Secure Collective Defense System*

C. Edward Chow, Yu Cai, David Wilkinson, and Ganesh Godavari
Department of Computer Science
University of Colorado at Colorado Springs
Colorado Springs, CO 80933-7150, USA
{chow, ycai, dbwilkin, gkgodava }@cs.uccs.edu

Abstract**—**In this paper, we present the design and implementation of the Secure COLlective Defense (SCOLD) system against DDoS attacks. The key idea of SCOLD is to follow intrusion tolerance paradigm and provide alternate routes via a set of proxy servers and alternate gateways when the normal route is unavailable or unstable due to network failures, congestion, or DDoS attacks. The BIND9 DNS server and its DNS update utilities are enhanced to support new DNS entries with indirect routing information. Protocol software for supporting the establishment of indirect routes based on the new DNS entries is developed for Linux systems. Experimental results show that SCOLD can improve the network security, availability and performance. Preliminary simulation results using NS2 indicate that the performance is scalable with respect to the indirect route initial setup overhead and processing overhead.

***Keywords**: Intrusion Detection, Intrusion Tolerance, DDoS, Secure Collective Defense, Indirect Route, secure DNS update*

1. Introduction

The increasing frequency and severity of network attacks nowadays reveal one of the fundamental security problems of today's Internet. Many network services like Domain Name System (DNS) and protocols like TCP were not originally designed with security as one of the basic requirements. The highly distributed and interdependent nature of Internet provides opportunities and resources for the coordinated and simultaneous malicious actions by some participants. Due to the same nature, it is difficult to enforce common security policies, measurements and coordination among the participants of Internet. Therefore, the existing Internet architecture needs to be strengthened and service protocols need to be enhanced or re-designed with security in focus.

The objective of the Secure Collective Defense (SCOLD) project is to create a secure collective Internet defense system that utilizes resources allocated by participating organizations. The key idea of SCOLD is to provide clients with alternate routes via a set of proxy servers when the normal route is unavailable or unstable. The main techniques utilized in SCOLD are Indirect Route and Secure DNS Update. SCOLD can be used to defend against DDoS attacks, or to provide alternate or additional routes to satisfy on-demand operational requirements.

The balance of this paper is organized as follows. In Section 2, we give an overview of the SCOLD system. In Section 3, we present the design and implementation of enhanced secure DNS update with multiple indirect route entries and DNS query via indirect route. In Section 4 the indirect routing using IP Tunnel is presented. In Section 5 we present experimental results and simulation results. Related work is surveyed in Section 6 while the conclusion is drawn in Section 7.

2. SCOLD System Overview

2.1 Motivation

Most organizations today deploy multiple gateways or multi-homing scheme to defend against a large scale DDoS attack. When the main gateway is under DDoS attack, the clients' traffic should be redirected to the alternate gateways. However, once the alternate gateways are exposed to public domain, they are subjected to DDoS attacks.

Therefore, the two challenges are how to utilize alternate gateways while hiding their IP addresses from public domain, and how to redirect the heterogeneous clients' traffic to alternate gateways.

2.2 System architecture

Most DDoS defense techniques presume the scenario where packets are transmitted along a normal Internet route while the intermediate network topology is unchanged. Under large-scale DDoS attacks, such techniques may suffer significant performance degradation.

The SCOLD system defends against DDoS attacks by setting up indirect routes between clients and target server via a collection of geographically separated proxy servers and alternate gateways. The traffic between clients and target server is transported over Internet through the indirect routes.

Figures 1-3 illustrate how SCOLD system works. Figure 1 shows a target site under DDoS attacks where R is the main gateway, and R1-R3 are the alternate gateways. In the figure the majority of the traffic from net-a.com is malicious, that of net-b.com is legitimate, and that of net-c.com is mixed.

* This research work was supported in part by a NISSC AFOSR Grant award under agreement number F49620-03-1-0207.

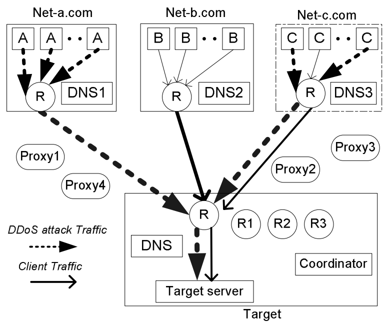

Figure 1: Target site under DDoS attack

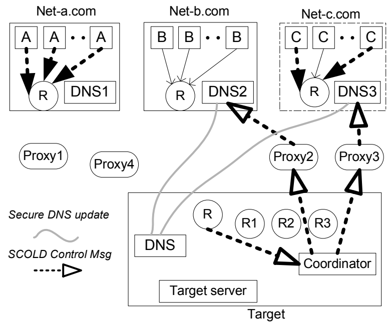

Figure 2: The control flow in SCOLD

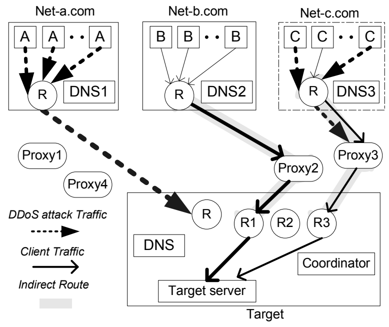

Figure 3: Indirect route in SCOLD

Figure 2 shows the control flow of the SCOLD system. When the target site is under DDoS attacks, its Intrusion Detection System (IDS) raises an intrusion alert and notifies the SCOLD coordinator, who sits in the same or trusted domain of the target server. The coordinator then notifies some selected proxy servers (proxies 2 and 3 here) to set up indirect routes. The proxy servers notify the DNS servers of the legitimate clients to perform a secure DNS update. The clients from net-b.com and net-c.com are notified with indirect routes, but net-a.com is not notified due to its malicious traffic pattern.

Figure 3 shows how an indirect route is setup in SCOLD system. After a secure DNS update, the client side DNS server gets the new DNS entry containing the designated proxy servers IP addresses. The clients query the DNS server, get the proxy server IP addresses, and can set up indirect routes to the target server via the selected proxy servers. The proxy servers examine the incoming traffic and relay it to a designated alternate gateway on target site.

On the client side, the name resolve library needs to be enhanced and the routing table needs to be modified to support the indirect routing. In enterprise environment, the internal clients go outside through an enterprise gateway (or an enterprise proxy server). Instead of modifying each client, only the enterprise gateway needs to be enhanced to support the indirect route.

Note that the scheme we proposed doesn't help existing connections. One way to affect the existing connections is to install software on the client machines that listen to the rerouting message from the SCOLD proxy servers.

In SCOLD, the IP addresses of the alternate gateways and the SCOLD coordinator(s) are revealed only to the trustworthy proxy servers to protect them from being attacked by malicious clients. The clients in public domain can connect to the target side through the designed proxy servers.

The proxy servers in SCOLD are enhanced with IDS and firewall filters to block malicious traffic that may try to come in through the indirect routes. The detection of intrusion on the proxy servers can provide additional information for identifying and isolating the spoofed attack sources. In Figure 3, the attack source from net-c.com could be more accurately identified by combing the intrusion detection results from the main gateway R and the proxy server 3.

A proxy server itself may suffer from DDoS attacks or get congested when large volume of traffic comes through it. Assuming a large collection of proxy servers available, the impact of heavy traffic can be alleviated by spreading traffic over multiple proxy servers. However, detecting and handling the comprised proxy servers is not an easy task. To avoid traffic analysis by intruders, multiple proxy servers can be deployed on each indirect route.

The procedure for resuming normal route is similar to setting up indirect route. The proxy servers need to notify the client DNS servers with another secure DNS update to restore the normal DNS records. The clients query the DNS server and start to resume the normal direct route. We can also set an "expiration time" on indirect route so that SCOLD can automatically revoke obsolete indirect routes.

All the control messages communicated in SCOLD system are encrypted using Secure Sockets Layer (SSL) and all nodes involved must be mutually authenticated. Experiments show that this is one of the major causes of overhead in SCOLD system.

Proxy servers can be provided by the participating organizations of SCOLD, or fee-based service providers.

2.3 More SCOLD applications

An enhanced SCOLD with integrated IDS can be used to defend against very large-scale DDoS attacks like MyDoom [4], which knocked the SCO website offline in 2004.

The SCOLD coordinator collects and analyzes the target server system load, available network bandwidth and the statistics of the client traffic. Based on the information, the coordinator can inform each proxy server what is the allowed maximum bandwidth usage connecting to the target server. The proxy servers equipped with admission control and rate-limiting mechanism can enforce such bandwidth throttling. In Figure 3, the coordinator may assign different allowed maximum bandwidth to proxy 2 and 3, depending on the sever load and client behavior. The integrated IDS can control aggressive or malicious clients and reserve resources for normal operation.

A slightly revised version of SCOLD can be used to protect the Root DNS servers from DDoS attacks, like the one caused a brief service disruption on the nine of the thirteen DNS root servers in 2002 [3]. In Figure 4, the DNS servers 1-3 are clients and the root DNS server is the target. The steps to set up indirect routes are similar to what we described before. The server side IDS raises alert and notifies the coordinator; the coordinator notifies the selected proxy servers (proxies 2, 3 here); the proxy servers notify the DNS servers with their IP addresses; the DNS servers then set up indirect routes to the root DNS via the proxy servers and the alternate gateways.

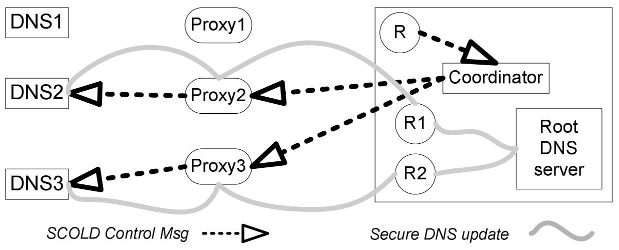

Figure 4: Protect the root DNS server

3. Secure DNS update

As discussed in Section 2, one of the challenges in SCOLD is to inform clients about the proxy server IP addresses. We propose to utilize existing DNS system by adding additional proxy server IP addresses in DNS record. The new DNS record in the DNS zone file looks 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
```

The first line is a normal DNS entry, containing host name and its IP address. The next 3 lines contain the IP addresses of proxy servers, as newly defined "ALT" (alternate) type. Such DNS entries need to be securely updated from target side DNS server to client side DNS servers upon request. When client queries its own DNS server, it gets informed whether an indirect route needs to be set up and how to set it up, by checking the entries with "ALT" type data.

During the DNS record transfer from the target DNS server to the client DNS server, the main gateway on target domain may be unavailable or unstable due to DDoS attacks. Therefore, we use indirect route to perform the update. Figure 5 illustrates how secure DNS update via indirect route works. The target side IDS raises intrusion alert, notifies the coordinator; the coordinator notifies the selected proxy server(s); the proxy server notifies the client DNS server; the client DNS server set up indirect route to the target DNS server via the proxy server and the alternate gateway; the client DNS server performs the secure DNS update and get DNS records from target DNS server.

The existing DNS server needs to be modified to support the new DNS record format. On client side, the domain name resolve library needs to be modified to enable the automated setup of indirect route. New routing entries with IP-over-IP interfaces need to be inserted to the routing table on a client.

There are other secure DNS solutions like DNSSEC [5] (DNS Security Extensions) and secure DNS dynamic update [6]. But they don't support the new DNS record format and the secure DNS update through indirect route, therefore, does not fit into the SCOLD context.

4. Indirect route

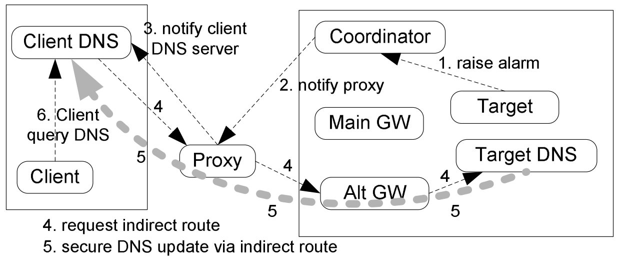

Figure 5: Secure DNS update via indirect route

We investigate several alternatives for implementing indirect route, including SOCKS proxy [7], Zebedee [8], IPSec [9] and IP tunnel [10].

SOCKS proxy server is like an old switchboard and can cross wire between connections. The main drawbacks of SOCKS are that it doesn't support UDP and FTP.

Zebedee is an application to establish an encrypted and compressed tunnel between two systems. But it requires specific configuration per network application.

IP tunnel is a technique to encapsulate IP datagram within IP datagram. This allows a datagram destined for one IP address to be wrapped and redirected to another IP address. IP tunnel provides what we want for indirect routing.

IPSec is an extension to the IP protocol which provides security to the IP and the upper-layer protocols. We believe whether client traffic needs to be encrypted is a client decision. Therefore, we choose IP tunnel to support basic indirect routing. However, the implementation using IP tunnel can be enhanced to use IPSec easily. IP tunnel and IPSec have been used widely in Virtual Private Network (VPN) [14] to set up "tunnels" between network nodes and redirect traffic.

The advantages of using IP tunnel are as follows. IP tunnel is a layer three protocol. All the upper layer protocols and applications can utilize it. Second, IP tunnel is a widely used protocol and supported by most modern operating systems. Last but not the least, IP Tunnel consumes limited system resources.

There is overhead associated with IP Tunnel due to the extra set of IP header and the reduced payload size. This can also cause fragmentation and hence introduce reassembly overhead. In our experiments, the overhead in term of response time varies between 30% and 200%. But compared with the impact of DDoS attack, which may cause unbearable delay, the overhead of IP tunnel is still in an acceptable range. Fragmentation overhead can be avoided if we restrict the message transfer size at the sender.

Figure 6 illustrates how the indirect route is set up by using IP tunnel. The client queries its DNS and get the IP addresses of proxy servers; the client sends a request to a proxy server for indirect route; if the proxy server grants permission, it notifies the designated alternate gateway; the alternate gateway notifies the target server, then an indirect route can be set up between the client and the target server via the proxy server and the alternate gateway. We set a timeout value at client side in case the communication is lost or the indirect route is broken.

5. Experimental and simulation results

In this section, we present some preliminary experimental and simulation results on a SCOLD prototype.

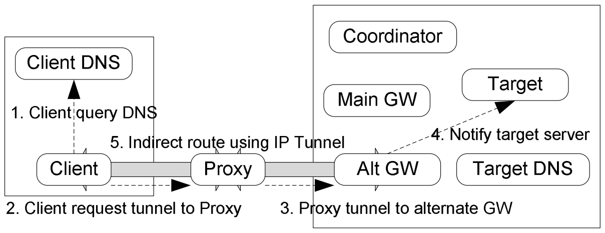

Figure 6: Indirect route by using IP tunnel

5.1 Prototype implementation

We implement the secure DNS update and DNS query via indirect route on Bind9 (v.9.2.2) DNS package [11], by modifying the nsreroute command, and putting add-ons to the BIND9 DNS server. The domain name resolve library is enhanced (v.2.3.2) by modifying the res_query() routine to support the indirect route. In Redhat Linux, the resolve library is usually located in /usr/lib or /lib directory, and named as libresolv-nnn.so (nnn is the version). The indirect route on Linux Redhat 8 and 9 is realized with IP tunnel. We also test indirect route on Windows 2000 server using IP tunnel. OpenSSL (v.0.9.6) [2] is utilized for authentication and encryption.

5.2 Experimental setup

We set up a testbed consists of more than 20 nodes with various machine settings. The testbed includes HP Vectra machines (PIII 500MHz, 256MB RAM, 100Mb Ethernet connection), HP Kayak machines (PII 233MHz, 96MB RAM, 10/100 Mb Ethernet connection), Dell machines (PIII 1GHz, 528MB RAM, 100 Ethernet connection) and vmware virtual machines (96MB RAM, 100 Mb virtual Ethernet connection, running on a Dell machine with dual PIII 1.2GHz and 4G RAM). The operating systems are Linux Redhat 8, 9 and Windows 2000 server.

StacheldrahtV4 [12] is used as the DDoS attack tool.

5.3 Analysis of the experimental results

We first evaluate the time taken to initially set up an indirect route in SCOLD. As discussed previously, there are three steps involved. Step 1, "IDS → coordinator → proxy". The overhead comes from the secure communication among nodes. Step 2, "Proxy → client DNS → perform secure DNS update". The overhead comes from the secure communication and the secure DNS update. Step 3, "client → client DNS → set up indirect route". The overhead comes from the secure communication, the client side resolve library processing overhead and the time to set up the indirect route.

Table 1 shows the initial setup time in SCOLD. It is observed that the overhead comes primarily from the secure DNS update and the secure communication among nodes. Table 2 further shows that the secure DNS update time increases dramatically when the number of client DNS servers increase. This suggests that there is a limit on how many client DNS servers a proxy server can handle concurrently.

Table 1: Initial setup time (second)

Step 1	**Step 2**	**Step 3**	**Total**
2.1	4.7	2.7	9.5

Table 2: Secure DNS update time (second)

1 DNS	**10 DNS**	**25 DNS**	**50 DNS**
4.7	25	96	240

Table 3 shows the processing overhead by using indirect route after it is set up. It comes from the IP tunneling overhead and more Internet hops involved in indirect route. We can observe that the overhead of indirect route in term of response time is about 70%. Further experiments shows the overhead varies from 30%–200%. However, under DDoS attack, the response time of using direct route increases dramatically (15 times to infinity), while the response time of using indirect route keep the same (Assuming no DDoS attack against proxy servers directly).

Table 3: Indirect Route processing overhead vs. Direct Route delay under DDoS attack

Test	No attack			Under DDoS attack		
	Direct Route	Indirect Route	**Indirect Route Overhead**	Direct Route	**Direct Route Delay**	Indirect Route
Ping	49 ms	87 ms	77%	1048 ms	21 times	same as no attack
HTTP(100k)	6.1s	11s	80%	109s	18 times	
HTTP(500k)	41s	71s	73%	658s	16 times	
HTTP(1M)	92 s	158s	71%	timeout	infinity	
FTP(100k)	4.2 s	7.5s	78%	67s	16 times	
FTP(500k)	23 s	39s	69%	345s	15 times	
FTP(1M)	52 s	88s	69%	871s	17 times	

Table 4: The influence of how many tunnels exist

Test	**1 tunnel**	**10 tunnels**	**50 tunnels**	**100 tunnels**
Ping	87 ms	87 ms	87 ms	87 ms
HTTP(100k)	11s	11s	11s	11s

Table 4 shows that the number of IP tunnels on a network node doesn't affect the performance, because IP tunnel itself consumes very limited system resources.

It is observed that, compared with the impact of DDoS attacks, SCOLD can improve the network security, availability and performance with acceptable initial setup overhead and processing overhead.

5.4 Preliminary simulation results

To further analyze the overhead in SCOLD, the ns2 simulator [13] is used to perform the simulation study for large-scale network. The topologies used in simulation are generated using GT-ITM [15]. We create transit-stub graphs with 100-500 nodes. We pick nodes in the same stub for target server, target DNS server, coordinator, main gateway and three alternate gateways. We randomly pick 10% nodes as proxy servers, 5% nodes as DDoS attackers, 20% nodes as clients and 4% nodes as client DNS servers.

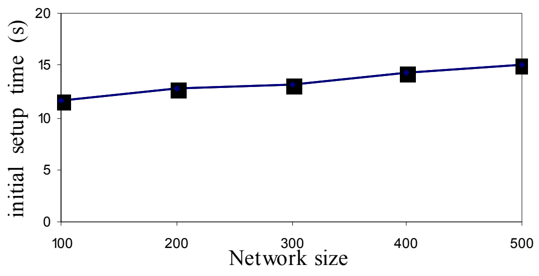

Figure 7: average initial setup time vs. network size

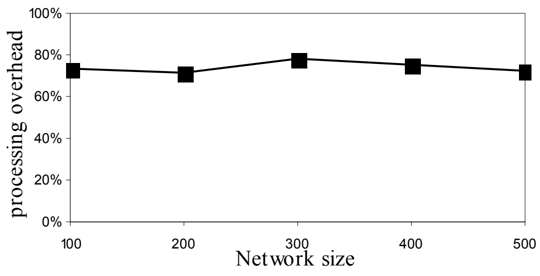

Figure 8: indirect route processing overhead vs. network size

For simplicity, we set the overhead of IP tunneling and the overhead of secure communication to be a fixed percentage with a small random change. We randomly generate background traffic whose average is 60% of the total network bandwidth. We generate DDoS attack traffic which can completely shutdown the victim. For simplicity, we assume proxy servers are not attacked directly.

Figure 7 shows that the average initial setup time of indirect route increases slowly when the network size increases. Figure 8 shows that the indirect route processing overhead keeps nearly constant when the network size increases. In both figures, SCOLD demonstrates good scalability with respect to the initial setup overhead and the processing overhead.

6. Related works

J. Mirkovic, et al. presented a taxonomy of DDoS attacks and DDoS Defense Mechanisms [16]. SCOLD falls into the category of reconfiguration and cooperative mechanism. Related works in reconfiguration mechanism include reconfigurable overlay networks [17], [18], resource replication services [19] and attack isolation strategies ([20]). These works focused on either adding more resources to the victim or isolating the attack machines by reconfiguring the network. SCOLD also changes the intermediate network topology but with different techniques and purposes. The cooperative DDoS defense mechanism is limited by the highly independent nature of Internet. SCOLD manages to utilize collective resources with tighten coordination and cooperation.

Akamai [1] is a distributed content delivery system which significantly alleviates service bottlenecks and shutdowns by delivering content from the Internet's edge. Akamai redirects client requests to the nearest available server likely to have the requested content. With more than 12,000 servers in over 1,000 networks, Akamai routinely delivers 15% of the total Web traffic. The similar between SCOLD and Akamai is that both redirect client traffic. Even though they are used for different purposes, they could benefit from each other by sharing the service servers.

7. Conclusions

We present the SCOLD architecture to defend against DDoS attacks by redirecting the traffic between clients and servers through indirect routes via proxy servers and alternate gateways. BIND9 DNS package is modified to support secure DNS update. IP tunnel is utilized to implement indirect routing. The preliminary results show that SCOLD can improve the network security, availability and performance. It is our hope that the research results can produce a valuable secure software package, and provide insights for network security and Internet cooperation.

Acknowledgement

This material is based on research sponsored by the Air Force Research Laboratory, under agreement number F49620-03-1-0207. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory or the U.S. Government.

References

[1] Akamai.com, http://www.akamai.com
[2] OpenSSL, http://www.openssl.org
[3] Internetnews, "Massive DDoS Attack Hit DNS Root Servers", http://www.internetnews.com/
[4] Internetweek, "SCO Moves Web Site To Battle MyDoom", http://www.internetweek.com/
[5] DNSSEC, http://www.dnssec.net/
[6] Secure DNS dynamic update, http://www.faqs.org/rfcs/rfc3007.html
[7] SOCKS proxy server, http://www.tldp.org/HOWTO/Firewall -HOWTO-11.html
[8] Zebedee, http://www.winton.org.uk/zebedee/
[9] IPSec, http://www.ietf.org/html.charters/ipsec-charter.html
[10] "IPIP tunnel", http://www.europe.redhat.com/documentation/HOWTO/Net-HOWTO/x1284.php3
[11] DNS BIND 9, http://www.isc.org/products/BIND/
[12] StacheldrahtV4, http://cs.uccs.edu/~scold/ddos
[13] NS2, http://www-mash.cs.berkeley.edu/ns
[14] Virtual Private Network, http://www.vpnc.org/
[15] GT-ITM, http://www.cc.gatech.edu/projects/gtitm/
[16] Jelena Mirkovic, et al. "A Taxonomy of DDoS Attacks and DDoS Defense Mechanisms", UCLA Technical Report
[17] D. G. Andersen, et al., "Resilient Overlay Networks," In Proceedings of 18th ACM SOSP, October 2001.
[18] Information Sciences Institute, "Dynabone", http://www.isi.edu/dynabone
[19] J. Yan, S. et al., "The XenoService – A distributed defeat for DDoS", In Proceedings of ISW 2000.
[20] BBN Technologies, "Applications that participate in their own defense," http://www.bbn.com/infosec/apod.html