with the ability to improve network security, reliability and performance.

PSMC has the following unique advantages over other multipath connections approaches:

a) **Compatibility**: PSMC utilizes and enhances existing TCP/IP protocols and network infrastructure to distribute, transport and reassemble packets. Unlike some multipath connection approaches (ie. link aggregation, multipath routing), which require changes on physical network infrastructure, PSMC only requires some feasible changes on network software and protocols. This ensures the compatibility with current Internet.

b) **Flexibility**: PSMC can be more conveniently and adaptively deployed in various network environments. PSMC gives the end users more control and flexibility on how to set up multipath connections.

c) **Usability**: A large number of applications in various categories could benefit from utilizing PSMC. For example, secure collective defense network (SCOLD), which provides alternate route for DDoS attacks intrusion tolerance. PSMC can also be utilized to provide additional bandwidth based on operational requirement in enterprise network. PSMC can be utilized to provide QoS for various applications, like video streaming.

We will systematically study PSMC in the following areas:

1) PSMC related algorithms. We develop proxy server selection algorithms for PSMC.

2) Protocols to support PSMC. We design and implement protocols to distribute, transport and reassemble packets for PSMC.

3) PSMC prototype and applications. We develop several PSMC applications like Secure Collective Defense (SCOLD) network, PSMC in wireless ad hoc network, PSMC providing additional bandwidth upon request, PSMC in QoS for video streaming, content switch with cluster servers, and parallel download from multiple mirror sites.

4) Security issues related to PSMC. We further investigate security issues related to PSMC, like the potential attacks, the misuse, and the collective defense mechanisms.

The research result and insight gained from PSMC could have broader impact on the protocols and security of today’s Internet.

References

[1] Johnny Chen, “New Approaches to Routing for Large-Scale Data Network”, PhD Thesis, 1998.

[2] S. Vutukury and J.J. Garcia-Luna-Aceves, “MDVA: A distance-vector multipath routing protocol,” Proceedings of the IEEE INFOCOM, pp. 557–564, 2001.

[3] IEEE 802, “IEEE 802.3ad Link Aggregation”, http://grouper.ieee.org/groups/802/3/ad/index.html

[4] Information Sciences Institute, “INTERNET PROTOCOL RFC: 791”, http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc0791.html

[5] D.B. Johnson and D.A. Maltz, “Dynamic Source Routing in Ad HocWireless
Networks”, In Mobile Computing, edited by Tomasz Imielinski and
Hank Korth, Chapter 5, Kluwer Academic Publishers, 1996, pp. 153-181.

[6] Lianfang Zhang, Zenghua Zhao, Yantai Shu, Lei Wang, and Oliver W.W. Yang; ``Load Balancing of Multipath Source Routing in Ad Hoc Networks''; In Proceedings of IEEE International Conference on Communications (ICC 2002), April 2002.

[7] C.E. Perkins and E.M. Royer. "Ad hoc On-Demand Distance Vector Routing." In Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and applications, February 1999, pp. 90-100.

[8] S. Lee and M. Gerla. “Split Multipath Routing with Maximally Disjoint paths in Ad Hoc Networks”. In Technical Report in University of California, 2000.

[9] Christoph Simon, “Howto to use more than one independent Internet connection”,
http://www.ssi.bg/~ja/nano.txt

[10] Pablo Rodriguez, Andreas Kirpal, Ernst W. Biersack, “Parallel-Access for Mirror Sites in the Internet”, Proceeding of Infocom, 2000. http://www.ieee-infocom.org/2000/papers/65.ps

[11] John Byers, Michael Luby, and Michael Mitzenmacher, “Accessing multiple mirror sites in parallel: Using tornado codes to speed up downloads,” in INFOCOM 99, Apr. 1999.

[12] N. F. Maxemchuk, “Dispersity Routing in Store and Forward Networks”,
Ph.D. thesis, University of Pennsylvania, 1975.

[13] N. F. Maxemchuk, “Dispersity Routing in High-Speed Networks”, computer networks and ISDN systems, vol. 25, 1993

[14] Sun Microsystems, “Link Aggregation Trunking”,

http://grouper.ieee.org/groups/802/3/trunk_study/tutorial/ahtrunk.pdf

[15] I. Cidon, R. Rom, and Y. Shavitt, “Analysis of Multi-Path Routing”, IEEE/ACM Transactions on Networking, vol. 7, no. 6, Dec. 1999, pp. 885-896.

[16] S. Murthy and J.J. Garcia-Luna-Aceves, “Congestion-Oriented Shortest Multipath Routing”, Proceedings of IEEE INFOCOM'96, San Francisco, CA, Mar. 1996, pp. 1028-1036.

[17] R. Ogier, V. Rutenburg, and N. Shacham, “Distributed Algorithms for Computing Shortest Pairs of Disjoint Paths”, IEEE Transactions on Information Theory, vol. 39, no. 2, Mar. 1993, pp. 443-455.

[18] D. Sidhu, R. Nair, and S. Abdallah, “Finding Disjoint Paths in Networks”, Proceedings of ACM SIGCOMM'91, Zurich, Switzerland, Sep. 1991, pp. 43-51.

[19] N. Taft-Plotkin, B. Bellur, and R. Ogier, “Quality-of-Service Routing Using Maximally Disjoint Paths”, Proceedings of IEEE IWQoS'99, London, UK, Jun. 1999, pp. 119-128.

[20] W.T. Zaumen and J.J. Garcia-Luna-Aceves, “Loop-Free Multipath Routing Using Generalized Diffusing Computations”, Proceedings of IEEE INFOCOM' 98, San Francisco, CA, Mar. 1998, pp. 1408-1417.

[21] Jerome H. Saltzer, David P. Reed, David D. Clark, “Source Routing for Campus-wide Internet Transport”, IFIP Working Group 6.4 Workshop on Local Area Networks in Zurich, 1980.

[22] Network Sorcery, Inc., “Loose Source Routing”,

http://www.networksorcery.com/enp/protocol/ip/option003.htm

[23] S.-J. Lee and M. Gerla, “AODV-BR: Backup Routing in Ad hoc Networks”, Proceedings of IEEE WCNC 2000, Chicago, IL, Sep. 2000.

[24] A. Nasipuri and S.R. Das, “On-Demand Multipath Routing for Mobile Ad Hoc Networks”, Proceedings of IEEE ICCCN'99, Boston, MA, Oct. 1999, pp. 64-70.

[25] V.D. Park and M.S. Corson, “A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks”, Proceedings of IEEE INFOCOM' 97, Kobe, Japan, Apr. 1997, pp. 1405-1413.

[26] J. Raju and J.J. Garcia-Luna-Aceves, “A New Approach to On-demand Loop-Free Multipath Routing”, Proceedings of IEEE ICCCN'99, Boston, MA, Oct. 1999, pp. 522-527.

[27] N.F. Maxemchuk, “Dispersity routing on ATM networks”, In Proc. IEEE Infocom, volume 1, pages 347-357, 1993.

[28] E. Gustafsson and G. Karlsson, “A literature survey on traffic dispersion,” IEEE Network, vol. 11, no. 2, pp. 28–36, March 1997.

[29] Vern Paxson, “Measurements and Analysis of End-to-End Internet Dynamics ”, Ph.D. dissertation at UC Berkley.

[30] Ellen W. Zegura, Mostafa H. Ammar, Zongming Fei, and Samrat Bhattacharjee, “Application-Layer Anycasting: A Server Selection Architecture and Use in a Replicated Web Service,” IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 8, NO. 4, AUGUST 2000, pp. 455-466.

[31] Kevin Lai and Mary Baker, "Nettimer: A Tool for Measuring Bottleneck Link Bandwidth", Proceedings of the USENIX Symposium on Internet Technologies and Systems, March 2001.

[32] Ellen W. Zegura, “GT-ITM: Georgia Tech Internetwork Topology Models”, http://www.cc.gatech.edu/projects/gtitm/

[33] John R. Koza, “Genetic Programming”, MIT Press, 1992.

[34] Jing Yang and Zhong Li, “Selecting best Redhat Mirror Sites for parallel download”, http://cs.uccs.edu/~cs522/proj2001/jyang.ppt

[35] Mark E. Crovella and Robert L. Carter, “Dynamic Server Selection in the Internet”, http://cs-www.bu.edu/faculty/crovella/paper-archive/hpcs95/paper.html

[36] P. Krishnan, D. Raz, Y. Shavitt, “The Cache Location problem”, in IEEE Trans. on Networking, 8 (5): pp. 568-582, Oct. 2000.

[37] Internetnews.com, “Massive DDoS Attack Hit DNS Root Servers”, http://www.internetnews.com/ent-news/article.php/1486981

[38] David Moore, Geoffrey M. Voelker and Stefan Savage, “Inferring Internet Denial-of-Service Activity 2001”, http://www.cs.ucsd.edu/~savage/papers/UsenixSec01.pdf

[39] Angela Cearns, Master Thesis “Design of an Autonomous Anti-DDoS network (A2D2)”, http://cs.uccs.edu/~chow/pub/master/acearns/doc/angThesis-final.pdf, 2002

[40] Steven Cheung, Ph.D. thesis “An Intrusion Tolerance Approach for Protecting Network Infrastructures”, http://citeseer.nj.nec.com/cheung99intrusion.html, 1999

[41] Network Associates Labs and Boeing, “IDIP Architecture”, http://zen.ece.ohiou.edu/~inbounds/DOCS/reldocs/IDIP_Architecture.doc, 2002.

[42] SVRLOC working group, “Service Location Protocol (SLP) Project”, http://www.srvloc.org/

[43] Jelena Mirkovic, Janice Martin and Peter Reiher, “A Taxonomy of DDoS Attacks and DDoS Defense Mechanisms”, http://www.lasr.cs.ucla.edu/ddos/ucla_tech_report_020018.pdf

[44] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, R. Morris, "Resilient Overlay Networks," In Proceedings of 18th ACM SOSP, October 2001.

[45] Information Sciences Institute, "Dynabone," http://www.isi.edu/dynabone/

[46] J. Yan, S. Early, R. Anderson, "The XenoService – A distributed defeat for distributed denial of service", In Proceedings of ISW 2000, October 2000.

[47] Asta Networks, "Vantage System Overview," http://www.astanetworks.com/products/vantage/

[48] BBN Technologies, "Applications that participate in their own defense," http://www.bbn.com/infosec/apod.html

[49] Michael D. Bauer,” Securing DNS and BIND”, ACM Linux Journal Volume 2000 Issue 78es, October 2000

[50] John Viega, Matt Messier & Pravir Chandra, “Network Security with OpenSSL”, O’Reilly, 2002

[51] Internet Software Consortium, “DNS BIND 9”, http://www.isc.org/products/BIND/

[52] OpenSSL, “OpenSSL”, http://www.openssl.org

[53] “Zebedee Secure IP tunnel”, http://www.winton.org.uk/zebedee/

[54] “SOCKS proxy server”, http://www.tldp.org/HOWTO/Firewall-HOWTO-11.html

[55] “IPIP tunnel”, http://www.europe.redhat.com/documentation/HOWTO/Net-HOWTO/x1284.php3

[56] Eric Green, “Glibc Howto”, http://www.imaxx.net/~thrytis/glibc/Glibc2-HOWTO.html

[57] Astalavista, Network Library Archive, http://www.astalavista.com/archive/index.asp?dir=ddos

[58] IPSec, “IP Security Protocol” , http://www.ietf.org/html.charters/ipsec-charter.html

[59] J. Liu and S. Singh. ATCP: TCP for Mobile Ad Hoc Networks, 2001.

[60] Edward Chow and Yu Cai, “Secure Collective Defense Network (SCOLD)”, http://cs.uccs.edu/~scold/doc/SCOLD_Final.doc

[61] Edward Chow and Yu Cai, “Algorithms for Selecting Multiple Mirror Sites for Parallel Download”, http://cs.uccs.edu/~scold/doc/mspd.doc

[62] Linux Virtual Server, “Virtual Server Scheduling Algorithms”, http://www.linuxvirtualserver.org/docs/scheduling.html

[63] Protocols.com, “TCP / IP Reference Page”, http://www.protocols.com/pbook/tcpip1.htm

[64] Julian Anastasov, “linux kernel patches”, http://www.ssi.bg/~ja/

[65] TCP/IP FAQ Maintainer, “TCP/IP FAQ”, http://www.faqs.org/faqs/internet/tcp-ip/tcp-ip-faq/part1/

[66] J. P. Gray and T. B. McNeill. SNA multiple-system networking. *IBM Systems Journal*, 18(2):263–297, 1979.

[67] Cisco, “Open System Interconnection Protocols”, http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/osi_prot.htm

[68] Eddie Runner, “Multi Path Interference”, http://www.mmxpress.com/technical/multipath.htm

[69] Srinivas Vutukury, “Multipath Routing Mechanisms For Traffic Engineering And Quality Of Service In The Internet”, Ph.D thesis, 2001

[70] IETF, “Multiprotocol Label Switching (mpls)”, http://www.ietf.org/html.charters/mpls-charter.html

[71] HAARP, “Antenna Array “, http://www.haarp.alaska.edu/haarp/ant3.html

[72] E. Yilmaz and Y. Manzano, "Surveying Formal and Practical Approaches for Optimal Placement of Replicas on the Web", http://websrv.cs.fsu.edu/research/reports/TR-020701.pdf

[73] L. Qiu, V. N. Padmanabhan, and G. M. Voelker, "On the placement of web server replicas," Proc. of IEEE INFOCOM, Mar. 2001

[74] S. Jamin, C. Jin, D. Raz, Y. Shavitt, and L. Zhang, "On the placement of internet instrumentation," Proc. of IEEE INFOCOM, Mar. 2000

[75] P. Krishnan, D. Raz, and Y. Shavitt, "The cache location problem," ACM/IEEE Transactions on Networking, vol. 8, no. 5, Oct. 2000.

[76] B. Li, M. J. Golin, G. F. Ialiano, and X. Deng, "On the optimal placement of web proxies in the internet," Proc. of IEEE INFOCOM, Mar. 1999.

[77] B. Krishnamurthy, and J. Wang, "On network-aware clustering of web clients," Proc. of ACM SIGCOMM, Aug. 2000.

[78] IPSec, "IPSec", http://www.ipsec-howto.org/x143.html

[79] VPN, VPN how-to, "http://www.tldp.org/HOWTO/VPN-HOWTO/"

[80] LVS, Linux Virtual Server, "http://www.linuxvirtualserver.org/"

[81] Jiong Yang, "Deliver Multimedia Streams with Flexible QoS via a Multicast DAG", 23rd International Conference on Distributed Computing Systems, 2003

Appendix A

Algorithms for Selecting Multiple Mirror Sites for Parallel Download

C. Edward Chow and Yu Cai

Department of Computer Science
University of Colorado at Colorado Springs

Colorado Springs, CO 80933-7150, USA

{chow, ycai}@cs.uccs.edu
Tel: (719)262-3110
FAX: (719)262-3369

Abstract

In this paper, we present a mathematical model that mimic the problem of parallel download from multiple mirror sites. Based on the model, we present algorithms for selecting the best subset of mirror sites for parallel download. The versions of brutal force algorithms and genetic algorithms are implemented. Performance of these algorithms on the simulated network topology as well as a real-world network topology is presented.

Keywords: Server selection, Parallel download, Genetic algorithms.

1. Introduction

With the recent development of internet, we are able to retrieve documents from multiple server sites, like the mirror sites, to increase the downloading speed, make better use of available network bandwidth and parallel processing speed of servers.

Recent work by Rodriguez, Kirpal, and Biersack [1] studied how to use the existing HTTP protocol for retrieving documents from mirror sites in parallel to reduce the download time and to improve the reliability. The proposed approach utilizes the HTTP 1.1 byte range header to retrieve specific data in a mirror server site, which requires no changes on existing server and client settings.

However, choosing the best mirror sites is not a trivial task and a bad choice will give a poor performance. Testing data [1, 2] shows that the performance of a bad choice might be 10 times slower than the best choice.

The document delivery speed between a server and a client depends on the server load, the file retrieving speed at the server, the available bandwidth between the server and the client, and the client load. Given a client, the document delivery speed from a particular server can be estimated by downloading a common short document existing on all mirror servers. Application layer anycast was proposed for selecting one of the replicated web server [9].

By using networking measurement tools, like pathchar, cprobe, we can estimate the network bottleneck and available bandwidth [3]. Kevin Lai and Mary Baker of Stanford University improve accuracy of the bottleneck bandwidth estimation by using better filtering technique in dynamic environment [4]. However, the accuracy of current network measurement methods still needed to be improved.

In this paper, we investigate two problems: one deals with finding the maximum parallel download speed without restricting the number of mirror servers. We call it pds problem. The other deals with finding the best group of k servers for parallel download. We called it k-pds problem. It is assumed that the network topology, the path bandwidth and server performance are known and static. Two versions of brutal force algorithm, as well as two versions of genetic algorithm, were implemented.

GT-ITM (Georgia Tech Internetwork Topology Models), which is one of the most commonly used internet topology models [6], is used to generate network topologies of varying sizes for evaluating their impact on the performance of the algorithms.
This paper is structured as follows. Section 2 defines the problems. Section 3 presents the algorithms for selecting multiple mirror servers for parallel download. Section 4 discusses the performance results. Section 5 is the conclusion.

2. Selecting Multiple Mirror Sites for Parallel Download

Assume that the network topology, the path bandwidth and server performance are static, the path between two end nodes will be fixed. The routing path map between a client and a set of mirror servers can then be modeled as a tree. We also assume that the documents to be retrieved are available on all mirror sites.

2.1. Network Model

Let G=(V, E) be a graph models the network topology, where V represents the set of nodes including those of the mirror server nodes, the clients, and the routers between the clients and the mirror server nodes; E represents the set of edges or link segments that connect the nodes in the network. Let M be the set of mirror servers. For a mirror server m, speed(m) represents the document retrieval and transfer speed of m. For an edge e, speed(e) represents the available bandwidth on edge e.
Given a client c and a set of mirror server M in G, there exists a set of shortest paths, P, from each of the mirror servers to the client c. Let path(c,m) be the path from mirror server m to the client c. Each path p in P, consists of a set of edges, $e_1,\ldots,e_n$. where n is the number of edges, or link segments in p. Without other traffic, the transfer speed over a path p in P, speed(p), is decided by the link segment with the smallest available bandwidth:
$\text{Speed}(p)=\min\{\text{speed}(e_1),\ldots,\text{speed}(e_n)\}$.
Without other traffic, the end-to-end download speed for selecting a mirror server, m, can be defined as endSpeed(c, m) and
$\text{endSpeed}(c,m)=\min\{\text{speed}(m),\text{speed}(\text{path}(c,m))\}$.
Let S be the subset of mirror servers selected for parallel download and n be the number of mirror servers in S. $S=\{S_1, S_2,\ldots,S_n\}$. Let pds(c, S) be the parallel download speed of using mirror server set S to client c. Note that if all paths from the mirror server in S to client c do not share any link segment. Then

$$\text{pds}(c, S)=\sum_{i=1}^{n} endSpeed(c, S_i)\ .$$

Unfortunately, some of these mirror servers may share some link segments to the client c. The actual download speed from a set of mirror servers may then be limited by the available bandwidth of the shared link segments.
Note that the set of link segments of path set P forms a tree. See Figure 1. The leaf nodes of the tree, S_1, S_2, S_3, and S_4, are mirror server nodes with 30, 25, 20, 9 as their document retrieval and transfer speed. R_1, R_2, and R_3 are the router nodes and root node of the tree is the client c. The edge label represents the available bandwidth of the edge. For the non-leaf node r, let mds(r, S) be the maximum download speed at node r using mirror server set S.

$$\text{mds}(r,S)=\sum_{i=1}^{ntree(r)}\min(speed(e_i), mds(r_i, S)) \quad (1)$$

where ntree(r) is the number of subtrees underneath r, and e_i is the edge that connects r to the subtree r_i. It is a recursive function. mds(m,S)=speed(m) if m is a mirror server node in S. mds(r, S)=pds(c, S) if r is the client node.

2.2. Problems to be solved:

There are several problems related to parallel download from a set of mirror servers. The first problem is the pds problem: given a network topology G, a client c, a set of mirror servers S, find the maximum download speed pds(c, S). The related question is how many mirror sites are needed to achieve the global max speed and how to choose the mirror sites? We can solve these problems with formula (1) above.

We have implemented a parallel download program similar to that in reference [1], and observed that the performance peaks typically at 5 or 6 Redhat mirror servers for a Linux machine at UCCS. Part of the reason is the re-assembly overhead. Therefore if we only want to choose a certain number of mirror sites, say 5 sites, what is the maximum download speed for 5 mirror sites? And a related question is which 5 sites to choose for achieving this speed? This is referred to as k-pds problem.

Let k-pds(c,S) be the maximum parallel download speed for the client c to use k of the mirror server set S, and k-pdsSubset(c, S) be the optimal subset of k servers with the maximum parallel download speed.

$$\begin{cases} \text{k-pds(c,S)=max\{mds(c, S')|S'} \in \text{S, S' has k nodes\}} \\ \text{mds(c, S')=} \sum_{i=1}^{ntree(r)} \min(speed(e_i), mds(r_i, S')) \quad (2) \end{cases}$$

In Figure 1, we give an example using formula (1) and (2).

mds(R2,S)=min(mds(S1,S),5)+min(mds(S2,S),8)=min(30,5)+min(25,8)=13, similarly, mds(R3, S)=16,

mds(R1,S)=min(mds(R2,S),40)+min(mds(R3,S),30)=min(13,40)+min(16,30)=29,

pds(c, S)=mds(c, S)=mds(R1, S),

3-pds(c,S)=max{mds(c,{S1,S2,S3}),mds(c,{S1,S2,S4}) ,mds(c,{S1,S3,S4}),mds(c,{S2,S3,S4})}=max{20,22,21,24}=24, and the subset of mirror servers to use 3-pdsSubset(c,S)={S_2,S_3,S_4}. Similarly, 2-pds(c,S)=17 and 2-pdsSubset(c, S)={S_2,S_4}.

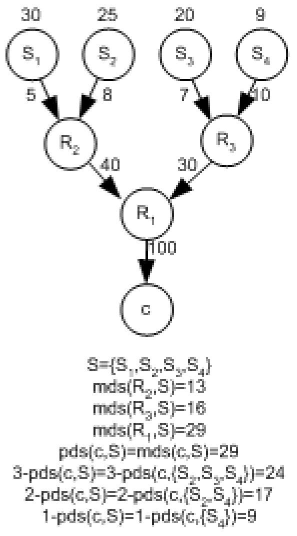


Figure 1. Parallel Download Tree and related download speed values.

3. Algorithm Implementation

Versions of brutal force algorithms and genetic algorithms were developed to solve the problem.

a) Brutal Force Algorithm for pds:

We use the algorithm to find the maximum parallel download speed pds(c, S). We refer to it as BF-pds algorithm. It implements formula (1) in Section 2.1. The complexity of this algorithm in worst case scenario is O(n), where n is the number of nodes in the network topology.

This algorithm is quick, but we have no control over how many mirror sites and which mirror sites to be chosen. In practice, we use it to find the upper bound of the maximum parallel download speed in network topology.

b) Brutal Force Algorithm for k-pds:

We use the algorithm to find the maximum download speed for the client c to use k of the mirror server set S, which is k-pds(c, S). We refer to it as BF-k-pds algorithm. It implements formula (2) in Section 2.2. The complexity of this algorithm in worse case scenario is $O(n^k)$, where n is the number of nodes, and k is the number of servers we want to choose.

c) Genetic Algorithm

We implement a fix-length genetic algorithm and a variable-length genetic algorithm.

The fix-length algorithm is used to find the k-pds(c,S) speed, the length of chromosomes is k and fixed. We refer to it as GA-k-pds algorithm.
The variable-length algorithm is used to find the pds(c, S), the length of chromosomes is smaller than a given number, and can be changed. We refer to it as GA-pds algorithm.
Formula (1) and (2) in Section 2 are recursive functions, therefore it is difficult to implement them in genetic algorithm. We used a revised version in genetic algorithm by scanning through all the server nodes to client node.
The variable-length genetic algorithm works as follow:

1) Assign the sequential server number, node number and path number to denote each server, node and path. Assign the initial bandwidth and server speed.
2) Initialize the first generation of chromosomes with random length by filling server number in chromosome.
3) Crossover and mutation at certain probability.
   Make sure no duplicated server in chromosome, and the length of chromosome is less than the given number. Several different crossover and mutation methods have been combined together for better performance [8].
4) Fitness function. For a given chromosome S', use the max download speed mds(c, S') as fitness function.
5) Run certain generations, and output the result.

Genetic algorithm provides more control and flexibility over the server selection. For example, if there are multiple selections of mirror servers, and all selections achieve the max download speed pds(c, S) or k-pds(c, S), how to choose the best one from all these selections? We can easily implement the selection criteria in genetic algorithm.

4. Testing Results

We tested the algorithms on simulated network topologies as well as a real-world network topology
Figure 2 is a sample routing tree with 20 nodes and 10 mirror sites, starting from a machine in Eurecom, to the mirror sites for Squid document [1]. Below is the testing result:

20 nodes, 10 mirror sites	
BF-pds	0.6 s
BF-k-pds	10 s
GA-k-pds	1.5 s
GA-pds	1.5 s

(s: second)

Figure 3 is a sample routing tree with 114 nodes and 11 mirror sites, starting from a machine in UCCS, to the mirror sites of Redhat [5]. Below is the testing result

114 nodes, 11 mirror sites	
BF-pds	0.7 s
BF-k-pds	2 m
GA-k-pds	2 s
GA-pds	2 s

(s: second, m: minute)

Figure 4 is a sample transit-stub hierarchical network topology derived from GT-ITM [7], we can derive routing tree structure from the network topology, by assuming the route from node to node is always the shortest route in terms of network bandwidth. Below is the test result on GT-ITM simulated network:

	BF-pds	BF-k-pds	GA-k-pds	GA-pds
150 nodes, 20 mirror sites	0.8 s	2 m	2 s	2 s
200 n, 20 m	0.8 s	2.5 m	2 s	2 s
300 n, 30 m	0.9 s	3 m	2 s	2 s
500 n, 50 m	0.9s	7 m	5 s	5 s
800 n, 100 m	1 s	12 m	6 s	6 s
1000 n, 100 m	1 s	20 m	7 s	7 s
1000 n, 200 m	1 s	30 m	8 s	8 s

(s: second, m: minute)

Figure 5 is a chart of algorithm execution time vs. simulated network size. BF-k-pds algorithm execution time uses primary y axis (on the left), and is measured by minutes. BF-pds, GA-pds and GA-k-pds algorithm execution time use secondary y axis (on the right), and are measured by seconds.

5. Conclusion

We discuss the related problems of selecting subset of mirror servers for parallel download. Given a network topology, a client and a set of mirror servers, we presented brutal force algorithms and genetic algorithms for finding the maximum parallel download speed pds(c, S) and for finding the subset of mirror server with maximum parallel download speeding k-pds(c, S), given the size of the subset as k.
Further work is needed to investigate how effective are the above algorithms in selecting the right subset of mirror servers for parallel download.

6. Reference

1) Pablo Rodriguez Andreas Kirpal Ernst W. Biersack, "Parallel-Access for Mirror Sites in the Internet", Proceeding of Infocom, 2000.

http://www.ieee-infocom.org/2000/papers/65.ps

2) Ratul Mahajan, "Aggregate Based Congestion: Detection and Control", April 2001. University of Washington.
3) Vern Paxson, "Measurements and Analysis of End-to-End Internet Dynamics ", Ph.D. dissertation at UC Berkley.
4) Kevin Lai and Mary Baker, "Nettimer: A Tool for Measuring Bottleneck Link Bandwidth", Proceedings of the USENIX Symposium on Internet Technologies and Systems, March 2001.
5) Jing Yang and Zhong Li, "Selecting best Redhat Mirror Sites for parallel download", http://cs.uccs.edu/~cs522/proj2001/jyang.ppt
6) Ellen W. Zegura, "GT-ITM: Georgia Tech Internetwork Topology Models", http://www.cc.gatech.edu/projects/gtitm/
7) Thierry Ernst, "Existing NS-2 Presentation: GT-ITM. Topologies",http://www.inrialpes.fr/planete/pub/mobiwan/Documents/ernst-ns-mobiwan-0501.ppt
8) John R. Koza, "Genetic Programming", MIT Press, 1992.
9) Ellen W. Zegura, Mostafa H. Ammar, Zongming Fei, and Samrat Bhattacharjee, "Application-Layer Anycasting: A Server Selection Architecture and Use in a Replicated Web Service," IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 8, NO. 4, AUGUST 2000, pp. 455-466.

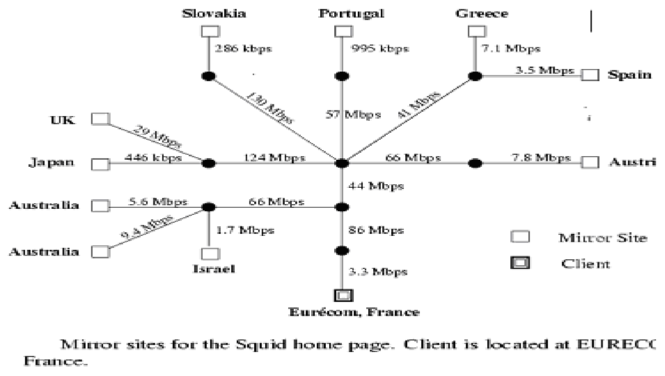


Figure 2. A sample routing tree with 20 nodes and 10 mirror sites in [1].

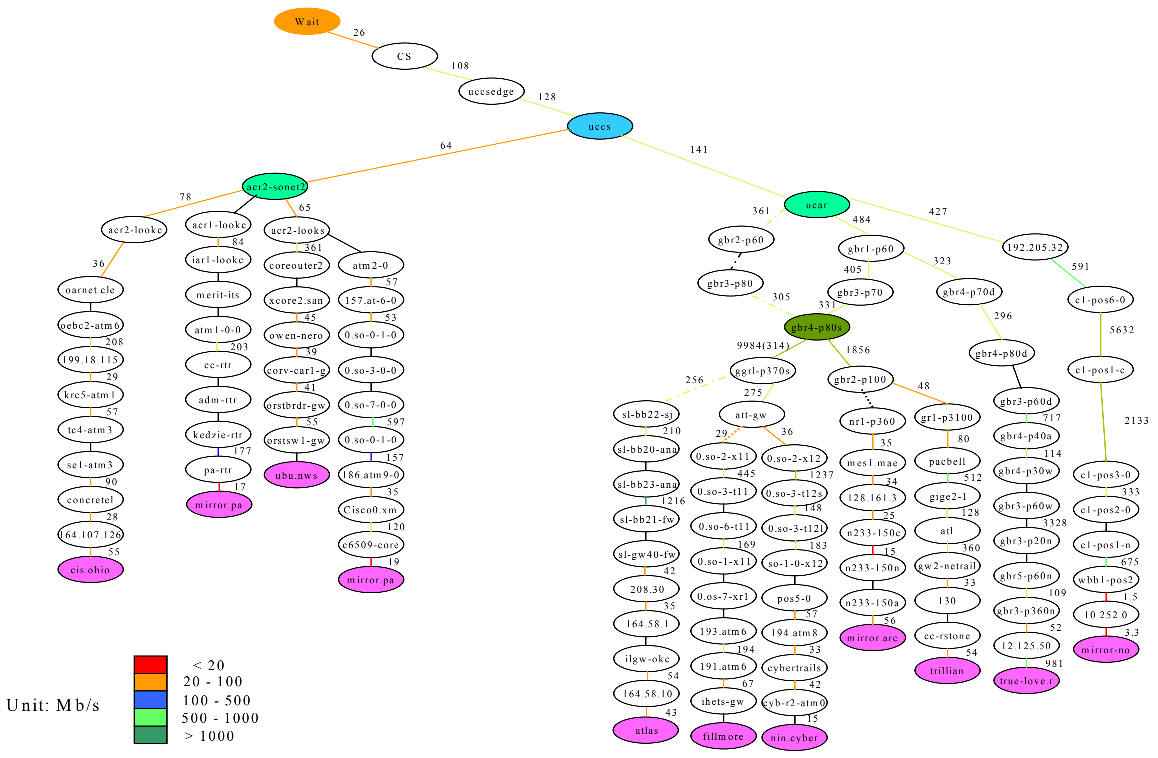


Figure 3. Network Topology of a UCCS client to a subset of Redhat mirror servers.

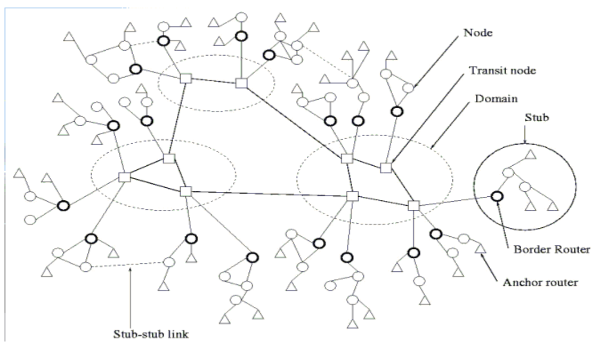


Figure 4. A sample transit-stub hierarchical network topology generate in GIT-ITM [6].

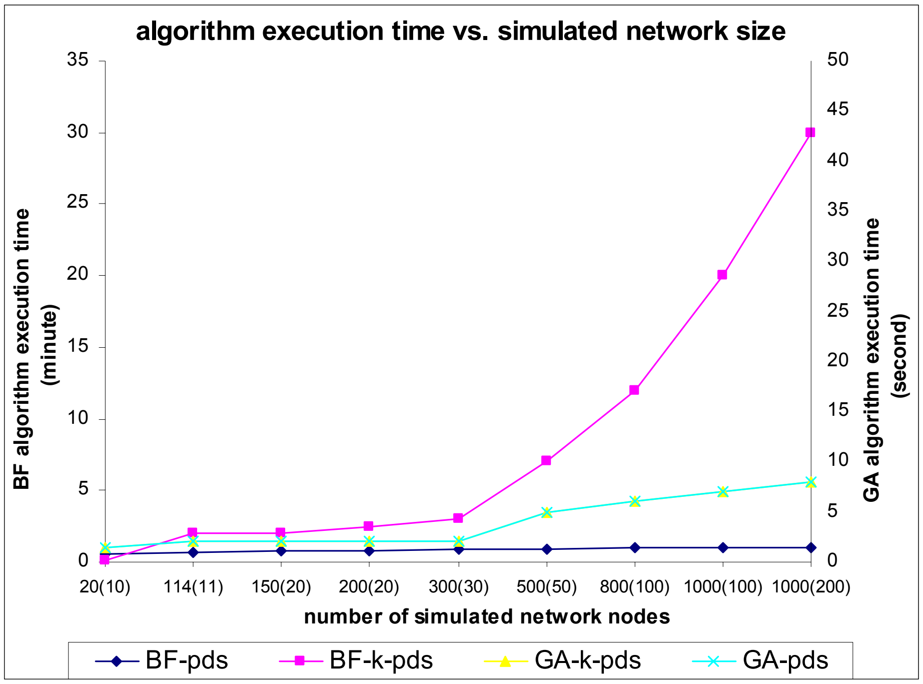


Figure 5: Algorithm execution time vs. simulated network size.
BF-k-pds algorithm execution time is measured by minutes.
BF-pds, GA-pds and GA-k-pds algorithm execution time are measured by seconds.

Appendix B

Secure Collective Defense (SCOLD) Network

C. Edward Chow
Yu Cai
David Wilkinson

Abstract: The increasing network attacks reveal one of the fundamental security problems of today's Internet. Many Internet services, such as DNS and routing protocols, were not originally designed with security as one of the basic requirements. It is difficult to modify the existing protocols or network architecture without significant work. At the same time, it offers an opportunity to create new, secure and reliable network protocols, and packet delivery systems. We present, in this paper, a prototype of the Secure Collective Defense (SCOLD) system that utilizes collective resources from participation organizations, tighten coordination and new cyber security defense techniques against Distributed Denial of Services (DDoS) attack. SCOLD tolerates DDoS attacks with alternate routes via a set of proxy servers with intrusion detection, and secure Domain Name System (DNS) updates. The research results and insights obtained from this project can improve the security of the networks and have a broader impact on the network architecture and the client side network software interface.

Index Terms: Intrusion Tolerance, DDoS, Secure Collective Defense, Alternate Route, Secure DNS update.

1. Introduction

The brief service disruption on the nine of the thirteen DNS root servers caused by DDoS bandwidth attacks on October 2002 [1] is one of the most prominent attacks on DNS recently. The increasing frequency and severity of network attacks reveal one of the fundamental security problems of today's Internet. Many Internet services, such as DNS and routing protocols, were not originally designed with security as one of the basic requirements. Therefore, the existing network architecture needs to be strengthened and protocols need to be enhanced or re-designed with security as the basic consideration.

A study conducted by the University of California, San Diego, detected approximately 12,805 Denial of Service attacks against more than 5000 targets during a three-week period in mid-2001 [2]. Even CERT, the authority that warns Internet users on security threats, fell victim to DDoS in May 2001 [2]. In a recent survey conducted by the SANS Institute on ``How to Eliminate the Ten Most Critical Internet Security Threats", the number-one Internet vulnerability reported by survey participants was BIND weaknesses [3]. BIND is the open-source software package that powers the majority of Internet DNS servers [18].

The general objective of the Secure Collective Defense (SCOLD) project is to create a secure collective Internet defense system that utilizes collective resources from participation organizations, tighten coordination and new cyber security defense techniques. SCOLD will tolerate Distributed Denial of Services (DDoS) attacks with alternate routes via a set of proxy servers with intrusion detection, and secure Domain Name System (DNS) updates.

Most of the organizations have multiple gateways and can deploy multi-homing schemes using alternate gateways when the main gateway is attacked. But once the alternate gateway's IP address is revealed, it is also subject to DDoS attacks. We propose to explore the use of protected indirect routes over a collection of geographical separated proxy servers to those alternate gateways and hide the IP address of those alternate gateways from the clients. The clients or client site DNS servers will be informed of the indirect routes through secure DNS updates. The new DNS entries will contain records like (Domain name, IP address, a set of IP addresses of designated proxy servers). Client requests will be sent to one of the selected proxy servers or spread over the set of designated proxy servers for better performance with aggregate bandwidth. The proxy server will be enhanced with integrated Intrusion Detection System (IDS) and a firewall filter to block intrusion traffic that may try to come in through the indirect route. The detection of intrusion on those proxy servers provides additional identification and isolation of the source of the attacks. The collection of proxy servers can be provided by participating organizations of a consortium, or branches of an organization. The proxy servers know the IP addresses of an alternate gateway of the target site and relay the packets from the clients over an IP tunnel to the target site. An indirect route can then be set up from the clients to the attacked target through designated proxy servers and alternate gateway.

This new set of network protocols and infrastructure can be used to defend against generic DDoS attacks as well as protect the root DNS servers from DDoS attacks.

1.2 Related Work

Recent work by Angela Cearns from University of Colorado, Colorado Springs, implemented an Autonomous Anti-DDoS network (A2D2) with enhanced SNORT IDS for detecting subnet spoof attacks and with adaptive rate limiting and Class Based Queuing (CBQ) firewall rules for effective intrusion handling [4]. Steven Cheung in University of California at Davis, introduced a wrapper-based solution to protect DNS, and a detection-based message authentication scheme to protect routers [5].

Network Associates Labs and Boeing developed the Intrusion Detection and Isolation Protocol (IDIP) to support real-time tracking and containment of attacks that cross network boundaries [7]. Service Location Protocol (SLP) is an IETF protocol that provides automatic client configuration for applications and advertisement for network services, i.e. locating IDIP nodes [8]. We are going to incorporate IDIP and SLP in future SCOLD system.

J. Mirkovic, J. Martin and P. Reiher from University of California at Los Angeles provided a compressive taxonomy of DDoS attacks and DDoS Defense Mechanisms [6]. According to the classification of DDoS defense mechanisms in the taxonomy, the SCOLD falls into the category of reactive, reconfiguration and cooperative. Reactive mechanism strives to alleviate the impact of the DDoS attack on the victim instead of eliminating the attacks. Reconfiguration mechanism changes the topology of the victim or the intermediate network. Cooperative mechanism is achieved through cooperation with other entities. Related works in Reconfiguration mechanism include reconfigurable overlay networks ([21], [22]), resource replication services [23], attack isolation strategies ([24], [25]), etc. These works focused on either adding more resources to the victim or isolating the attack machines. But the SCOLD system manages to redirect client traffic through a set of proxy servers by indirect route under DDoS attacks. Cooperative mechanism is limited by the highly independent nature of Internet. The SCOLD system tries to utilize collective resources from participation organizations with tighten coordination and cooperation.

The balance of this paper is organized as follows. In Section 2, we introduce the design of SCOLD system. In Section 3, we present the design of secure DNS update. In Section 4, we present the design of indirect route through IP tunneling. Test and performance results are presented in Section 5. Our conclusions and future works are in Section 6.

2. SCOLD Design

2.1 SCOLD Overview

There are two basic approaches to defend against DDoS attacks: one is called *intrusion blocking/tracking* that actively tracks down and blocks the traffic generated from those infected machines, and identifies the mastermind intruder; the other is called *intrusion tolerance* that studies how to tolerate intrusion and provides alternate routes for legitimate clients to access the target site.

The SCOLD system deals with mainly the latter approach by designing new software modules and protocols for providing such alternate routes. But by dividing the clients to come in through different proxy servers with intrusion detection devices, the SCOLD system can also provide useful information for tracking down the intruder.

One of the critical components in the SCOLD system is the shared usage of a collection of geographically distributed proxy servers, either provided by a service provider or contributed by each participating organization of a consortium.

When a site is attacked, its intrusion detection system will generate the security alarms and send a request to the SCOLD coordinator for setting up indirect route. The coordinator then updates the DNS servers of the client sites through secure DNS updates. The clients get the IP addresses of the proxy servers, and send packets through the designated proxy servers. The designated proxy server knows the IP addresses of an alternate gateway of the target site and relays the packets from the clients over an IP tunnel to the target site. The designated proxy server is integrated with the intrusion detection system to detect and block potential DDoS attacks on these alternate gateways.

The secure DNS update utilizes the Secure Socket Layer protocol for authentication and encryption. The existing DNS servers need to be modified to save the new cache entries with the domain name and IP address of the target machine, and the IP address of the designate proxy servers. The hostname resolving library on a client machine needs to be modified to accept the new type of DNS query results. For the exploratory prototype, we propose to modify the popular BIND 9 DNS software package from Internet Software Consortium [9, 10].

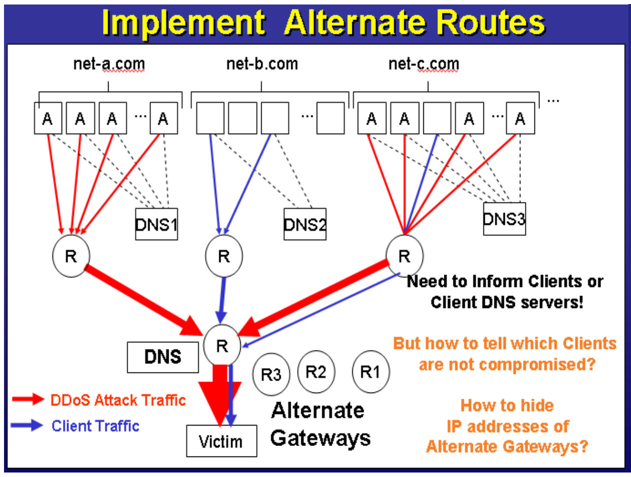

Implement Alternate Routes
net-a.com
net-b.com
net-c.com
A
DNS1
DNS2
DNS3
R
DNS
Victim
R3
R2
R1
Alternate Gateways
Need to Inform Clients or Client DNS servers!
But how to tell which Clients are not compromised?
How to hide IP addresses of Alternate Gateways?
DDoS Attack Traffic
Client Traffic

Figure 1: DDoS attack without alternate routes [11]

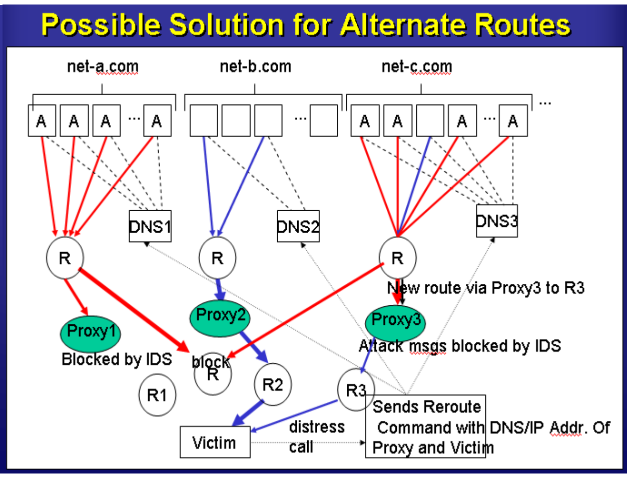

Possible Solution for Alternate Routes
net-a.com
net-b.com
net-c.com
A
DNS1
DNS2
DNS3
R
Proxy1
Proxy2
Proxy3
Blocked by IDS
block
New route via Proxy3 to R3
Attack msgs blocked by IDS
R1
R2
R3
Victim
distress call
Sends Reroute Command with DNS/IP Addr. Of Proxy and Victim

Figure 2: DDoS attack with alternate routes [11]

Figure 1 shows the target under DDoS attack, without the implementation of alternate routes. As a consequence, the bandwidth of legitimate clients is greatly reduced. Figure 2 shows the target under DDoS attack, with the implementation of alternate routes. All users will get updated alternate DNS entry information. But the attack network will be blocked at proxy servers, and the legitimate users will be re-directed to alternate gateway, then to the final destination.

2.2 Design Considerations

A. The need to hide alternate gateway

Organizations that have multiple gateways can deploy alternate gateways when the main gateway is attacked. But once the alternate gateway's IP address is revealed on internet, it is also subject to DDoS attacks. We can use protected indirect routes over a collection of proxy servers to those alternate gateways. Therefore, the proxy servers will act as the frontline against DDoS attacks, and the IP address of the alternate gateways were hide.

B. The need for indirect routing

The normal route from the client to the intended target will be severely impacted by DDoS attack. Therefore, we need to use indirect routing to route the client traffic pass through the proxy server, then the alternate gateway, and finally the target. One way for indirect routing is using IP tunneling.

C. The need for secure DNS update and secure connection

The dynamic update of DNS record needs to be secure and authenticated. Otherwise the malicious clients can send fake DNS record update and redirect the traffic of legitimate clients.

For the similar reason, certain connections between proxy servers, alternate gateways, target and coordinators need to be secure and mutually authenticated. Because of the overhead of secure connection, we need to keep the minimum usage of secure connection.

D. The advantage of using a set of proxy servers.

Using a set of proxy servers can avoid vulnerable bottleneck points in the intermediate network. If a selected proxy server gets severe DDoS attacks, the SCOLD can inform the clients to use other available proxy servers, and reset the indirect route.

A collection of geographical separated proxy servers also provide the possibility to spread the load and client request, and overcome the overhead associated with indirect routing and secure DNS update.

3. Secure DNS update

We have extend Bind9 v.9.2.2 DNS server software package, which is written and maintained by the Internet Software Consortium [18], to enable secure DNS update and alternate proxy IP address, by modifying the nsreroute command, and an add-on to the BIND9 DNS software.

In the rest of the paper, we use "Client" specifically refer to a client machine in the SCOLD system in Firgue 3. Same for "Proxy", "Gateway", "Target", "Coordinator", "ClientDNS" and "TargetDNS".

The Steps for secure DNS update are as follows (Figure 3):
1. The Target gets DDoS attack, it will raise an alarm and notify the Coordinator to issue a command for secure DNS update.
2. The Coordinator has a list of available proxy servers IP addresses for the Target. The Coordinator sends message to the TargetDNS, updates its DNS records with Proxy IP addresses as ALT data type. 3. The Coordinator sends message to the Proxy.
4. The Proxy sends message to the ClientDNS to notify that an indirect route is ready to be set up.
5. The Client queries the ClientDNS, and the ClientDNS queries the TargetDNS to get DNS record. (The DNS record alive time is set to be very short to enable dynamic DNS update).
6. If the route from the ClientDNS to the TargetDNS is not severely affected by DDoS attack, the Client DNS will talk to the TargetDNS through normal Internet route and fetch the updated DNS records. Otherwise the ClientDNS needs to setup indirect route to the TargetDNS through the Proxy and the Gateway as described in Section 4.
7. Restore normal DNS record. Once the DDoS attacks stop, the Target will notify the Coordinator and ask it to issue commands to restore normal DNS record with similar steps in steps (1-6).

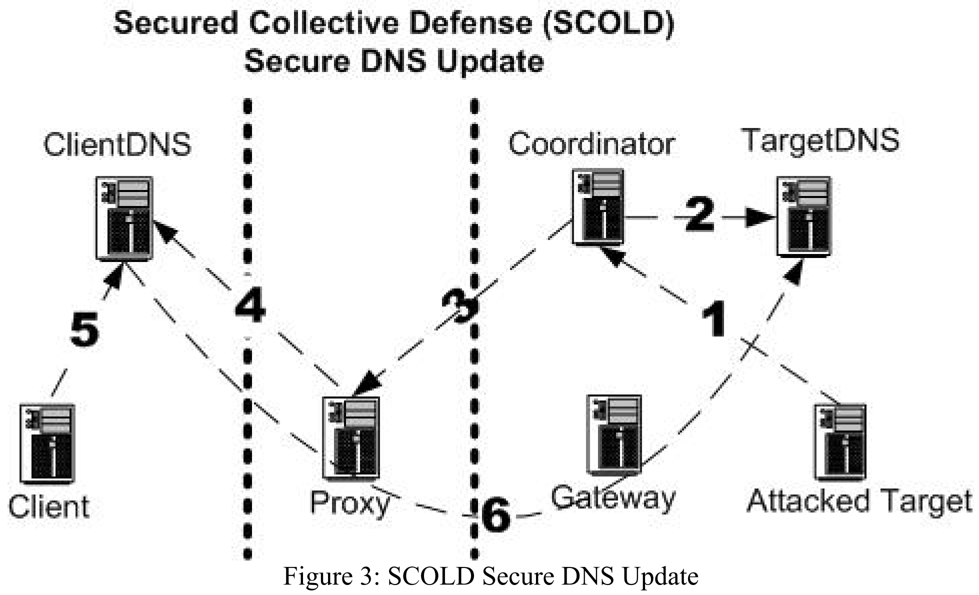


Figure 3: SCOLD Secure DNS Update

This updated zone file with alternate proxy IP addresses will look like the following:

```
target.targetnet.com. 10 IN A 133.41.96.71
target.targetnet.com. 10 IN ALT 203.55.57.102
                      10 IN ALT 203.55.57.103
                      10 IN ALT 185.11.16.49
                      10 IN ALT 221.46.56.38
```

After the secure DNS update, the client will get a DNS record with multiple proxy server IP addresses as ALT type in addition to normal domain name and IP address. An indirect route can then be setup from client machine through the selected proxy server.

We use OpenSSL to authenticate and encrypt the control message communicated between DNS server and the SCOLD coordinator. [12]

4. **Indirect Route**

We have investigated several alternatives for implementing indirect routing. The alternatives include Zebedee[15], SOCKS proxy server[16], and IP tunnel[17].

IP tunnel (also called IP encapsulation) is a technique to encapsulate IP datagram within IP datagrams. This allows datagrams destined for one IP address to be wrapped and redirected to another IP address. The IP tunnel can be set up from Linux to Linux, windows to windows, or between Linux and windows (windows must be Windows 2000 server and above).

The advantage of using IP tunnel is that it is a layer three protocol; therefore all the upper layer protocols and applications can use it transparently. Second, IP tunnel is installed with Redhat Linux 8 and 9 by default, and setting up IP tunnel doesn't require additional software installation or kernel compilation. The other advantage of IP Tunnel is that IP tunnel is essentially a device descriptor, it consumes limited system resources on the server, therefore the number of IP tunnels available is primitively limited by server resources.

Even through IP Tunnel supports any upper layer transport protocols, it increases overhead by an extra set of IP headers. Typically this is 20 bytes per packet, so if the normal packet size (MTU) on a network is 1500 bytes, a packet that is sent through a tunnel can only be 1480 bytes big, therefore the payload size is reduced. This also causes fragmentation and reassembly overhead. But these overheads can be reduced or avoided by setting smaller MTU at the client side.

4.1 Indirect Route by Modifying Client's Resolve Library

The resolve library is a shared library located usually in /usr/lib or /lib directory in Redhat Linux, named as libresolv-nnn.so (nnn is the version). Also there are a couple of soft link to libresolv-nnn.so which named as libresolv-nnn.so.1, libresolv-nnn.so.2 in the same directory. The resolve library is a dynamically loadable library which is usually called when client queries the target by domain name. The resolve library will then query the DNS and return the corresponding IP address of target.

The resolve library is usually distributed as part of the glibc package [13]. Since resolve library is part of the system library, be extremely careful when modify, compile and install the library. A small error can easily make the system unstable or even crash it. We modified one of the resolve library routines named res_query() to enable indirect route. The resolve library we modified is version 2.3.2, and is tested on Linux Redhat 8 and 9.

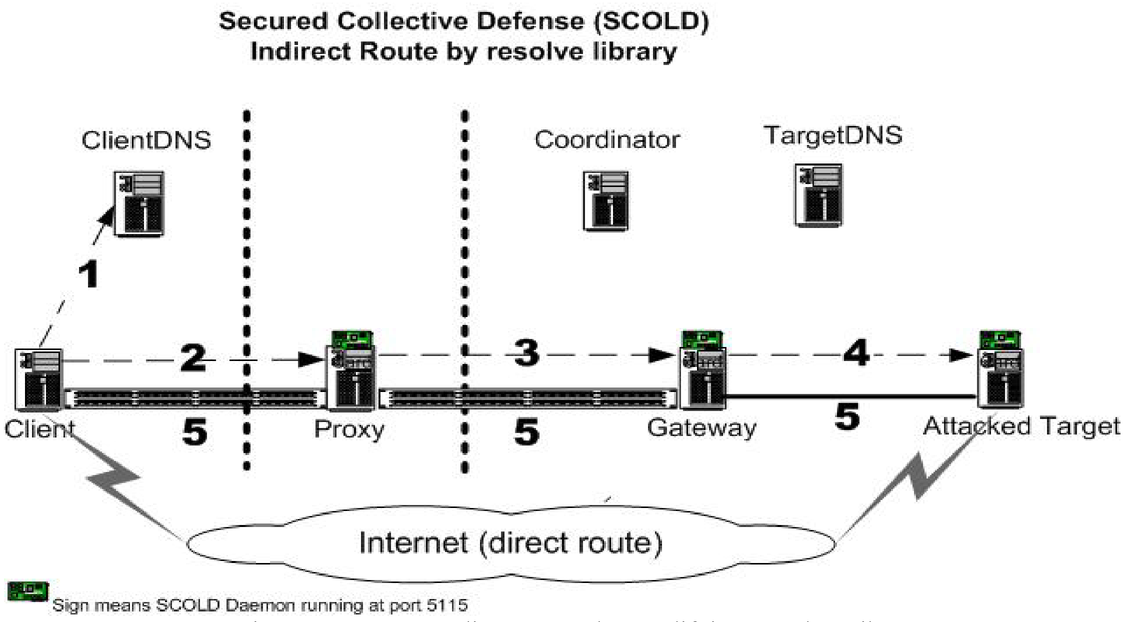

Secured Collective Defense (SCOLD)
Indirect Route by resolve library
ClientDNS
Coordinator
TargetDNS
1
2
3
4
5
Client
Proxy
Gateway
Attacked Target
Internet (direct route)
Sign means SCOLD Daemon running at port 5115

Figure 4: SCOLD Indirect Route by Modifying Resolve Library

The steps of setting up indirect route using resolve library are as follows (Figure 4):
1. The Client queries the Target by hostname, and the resolve library on the Client will query the ClientDNS for name resolution. The Client will get updated DNS information with proxy IP addresses beside normal Target IP address.
2. In resolve library, the Client will choose one of the Proxy IP addresses, send a message to that Proxy, and setup IP tunnel from the Client to the Proxy. If the Proxy fails to response, the Client will choose the next available Proxy IP.
3. The Proxy will choose one of the alternate gateways, send a message to that gateway; also, the Proxy will re-set its routing table and firewall rules, setup the IP tunnels to Client and to Gateway.
4. The Gateway will send a message to the Target, re-set the routing table and firewall rules on the Gateway, and setup the IP tunnel to Proxy. After getting the message from the Gateway, the Target will re-set its routing table and firewall rules to enable indirect route.
5. The indirect route is set up. The Client can access the Target through indirect route.
6. Clean up the indirect route: When the DDoS attacks stop, through the secure DNS update, the clientDNS will get normal IP/hostname without proxy IP. Once the Client query the Target by hostname, the Client will find out there is no proxy IP now. Therefore, in resolve library, the Client will find out the existing IP tunnel is not needed anymore, and will issue commands to clean up the IP tunnels, restore the routing table and firewall rules, with similar step ins (1 - 5).

In this schema, only the proxy servers are exposed to the clients. The proxy servers are equipped with DDoS IDS, like Snort, and function as the frontline fighting against possible DDoS attacks from malicious clients. We don’t want to expose the Gateway or the Coordinator to the Client. Therefore the malicious clients can not find the Gateway or the Coordinator, and launch DDoS attacks against them.

In this schema, the indirect route needs to be set up at run time. Pre-setup tunnels might not fit the needs of the client because the chosen proxy server and the chosen gateway are not available until run time.

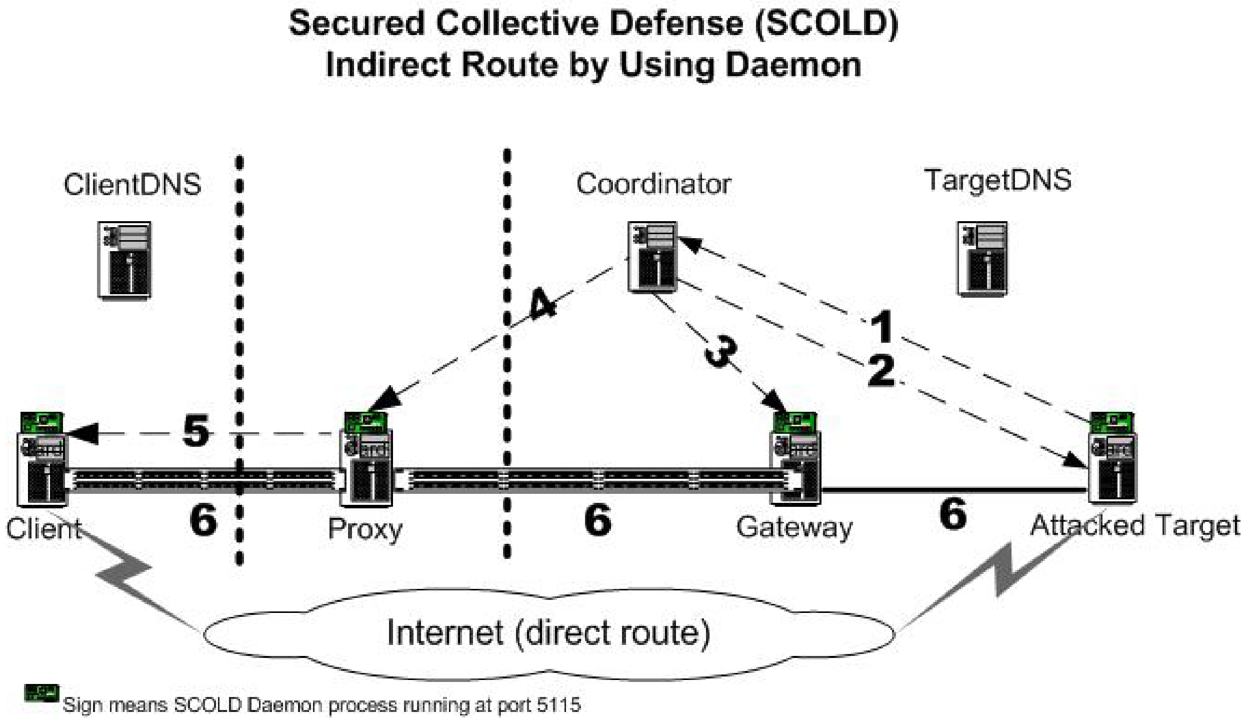
Advantage and disadvantage of this schema are as follows:
1) Once IP tunnel is set up, all the upper layer applications and protocols, like http, ftp, ICMP can use IP tunnel transparently.
2) The number of IP tunnels that we can set up is only limited by system resources on server.
3) There is overhead associated with IP tunnel. Usually the response time increase by 100% - 300%. But compared with the impact of DDoS attack, the overhead is still acceptable.
4) The client side needs to use the modified resolve library, which can be easily distributed through system update. The drawback is that the resolve library will only be called when client queries target by hostname. If client queries target by IP address, then the resolve library won't be called.
5) Needs to update DNS server and Bind software to support ALT type address.
6) Needs a set of participating proxy servers. If one proxy down, we can use other available proxy servers

to enable indirect route.
7) The control messages communicated between SCOLD Daemon on the Proxy, Gateway and Target are SSL encrypted and mutually authenticated. There is overhead associated with that.

One interesting problem in this schema is the coordination between the Client and Proxy. After the Client sends control message to the Proxy and get confirmation message back, the Client will set up an IP tunnel towards the Proxy. At same time, the Proxy will also set up IP tunnel towards the Client. If for some reason, the Proxy fails to set up the IP tunnel, then the two-way communication between the Client and the Proxy is broken. Currently we use timeout on client side to solve this problem. If time out, the Client will pick another proxy severs and set up indirect route.

4.2. Indirect Route with Client Running SCOLD Daemon

The other way to implement indirect route of IP tunnel is to let a daemon server process running on the Client, Proxy, Gateway, Target and Coordinator machine, listening to a certain port, waiting for control message, setting up IP tunnels and setting routing tables / firewall rules accordingly. The control message communicated between SCOLD Daemon on the Coordinator, Proxy, Gateway and Target is SSL encrypted and mutually authenticated.


Secured Collective Defense (SCOLD)
Indirect Route by Using Daemon
ClientDNS
Coordinator
TargetDNS
1
2
3
4
5
6
6
6
Client
Proxy
Gateway
Attacked Target
Internet (direct route)
Sign means SCOLD Daemon process running at port 5115
Figure 5: SCOLD Indirect Route by Using Daemon

The steps of setting up indirect route using daemons are as follows (Figure 5):
1. The Target gets DDoS attacks, raises alarm and notifies the Coordinator.
2. The Coordinator sends message to the Target, asking the Target to re-set its routing table and firewall rules.
3. The Coordinator sends message to the Gateway, asking it to re-set its routing table and firewall rules, and set up an IP tunnel to proxy.
4. The Coordinator sends message to the Proxy, asking it to re-set its routing table and firewall rules, set up IP tunnels to the Gateway and to the Client.
5. The Proxy sends message to the Client, asking it to re-set its routing table and firewall rules, set up an IP tunnel to the Proxy. Upon getting the control message from the Proxy, the Client needs to verify the Proxy, and set up IP tunnels to the Proxy accordingly.
6. The indirect route is set up. The Client can access the Target through indirect route.
7. Clean up the indirect route: When the DDoS attacks stop, the Target will notify the Coordinator. The Coordinator will issue commands to clean up the IP tunnels, restore the routing table and firewall rules, with similar steps in (1 - 6).

When the Proxy sends message to the Client, the Proxy needs to present a certificate signed by certain root CA, like Verisign, or the Coordinator. The Client needs to verify the certificate and decide whether it is acceptable. The Client may also query the ClientDNS, get the proxy IP addresses, and verify the IP of the Proxy.

Compared with schema in 4.2, this schema requires a SCOLD daemon running on client side. This may give the coordinator or the proxy server more control over the client, but it might also raise security concerns on client. But in this schema, the clients can get the IP addresses of proxy servers directly from the proxy servers, secure DNS update is necessary.

4.3. Indirect Route by Using Proxy as NAT Server

In this schema, the proxy server will function like a NAT server. The DNS Records in the clientDNS contain pairs of IP Address / Hostname. The Hostname is still the target name, but the IP address will be the proxy IP instead of real target IP. Therefore the Client's traffic towards the Target will be route to the Proxy. The Proxy will do a NAT translation, set up indirect route, and forward to the Target through indirect route.

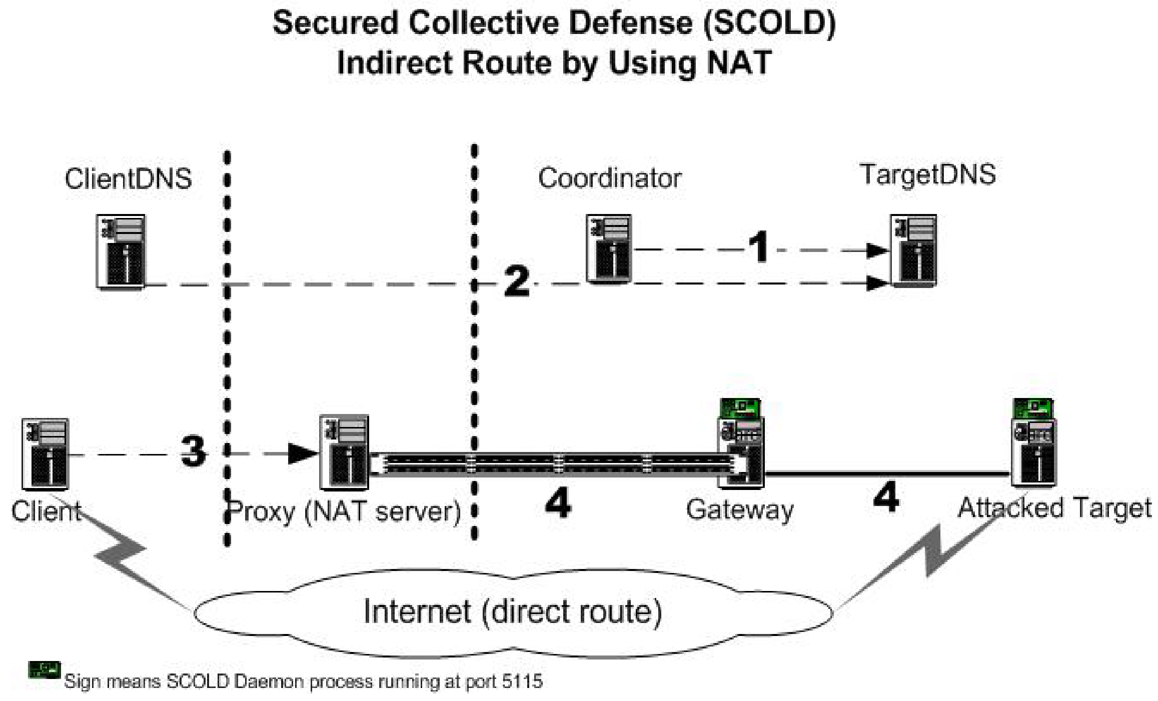

Secured Collective Defense (SCOLD)
Indirect Route by Using NAT
ClientDNS
Coordinator
TargetDNS
1
2
3
Client
Proxy (NAT server)
4
Gateway
4
Attacked Target
Internet (direct route)
Sign means SCOLD Daemon process running at port 5115

Figure 6: SCOLD Indirect Route by Using NAT

The steps are as follows (Figure 6)

1. The Coordinator notify the TargetDNS to update its DNS record by replacing the Target IP with the Proxy IP.
2. The Client queries the Target. The ClientDNS get updated DNS record from the TargetDNS.
3. The Client's traffic will be route to the Proxy.
4. The Proxy do a NAT translate, set up indirect route to the Target, and forward the client's traffic to the Target.

This schema requires no change on client side, and very little changes on ClientDNS to enable dynamic DNS update. But this schema has a scalable problem with large number of protected targets. Because it requires the proxy server to reserve a reasonable number of IP addresses, one to one corresponding to the Target. Also, the client will not get the real IP of target, and this might cause certain problems with some applications, like ssh, who cache the remote IP locally. Or it might raise a false alarm for IP spoofing if client runs certain internet security protection software or firewall.

4.4. Reset Indirect Route under severe DDoS attacks

The IP addresses of alternate gateways and coordinator are hidden from the clients, but the IP addresses of proxy servers are exposed to the clients. Therefore, the proxy servers are subject to the possible DDoS attacks.

The proxy servers are equipped with Intrusion Detection System (IDS). But under severe DDoS attacks, the selected proxy server might response slowly or even stop responding. In this case, the indirect route needs to be reset up and old indirect route needs to be cleaned up.

The steps of re-setting up indirect route are as follows (Figure 7):

1. There is an indirect route set up already, but the selected Proxy is under severe DDoS attacks.
2. The attacked Proxy server will notify the Coordinator to re-set up the indirect route.
3. The Coordinator will notify the ClientDNS and the TargetDNS with secure DNS update, eliminating the Proxy from the list of available proxy servers.
4. The Coordinator will notify the Proxy, the Gateway, the Target to clean up the old indirect rotue.
5. The Client will temporarily lose connection to the Target. When time out, the Client will query the ClientDNS and get the new proxy server IP address.
6. The Client will set up indirect route though the newly selected proxy server.

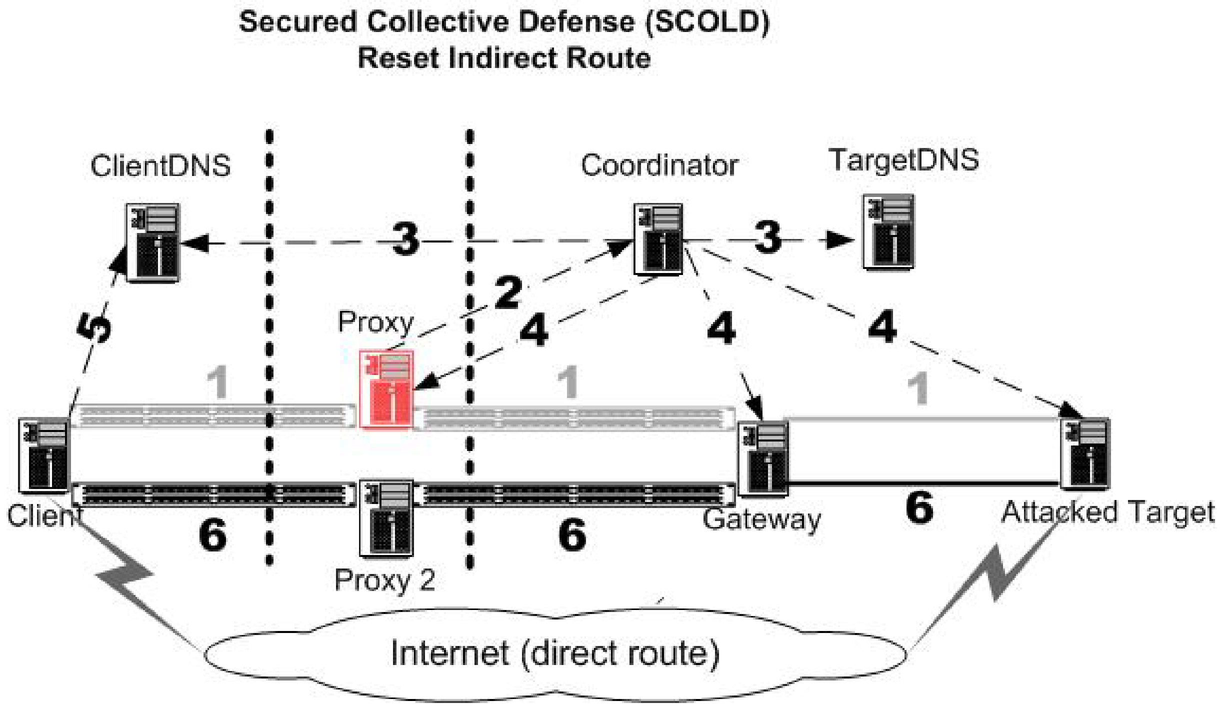


Figure 7: SCOLD Reset Indirect Route under DDoS attack

5. Test and Result.

To evaluate the overhead of indirect routing over IP tunnel and secure DNS update, we measure the response time and the throughput on direct and indirect routes in the testbed (Figure 8).

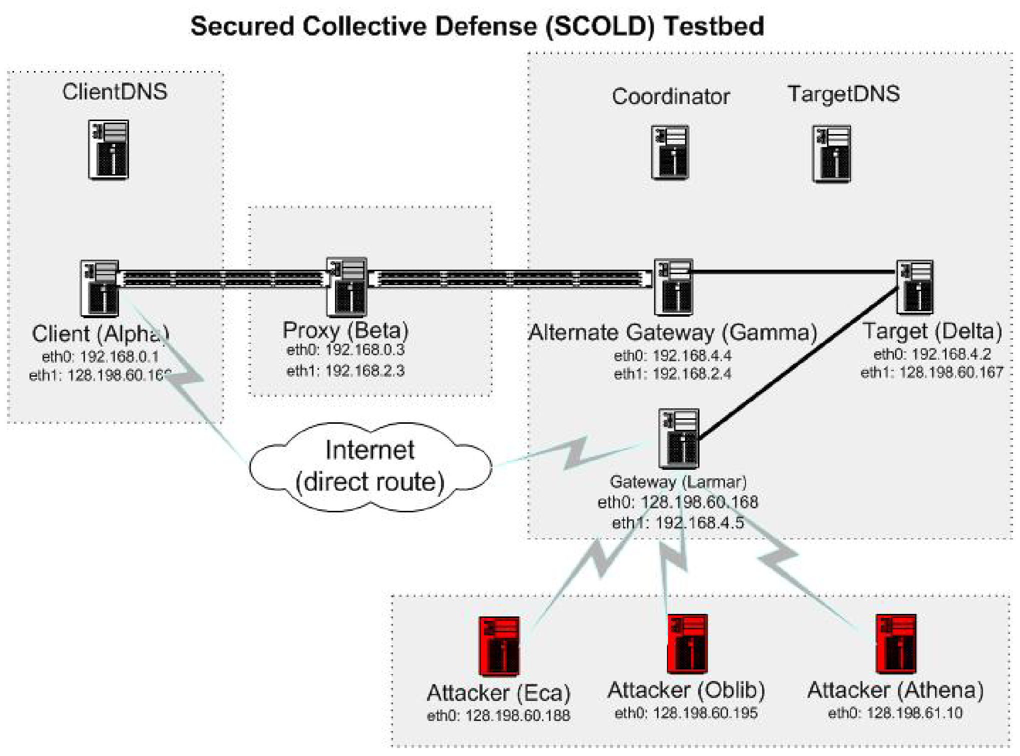


Figure 8: SCOLD testbed

5.1 Testbed machine configuration

Machine	Settings
Client	• Model: HP Vectra VL • CPU: Intel Pentium III 501.143 MHz • RAM: 255 MB • Hard Drive: 8455 MB • Network Interface: 3com 100 Mb • OS: Red Hat Linux 9.0
Proxy	Same
Gateway	Same
Target	Same
Coordinator	Same
ClientDNS	Same
TargetDNS	Same
Attacker	Same
Attacker	Same
Attacker	Same

StacheldrahtV4 [14] is used as the DDoS attacking tools. Among DDoS tools, StacheldrahtV4 is considered stable and sophisticated and is able to launch attacks in ICMP, UDP and TCP protocols.

5.2 Test Result and Analysis

Table 1: Ping Response Time Test (ping from client to target)

Test	No DDoS attack, direct route	DDoS attack, direct route	No DDoS attack, indirect route	DDoS attack, indirect route
Ping	0.49 ms	225 ms	0.65 ms	0.65 ms

In table 1, under DDoS attack, the ping response time increases dramatically. There are overhead associated with indirect route, but compared with impact of DDoS, the overhead is still acceptable. The response time of indirect route is not impacted too much by DDoS attack.

Table 2: SCOLD FTP/HTTP download Test (from client to target)

	No DDoS attack,		DDoS attack, direct		No DDoS attack,		DDoS attack, indirect	
Doc Size	FTP	HTTP	FTP	HTTP	FTP	HTTP	FTP	HTTP
100k	0.11 s	3.8 s	8.6 s	9.1 s	0.14 s	4.6 s	0.14 s	4.6 s
250k	0.28 s	11.3 s	19.5 s	13.3 s	0.31 s	11.6 s	0.31 s	11.6 s
500k	0.65 s	30.8 s	39 s	59 s	0.66 s	31.1 s	0.67 s	31.1 s
1000k	1.16 s	62.5 s	86 s	106 s	1.15 s	59 s	1.15 s	59 s
2000k	2.34 s	121 s	167 s	232 s	2.34 s	122 s	2.34 s	123 s

In table 2, under DDoS attack, the http/ftp download time increases dramatically. But the response time of indirect route is not impacted too much by DDoS attack.

Table 3: SCOLD Resolve Library Overhead Test

	Original Resolve Library	Enhanced Resolve Library
Ping	0.13 s	0.14 s
FTP download 100k	0.14 s	0.14 s
FTP download 500k	0.65 s	0.66 s
HTTP download	4.5s	4.6 s
HTTP download	31.1 s	31.1 s

In table 3, the overhead of modified resolve library is very limited.

Table 4: Time to Set up Indirect Route in SCOLD

Ping	Less than 1 s
FTP download	Less than 1 s
HTTP download	Less than 1 s

In table 4, the time used to set up indirect route is acceptable, compared with the possible delay caused by DDoS attack. The delay is primarily caused by the SSL authentication and communication.

As we can see from the above testing data, there is overhead associated with IP tunnel. The overhead occurs in the indirect route include more hop, more protocol processing (it goes through proxy server; and

IP over IP overhead related to fragmentation and reassembly). But when the main gateway got attacked, the performance on the direct route will go from seconds to days or infinity. Therefore the IP tunnel overhead is still acceptable.

6. Conclusion and Future Work

It is our hope that the preliminary research results of the SCOLD project will produce a valuable secure software package, and provide valuable insights for the network security related proposals.

Currently we are focus on the secure DNS update and indirect route. Future works include:

1) Incorporate the Intrusion Detection and Isolation Protocol (IDIP) and Service Location Protocol (SLP) in the SCOLD design. These will enhance the existing A2D2 architecture and make it more robust [19, 20].

2) The Linux-based proxy server needs to be integrated with the enhanced intrusion detection SNORT plug-in created in the Autonomous Anti-DDoS test bed project [4].

3) Algorithms need to be designed to choose the best proxy server or subset of servers from a given set of proxy servers, to enable maximum bandwidth usage.

4) The infrastructure of SCOLD system provides the possibility of using multiple-path routing to get better performance and overcome the overhead of indirect route and SSL connection. We need to design the protocol and algorithm for multiple-path routing using proxy servers in SCOLD project.

5) Social study of collective defense dynamic, interaction and politic among participating organizations.

6) Recruit interested organization to join the SCOLD project and contribute resources like proxy servers.

7) Test SCOLD on large scale Internet domain with large number of clients. Find out the scalability of proposed schemas.

7. References

1. Internetnews.com, “Massive DDoS Attack Hit DNS Root Servers”, http://www.internetnews.com/ent-news/article.php/1486981

2. David Moore, Geoffrey M. Voelker and Stefan Savage, “Inferring Internet Denial-of-Service Activity 2001”, http://www.cs.ucsd.edu/~savage/papers/UsenixSec01.pdf

3. The SANS Institute, “How To Eliminate The Ten Most Critical Internet Security Threats” http://www.sans.org/top20/top10.php, 2001

4. Angela Cearns, Master Thesis “Design of an Autonomous Anti-DDoS network (A2D2)”, http://cs.uccs.edu/~chow/pub/master/acearns/doc/angThesis-final.pdf, 2002

5. Steven Cheung, Ph.D. thesis “An Intrusion Tolerance Approach for Protecting Network Infrastructures”, http://citeseer.nj.nec.com/cheung99intrusion.html, 1999

6. Jelena Mirkovic, Janice Martin and Peter Reiher, “A Taxonomy of DDoS Attacks and DDoS Defense Mechanisms”, http://www.lasr.cs.ucla.edu/ddos/ucla_tech_report_020018.pdf

7. Network Associates Labs and Boeing, “IDIP Architecture”, http://zen.ece.ohiou.edu/~inbounds/DOCS/reldocs/IDIP_Architecture.doc, 2002.

8. SVRLOC working group, “Service Location Protocol (SLP) Project”, http://www.srvloc.org/

9. Michael D. Bauer,” Securing DNS and BIND”, ACM Linux Journal Volume 2000 Issue 78es, October 2000

10. John Viega, Matt Messier & Pravir Chandra, “Network Security with OpenSSL”, O’Reilly, 2002

11. Edward. Chow, “Security Related Research Projects at UCCS network research lab”, http://cs.uccs.edu/~Echow/research/security/uccsSecurityResearch.ppt, 2002

12. OpenSSL, “OpenSSL”, http://www.openssl.org

13. Eric Green, “Glibc Howto”, http://www.imaxx.net/~thrytis/glibc/Glibc2-HOWTO.html

14. Astalavista Network Library Archive. http://www.astalavista.com/archive/index.asp?dir=ddos

15. “Zebedee Secure IP tunnel”, http://www.winton.org.uk/zebedee/

16. “SOCKS proxy server”, http://www.tldp.org/HOWTO/Firewall-HOWTO-11.html

17. “IPIP tunnel”, http://www.europe.redhat.com/documentation/HOWTO/Net-HOWTO/x1284.php3

18. “DNS BIND 9”, http://www.isc.org/products/BIND/

19. Network Associates Labs and Boeing, “IDIP Architecture”, http://zen.ece.ohiou.edu/~inbounds/DOCS/reldocs/IDIP_Architecture.doc, 2002.

20. SVRLOC working group, “Service Location Protocol (SLP) Project”, http://www.srvloc.org/

21. D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, R. Morris, "Resilient Overlay Networks," In Proceedings of 18th ACM SOSP, October 2001.

22. Information Sciences Institute, "Dynabone," http://www.isi.edu/dynabone/

23. J. Yan, S. Early, R. Anderson, "The XenoService – A distributed defeat for distributed denial of service", In Proceedings of ISW 2000, October 2000.

24. Asta Networks, "Vantage System Overview," http://www.astanetworks.com/products/vantage/

25. BBN Technologies, "Applications that participate in their own defense," http://www.bbn.com/infosec/apod.html