A Stateful CSG-based Distributed Firewall Architecture for Robust Distributed Security

V. Ramsurrun, and K. M. S. Soyjaudah
Electrical & Electronic Engineering Department
University of Mauritius (UoM)
Réduit, Mauritius
vishamr2000@gmail.com, ssoyjaudah@uom.ac.mu

***Abstract*—Distributed firewalls have been developed in order to provide networks with a higher level of protection than traditional firewalling mechanisms like gateway and host-based firewalls. Although distributed firewalls provide higher security, they too have limitations. This work presents the design & implementation of a new distributed firewall model, based on stateful Cluster Security Gateway (CSG) architecture, which addresses those shortcomings. This distributed security model adopts a bottom-up approach such that each cluster of end-user hosts is first secured using the CSG architecture. These different CSGs are then centrally managed by the Network Administrator. A file-based firewall update mechanism is used for dynamic real-time security. IPsec is used to secure the firewall policy update distribution while X.509 certificates cater for sender/receiver authentication. The major benefits of this approach to distributed security include tamper resistance, anti-spoofing, anti-sniffing, secure real-time firewall updating, low overall network load, high scalability and low firewall convergence times.**

Keywords-stateful CSG architecture, distributed firewall, distributed cluster security, Layer 2 per-packet load balancing

I. INTRODUCTION

Distributed firewalls have been devised with a view to address the problems of traditional firewalls like gateway and host-based firewalls. Although distributed firewalls achieve their purpose, they, too, are not free of shortcomings (Section 2). Problems like increase in processing load on end-user hosts due to the packet filtering strain, decrease in overall network performance because of dynamic firewall updating, and user tampering in particular, hamper the deployment and usage of distributed firewalling solutions. In light of the shortcomings of distributed firewalls, a new approach to distributed firewalling, based on the stateful CSG architecture, has been designed and implemented in order to overcome those limitations. The CSG architecture [1] provides a methodology for grouping together multiple networking elements such as routers, security gateways, and switches in order to create more secure, more reliable switched network clusters. The motivation behind the CSG-based distributed firewall design is that if robust security is provided at the very cluster level, the whole of the network will become more secure as we can reduce the occurrence of both insider & external attacks, and limit their spread & effects more readily. A 2-active-node stateful CSG is used for protecting each end-user cluster in our working prototype.

In this paper, we perform the following:

1. Review of the strengths & limitations of distributed firewalls.
2. Use of the stateful CSG to implement a new robust distributed firewall model.
3. Qualitative comparison of its strengths & weaknesses with other major software-based & hardware-based distributed firewall architectures available.

II. DISTRIBUTED FIREWALLS

Pioneered by Steven Bellovin [2] in 1999, distributed firewalls have been created in response to the limitations of both gateway & host-based firewalls, and more specifically, in order to prevent insider attacks. According to Ioannidis et al. [3], a distributed firewall is a mechanism that enforces a centralized security policy but the latter is applied at the edges. Distributed firewalls are basically centrally managed host-resident security software applications that protect a network's critical endpoints against unwanted intrusions. The conceptual design of distributed firewalls rests upon three elements:

1. A general policy language that is used for defining security policies that are distributed to the firewall endpoints forming the distributed firewall. Examples of general policy languages include KeyNote [4] and Firmato [5].
2. Network-wide mechanisms for the distribution and application of the security policy files to the distributed firewall endpoints.
3. IPsec: security protocol that provides network-level encryption for the secure transmission of the security policy.

The financial support of the Tertiary Education Commission (TEC) of Mauritius is gratefully acknowledged.

A. *Major Strengths*

- *Centralized management*

Security policies are formulated centrally and then distributed to the different endpoints for enforcement. Coherence of security policies over the network and control over their deployment is enhanced and maintained [3].

- *Defense in depth*

When used together with the gateway firewall, distributed firewalls provide multiple layers of defense that an attacker has to pierce through. This makes the task of the attacker much more difficult, allows time to other defense mechanisms to counter the threat effectively, and thus, delay & prevent its spread in the network [6].

B. *Major Limitations*

- *User tampering*

According to Wei Li [7], this represents the biggest problem in distributed firewalls. Users requiring administrator privileges to work, can modify host-based firewall rules at will or completely remove the firewall, thereby exposing those hosts to attacks. Hackers can, in turn, use those hosts as base for launching attacks from inside the network. Both internal & remote hosts can be attacked.

- *Decrease in network performance*

The utilization of real-time security policy updates will add considerable strain on the network with all the traffic that is being generated by the distributed firewall. As a result, the network becomes more vulnerable to DoS attacks [7].

- *Increase in host load*

There is degradation in host performance. The host-level packet filtering adds considerable load on hosts with limited resources. In addition, with the implementation of other security tools at the host level like real-time host-based intrusion detection systems and Portsentry, as in the security model devised by M. Gangadharan and K. Hwang [6], [8] hosts will be heavily taxed.

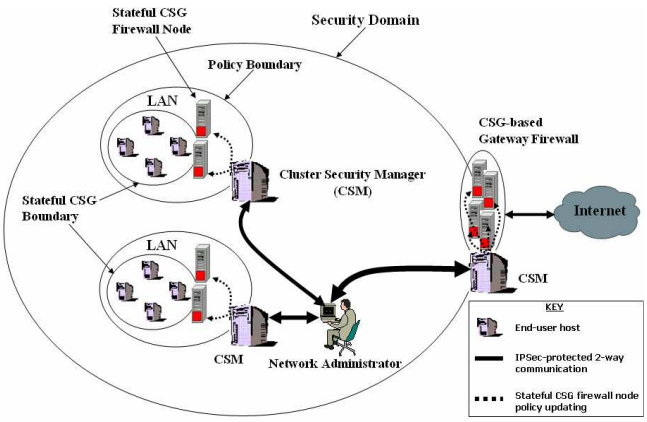

Figure 1. The stateful CSG-based distributed firewall architecture.

- *High reconfiguration time of host-resident components of distributed firewalls*

Since distributed firewalls allow for dynamic updating of security policies, the bigger the size of a network, the more time it takes to re-deploy security policies. The convergence time of the network hosts and their firewalls is much higher as it is directly dependent on the number of hosts found on the network.

III. DESIGN OF THE STATEFUL CSG-BASED DISTRIBUTED FIREWALL

The stateful CSG-based distributed firewall is as shown in Fig. 1. This novel distributed firewalling architecture consists of four main components, namely the Network Administrator machine, the Cluster Security Manager (CSM), the stateful CSG & the CSG-based gateway firewall, and several sub-systems like the Policy Repository, the Policy Distributor & the Policy Handler. In our test implementation, each end-user cluster is protected by a 2-active-node stateful CSG. Each CSG-protected end-user cluster possesses a dedicated CSM, whose main job is to receive firewall updates from the Network Administrator and forward them to the CSG firewall nodes falling under its responsibility. The workings of these different components are described in greater detail below:

A. *The Network Administrator Machine*

This machine is used by the Network Administrator for managing the various network components. It is from this computer that the Network Administrator updates CSG firewall nodes. This machine contains two major components – the Policy Repository and the Policy Distributor.

1) The Policy Repository: The Policy Repository is a central database where all the firewall scripts deployed in the network are stored. All the firewall updates are also stored there. The Network Administrator can thus consult the existing firewall scripts in order to create new firewall update files when the network is under attack. The firewall scripts and update files are stored in usable forms (for example, as *.sh* files) so that they can be directly applied onto the firewall nodes. All scripts and updates pertaining to a particular cluster are stored together for easy referencing. File versioning and creation details are also kept.

2) The Policy Distributor: The Policy Distributor is used by the Network Administrator for sending firewall updates to Cluster Security Managers (CSMs). The Policy Distributor establishes end-to-end connections with the appropriate CSMs. These connections are authenticated & encrypted for secure transmission of firewall updates across the network. This helps in preventing threats like man-in-the-middle attack, replay attack and IP address spoofing. Firewall updates are distributed to firewall nodes via CSMs because direct updating of firewall nodes will require secure connections (for instance,

using IPsec) between the Policy Distributor and the firewall nodes themselves. This will significantly increase the processing strain on the firewall nodes as they will then be acting as IPsec gateways. IPsec packet processing, together with Ebtables & IPTables packet filtering, will considerably reduce the efficiency and throughput of the firewall nodes. The firewall update distribution mode is determined by the number of secure connections to be established. Unicast transmission mode is preferred over multicast as not all the CSMs will need updating at a particular point in time. There may be firewall updates that are meant only for one cluster and not for the rest of the clusters in the network. If all the CSMs are made part of a single multicast group, then all of them will have to accept the firewall updates. This will cause their respective rulesets to increase in size unnecessarily, and this will potentially affect firewall performance.

B. *The Cluster Security Manager (CSM)*

The CSM is the first and foremost recipient of firewall updates from the Policy Distributor. It is the endpoint of the secure connections established by the Policy Distributor. Each end-user cluster has exactly one CSM. The CSM consists of a user-level process that waits for firewall updates from the Policy Distributor and then distributes them to the stateful CSG firewall nodes falling under its responsibility.

C. *The Stateful CSG*

It comprises of multiple active firewall nodes working in parallel to filter traffic (intra-cluster, inter-cluster and remote communication traffic) travelling to/from the end-user hosts of a particular cluster. The CSG architecture uses a different type of load balancing – the Ebtables distributed sender-initiated MAC-based per-packet load balancing (PPLB) scheme, where the load balancing is done by the end-user nodes themselves. PPLB helps make optimum usage of network links by allowing for equal distribution of traffic along those links. This Layer 2 PPLB scheme has been developed primarily for a seamless integration in load balancing setups involving stealth firewalls, especially where IP addressing is not used. It load balances network traffic onto MAC addresses rather than IP addresses. However, it can be successfully utilized in IP-based networks as well. This scheme is advantageous as it prevents the creation of single points of failure by removing the need for a dedicated load balancer, and it integrates well in already-in-place switched networks so that no major network re-design is required. The CSG architecture deployed for each end-user cluster provides the following security mechanisms:

- Layer 2 and Layer 3 packet filtering using Ebtables and IPTables respectively.
- Network Access Control (NAC) using MAC ACLs applied on specific switch ports [1] to ensure that end-user hosts communicate only via the firewall nodes.
- Port security [9] so as to prevent source MAC address spoofing.

1) The Policy Handler: The Policy Handler runs on each of the firewall nodes. It receives updates from its CSM and integrates them in the current firewall ruleset. Since updates are in directly usable format, firewall rules can be inserted or deleted easily.

D. *The Stateful CSG-based Gateway Firewall*

The gateway firewall, which is the first line of access control & protection against external attacks, needs dynamic updating as well in the face of emerging threats. In our security model, a CSG-based gateway firewall is used, where load balancing and failover techniques not only help in eliminating the single point of failure, but also help boost firewall throughput and reliability. Like the end-user clusters, the CSG-based gateway firewall, too, has a CSM for receiving firewall updates from the Network Administrator.

IV. IMPLEMENTATION DETAILS

The high-level architecture of the stateful CSG-based distributed firewall is as shown:

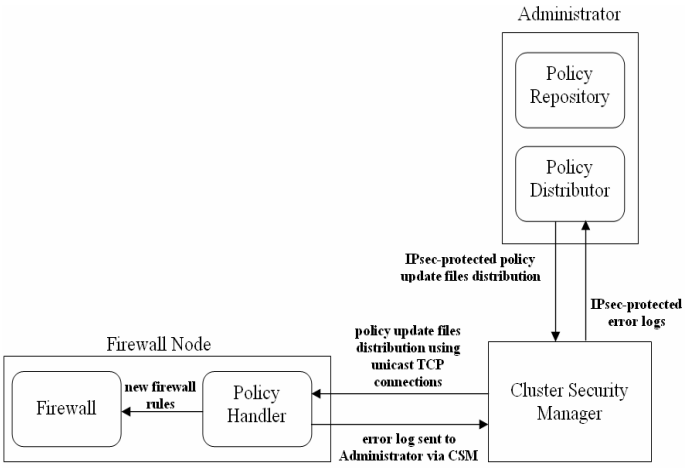

Figure 2. High-level implementation details of the stateful CSG-based distributed firewall.

The different software components are implemented as follows:

A. *The Policy Distributor*

The Policy Distributor (*/etc/dfw/pol_d.c*) is implemented as a user-space program written in C. The program takes as argument the full pathname of firewall update file and the IP addresses of the appropriate CSMs. TCP is used for distributing the firewall updates to the CSMs as it provides reliable delivery of packets, which is crucial to the delivery of firewall updates. The Policy Distributor uses the *pd_updatehandler()* function to read the specified firewall update file, and the *pd_sendupdate()* function sends the update out to each specified IP address.

B. *The Cluster Security Manager (CSM)*

The CSM is made up of two user-space parts - a firewall update receiving part and a firewall update sending part. Both parts make use of TCP sockets. The receiving & sending parts are implemented in the */etc/dfw/csm.c* file. In the receiving part, the packets from the Policy Distributor are read by the *csm_updatehandler()* function. The firewall update file is reconstructed on the CSM so as to ensure that the file is received error-free and in its entirety. If an error occurs, it is logged and the Network Administrator is notified. In the sending part, the *csm_sendupdate()* function is invoked, which then sends the firewall update out to each of the CSG firewall nodes via unicast TCP connections. The IP addresses of CSG firewall nodes are kept in a file (*/etc/dfw/fw_list.txt*) that is created by the Network Administrator on the CSM. The *csm_sendupdate()* function reads *fw_list.txt* in order to determine the IP addresses to which the updates have to be sent. Both active and backup firewall nodes are updated. Any communication errors between the CSM and the firewall nodes are logged and the Network Administrator is notified of them. For example, if ever one or more of the firewall nodes becomes inaccessible to the CSM when *csm_sendupdate()* tries to send a particular firewall update, say because of damaged cable or NIC failure on the firewall node(s), the connection error thereby generated is caught and recorded in a log file, which is eventually sent to the Network Administrator. Multicasting is not used in the sending part as UDP does not provide reliability of packet delivery. Reliability can, nonetheless, be added to a UDP multicast application by incorporating features like positive acknowledgements, lost packet retransmission, use of sequence numbers and packet re-ordering. However, adding reliability increases the program complexity [10], which is not warranted when it comes to updating such small numbers of firewall nodes per cluster. The CSM is deployed in a failover configuration such that if it fails, a backup machine will take up its place. Keepalived [11] is used for this purpose.

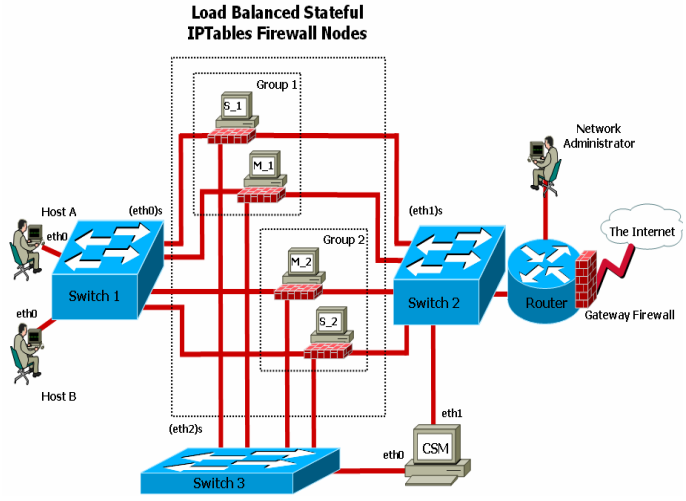

Figure 3. The 2-active-node stateful CSG as when applied to one end-user cluster.

C. *The Stateful CSG*

For our working prototype, a 2-active-node stateful CSG is utilized to secure each end-user cluster as shown in Fig. 3. The MAC, IP & virtual IP (VIPs) addresses for the above setup are as shown in Table 1. The components of the stateful CSG architecture & their configurations are given below:

- *End-user node configuration*

All IP traffic emanating from any end-user node (for example, hosts A & B) is forced to pass through the active firewall hosts M_1 and M_2. The traffic is load balanced onto M_1 and M_2 using the new *lbdnat_rr* target of Ebtables [12] that we created. *lbdnat_rr* performs Layer 2 per-packet load balancing and makes use of the Round-Robin algorithm in order to load balance IP traffic to any number of firewall hosts. Software packages like an enhanced version of Ebtables-v2.0.8-rc3, and bridge-utils-1.2 [13] are installed. A 1-port bridge is created on each end-user host, with the *stp* feature turned off. The destination MAC address of all the frames leaving any end-user host is changed to that of the firewall hosts M_1 and M_2, that is, *M_1_eth0* and *M_2_eth0*, in a round robin manner on a per-packet basis. The Ebtables rule that is used looks as follows:

```
root# ebtables -t nat -A OUTPUT -p ipv4 -o eth0 -j
lbdnat_rr --to-lbdst_rr M_1_eth0,M_2_eth0
```

- *Stateful CSG firewall node configurations*

The CSG firewall setup is made up of four PCs, two active master firewall nodes, M_1 & M_2, and two standby backup firewall nodes, S_1 & S_2. Several software packages are installed on the firewall nodes, namely bridge-utils-1.2, ebtables-v2.0.8-rc3, iptables-1.3.7, keepalived-1.1.13-6.fc5.i386.rpm, macchanger-1.5.0 (only on backup nodes), conntrackd-0.9.2 (enhanced code), libnetfilter_conntrack-0.0.50 and libnfnetlink. Each firewall node has three NICs installed. Bridge-utils is used on each firewall node to create a bridge device, *br0*, to which interfaces *eth0* and *eth1* are added. Besides their normal IP addresses, VIP addresses are also

TABLE I. MAC, IP & VIP ADDRESSING FOR THE STATEFUL CSG TEST SETUP

vrrp sync group	Node	NIC	IP address	VIP address
-	Host A	A_eth0	192.168.10.2/24	-
-	Host B	B_eth0	192.168.10.3/24	-
G1	S_1	S_1_eth0	192.168.10.100/24	-
		S_1_eth1	192.168.0.100/24	-
		S_1_eth2	192.168.100.100/24	-
G1	M_1	M_1_eth0	192.168.10.110/24	192.168.10.10/24
		M_1_eth1	192.168.0.110/24	192.168.0.10/24
		M_1_eth2	192.168.100.110/24	-
G2	M_2	M_2_eth0	192.168.10.120/24	192.168.10.20/24
		M_2_eth1	192.168.0.120/24	192.168.0.20/24
		M_2_eth2	192.168.100.120/24	-
G2	S_2	S_2_eth0	192.168.10.130/24	-
		S_2_eth1	192.168.0.130/24	-
		S_2_ eth2	192.168.100.130/24	-

assigned to these interfaces but only on the master firewall nodes M_1 & M_2. Interface *eth2* is not enslaved to *br0*, and hence, is kept separate from the travel path of end-user traffic. *eth2* is made inaccessible to all nodes except the CSM and the other firewall nodes. It is through this interface that the CSM nodes update their respective CSG firewall nodes. Ebtables and IPTables provide Layer 2 & Layer 3 packet filtering respectively. Keepalived [11] caters for high availability in the event that one of the active master firewall nodes fails. For instance, if M_1 fails, S_1 automatically takes up the role of master and becomes active. The VIP addresses on the *eth0* and *eth1* interfaces of M_1 are also re-assigned to the corresponding interfaces on S_1, thus ensuring continuity of the services offered. The master & backup nodes monitor each other by sending regular multicast advertisements [14]. The multicast traffic generated by Keepalived is kept separate from that generated by the end-user nodes for security reasons and for minimizing the loss of update messages and *vrrp* advertisements. This multicast traffic is sent via the *eth2* interface of the firewall nodes and is restricted to a separate network on Switch 3. GNU Mac Changer [15] helps in performing MAC address takeover. It is installed only on backup firewall nodes, with appropriate *notify_master* & *notify_backup* scripts defined on them [16]. It is used in conjunction with Keepalived as the latter allows us to define scripts that can be called during state transitions (from backup to master and vice-versa). For example, the *notify_master* keyword defines a script that runs on the backup node every time the latter becomes master. Since the Ebtables per-packet load balancing is a Layer 2 load balancing scheme, network traffic is load balanced onto MAC addresses rather than IP addresses. Hence, in case of a failover, the backup node has to have the same MAC addresses as the failed master node for per-packet Layer 2 load balancing to continue to work. This is achieved by changing the MAC addresses of the backup node to that of the failed master node during the state transition from backup to master using a *notify_master* script defined on the backup node. Also, a modified version of the Conntrackd [17] package is used for connection state synchronization on all four firewall nodes, which is required for stateful firewalling and PPLB to work together. When a *conntrack* entry is created in the internal cache of Conntrackd in one of the master (active) firewall nodes, updates are sent to the other three firewall nodes, both active & standby ones, of that particular CSG via multicast. All four PCs therefore have the same *conntrack* entry created in their respective Conntrackd internal cache. Consequently, in the case of a failover taking place in one of the *vrrp* synchronization groups, the new master will continue to forward traffic pertaining to already established connections as it already has the necessary connection state information. The multicast traffic generated by Conntrackd is kept separate from that generated by the communicating end-user hosts for the same reasons as that for Keepalived. It is sent via *eth2* and is restricted to Switch 3.

- *Switch Configuration*

The stateful CSG architecture requires the use of three switches. Switches 1, 2 & 3, each connect to one of the NICs of each firewall node. While switches 2 & 3 can be ordinary unmanageable switches, Switch 1 has to be a manageable one with support for MAC ACLs. This feature is used for Network Access Control so that traffic from end-user hosts is forced to pass through the firewall nodes for inspection. To this end, the switch used as Switch 1 is a Cisco Catalyst 2970 switch. A named MAC ACL [18], used for filtering traffic, is applied on a per-port basis on all switch ports for inbound direction except those to which the firewall nodes are connected.

```
Switch (config)# mac access-list extended mac1
Switch (config-ext-mac1)# permit any host M_1_eth0
Switch (config-ext-mac1)# permit any host M_2_eth0
```

The above ACL rules are used to permit only frames that have the MAC addresses *M_1_eth0* or *M_2_eth0* as destination MAC address to be forwarded by Switch 1. Any other destination MAC address will cause the frames to be dropped.

1) The Policy Handler: The Policy Handler (*/etc/dfw/fw_policy_handler.c*) is implemented as a user-space TCP application that runs on the firewall nodes. It waits for connections from the CSM. The *fw_updatehandler()* function reconstructs the firewall update file received from the CSM and applies it to the current firewall ruleset using the *system()* system call. Since the firewall update file contains IPTables rules, it is already in usable form and can thus be applied directly. Any error is caught & logged. The Policy Handler notifies the Network Administrator of errors via the CSM so that any communication between the firewall nodes and the Network Administrator machine remains secure.

D. Router Configuration

In order to obtain the full benefit of per-packet firewall load balancing, the load balancing must be supported on both sides of the firewall nodes. Hence, traffic going towards the end-user cluster from the router is also load balanced onto the active stateful CSG firewall nodes. Since the Ebtables Layer 2 PPLB scheme cannot be used on the router, the router is configured accordingly in order to perform PPLB. A Cisco 2600 series router is used. Static routing, enabling *CEF*, and using the *"ip load-sharing per-packet"* command on the router interface *Fa0/0* help achieve round-robin PPLB. The following configuration is used to load balance incoming Internet traffic:

```
# Adding static routes
router(config)# ip route 192.168.10.0 255.255.255.0
VIP_M_1_eth1 10
router(config)# ip route 192.168.10.0 255.255.255.0
VIP_M_2_eth1 10
# Enabling CEF
router(config)# ip cef
# Enabling PPLB on router's Ethernet interface
router(config-if)# ip load-sharing per-packet
```

Two equal-cost static routes are installed on the router for the end-user subnet 192.168.10.0/24. Each static route uses the VIP address applied to the *eth1* interface of one the active (master) firewall nodes as next-hop IP address. The VIP addresses are used as next-hop IP addresses such that if a master firewall node fails, its VIP will go to the backup firewall node and load balancing will continue.

E. The Stateful CSG-based Gateway Firewall

A 4-active-node stateful CSG is used to implement the gateway firewall. The main difference between the CSG used for protecting an end-user cluster and the CSG for the gateway firewall is that, in the latter implementation, the four active firewall nodes are sandwiched between two routers which are responsible for load balancing network traffic on a per-packet basis onto the firewall. Static routing, enabling *CEF*, and using the *"ip load-sharing per-packet"* command on the router interfaces help achieve per-packet round-robin load balancing.

F. IPsec

In our design, IPsec is used for securing the distribution of firewall updates from the Network Administrator machine to CSMs, and for securing error reports from CSMs to the Network Administrator machine. The paths between a CSM and its cluster's CSG firewall nodes are not IPsec-protected as they are inherently secure. This is because the firewall nodes accept firewall updates destined for their local process only from the CSM of the cluster to which they belong. This traffic is only accepted from the *eth2* interface on the firewall nodes. This interface is not enslaved to the bridge device on the firewall nodes such that it can only be accessed by the CSM. Each CSM communicates only with Network Administrator machine and the CSG firewall nodes falling under its responsibility. The Network Administrator machine's identity is checked via the utilization of digital certificates. Direct IPsec connections are not established with the firewall nodes so as not to strain them with IPsec packet processing. This processing is handled by the CSM. The latest Linux kernels provide native support for IPsec. The IPsec-tools 0.6.7 package is installed for a 2.6.20.4 Linux kernel in which all the necessary kernel IPsec options and cryptographic algorithms in the CryptoAPI have been selected. Transport mode is used to secure the host-to-host connections between the Network Administrator machine and the CSMs as both types of node are actual participating source/destination pairs – the CSMs receive the firewall updates while the Network Administrator machine receives error logs. The IPsec-tools package contains the IKE daemon, *racoon*, and the *setkey* utility [19]. *racoon* is used for the setting up of automatically keyed IPsec connections while the *setkey* utility is used for manipulating parameters stored in the Security Association Database (SAD) and Security Policy Database (SPD). X.509 certificates are used for authentication as digital certificates are difficult to forge and represent the most secure method to manage keys. *openssl* is used to generate those certificates. For testing purposes, a Certificate Authority (CA) is assumed.

1) Benchmark results: Two user-land tools, Iperf [20] and Netio [21], are used to provide some throughput estimates of the secure connections between the Network Administrator machine and a CSM. Iperf generates TCP connections, which involve requests and replies. Netio, too, generates TCP connections but sends packets of varying sizes. Two Linux machines are used in the test setup and they have the following configurations:

- Development platform – Fedora Core 5 with an upgraded kernel of 2.6.20.4.
- Pentium 4, 1.60 GHz, 256MB RAM
- 10/100 Network Interface Cards

The test parameters for Iperf are:

Window size: 214KB
Maximum Segment Size (MSS): the default for Iperf - 1460

The results obtained with Iperf and Netio respectively for different IPsec transforms are as follows:

TABLE II. PERFORMANCE RESULTS FROM IPERF FOR DIFFERENT IPSEC TRANSFORMS

Transform	Bandwidth (Mb/s)
w/o IPsec	94.1
3DES & MD5	46.7
DES & MD5	89.8
3DES & SHA1	35.4

TABLE III. PERFORMANCE RESULTS FROM NETIO FOR DIFFERENT IPSEC TRANSFORMS

Packet size	Bandwidth w/o IPsec (KB/s)	Bandwidth with IPsec (KB/s)		
		3DES & MD5	***DES & MD5***	***3DES & SHA1***
1KB	11474	5767	10386	4159
2KB	11511	5803	10588	4465
4KB	11512	5835	10918	4466
8KB	11511	5854	10991	4485
16KB	11507	5848	10983	4471
32KB	11508	5812	10861	4457

V. THREAT MODEL

The effectiveness of the design of the new stateful CSG-based distributed firewall has been assessed qualitatively against various types of threats originating from both inside and outside the network. The way in which the new security model counters these threats is described below:

1. Insider attacks

The insider attacks come in two flavors – intra-cluster & inter-cluster attacks. If a malicious end-user wants to attack another machine found on the same cluster (intra-cluster attacks), the traffic emanating from the malicious end-user's machine will be forced to pass through the CSG firewall nodes for inspection. Direct communication between end-user hosts is prevented by MAC ACLs placed on the switch connecting

them. In the case of inter-cluster attacks, the traffic from the attacker undergoes packet filtering several times along the way to its destination. The packet filtering is performed once by the CSG of each of the clusters involved in the communication. Also, since each CSG has cluster-specific firewall rules defined for both ingress and egress packet filtering in addition to the general network-wide security policy, the fine-grain access control thus achieved, together with successive packet filtering, help limit inter-cluster attacks. Hence, all these mechanisms help contain insider attacks as close as possible to the source.

2. IP & MAC address spoofing

An attacker may also try to impersonate different nodes in the cluster like end-user nodes, firewall nodes or CSMs, using IP/MAC address spoofing. This is prevented by several security mechanisms that have been put in place. Port security, one of the in-built security mechanisms provided by the 2970 series Catalyst switch, is used to prevent a malicious end-user node from spoofing the source MAC address of its outgoing packets. Port security can be set on each individual switch interface (port). Both static and dynamic secure MAC addresses can be configured. Also, the first address dynamically learned by the switch port can be converted into the secure address for that port by enabling "sticky learning". *Sticky* secure MAC addresses can then be saved in the configuration file of the switch so that the switch interface does not need to dynamically reconfigure them when the switch restarts. The number of MAC addresses that can access a particular port can be limited to one. Hence, if a malicious end-user tries to use a source MAC address other than the authorized one, a security violation will occur and a *syslog* message will be logged. The advantage of port security is that it discards packets with spoofed source MAC addresses such that the firewall nodes do not have to waste CPU cycles in processing them.

In order to guard against IP address spoofing, Ebtables rules have been formulated and installed on the CSG firewall nodes. The *--among-src* match of Ebtables allows several MAC/IP source address pairs to be defined, and packet headers are checked against these defined pairs. Spoofed packets from end-user nodes can thus be identified and discarded.

```
ebtables -A FORWARD -p IPV4 --among-src
00:01:02:03:04:05=192.168.10.2,11:12:13:14:15:16=19
2.168.10.3,21:22:23:24:25:26=192.168.10.4 -j ACCEPT
```

3. Denial of Service attacks

DoS attacks generally come under two main categories – bandwidth depletion attacks and resource depletion attacks. These two types involve consumption of resources like bandwidth and CPU cycles respectively [22]. The use of load balancing techniques in the CSG spreads the packet filtering strain over multiple firewall nodes and prevents the latter nodes from quickly becoming chokepoints. There are a variety of DoS attacks and not all can be handled by distributed firewalls. DoS attacks, which rely on IP spoofing mechanisms, can be handled quite well since IP spoofing is difficult to realize within the CSG architecture as described above. Moreover, egress filtering is performed at the very cluster level in order to throttle these types of attacks as close as possible to the source as recommended in RFC 2827 [23].

4. Packet Sniffing

Sniffing is one of the main ways attackers use to gather network-related information. The MAC ACLs on the switch help restrict multicast/broadcast traffic on a cluster. No end-user node is allowed to send multicast/broadcast traffic. This is because the only destination MAC addresses allowed in outgoing packets from end-user nodes are those of the firewall nodes. Hence, there is significantly less traffic on the switch that attackers can sniff. Moreover, putting the NIC of an end-user node in promiscuous mode will not allow sniffing of unicast traffic of other end-user nodes as the switch makes use of virtual circuits. Thus, that traffic is restricted only to the two communicating nodes.

5. Rule tampering

Rule tampering by malicious end-users is prevented as the filtering rules are not found on the end-user nodes, but rather on dedicated CSG firewall nodes. Hence, even if an end-user has root access on his machine, he will still have to comply with the security policy defined for the network if he wants to send traffic. The only rules found on an end-user machine are Ebtables rules that are used for load balancing outgoing traffic. Hence, the maximum an un-cooperating "insider" can do is to change the load balancing rule.

VI. Related Work

Several distributed firewall implementations have been developed over the years. These implementations fall in two broad categories, namely software-based and hardware-based. Some of the major software-based distributed firewalls are as follows:

The concept of distributed firewall, as expounded by Bellovin, was implemented by Ioannidis et al. [3]. That implementation later evolved into the Strongman distributed firewall Architecture [24]. KeyNote was used as an intermediate common language for translating different high-level policy languages into KeyNote policies/credentials, and helped manage access control in large heterogeneous networks [25] that made use of diverse security mechanisms. IPsec was used for user/host authenticator, secure distribution of credentials and firewall traffic protection.

Smokey [26] is a user-based distributed firewall system that was responsible for putting in place a centralized security policy on member nodes of a distributed system. The policy was distributed on a per-user basis and provided as much access as was required. A local policy manager retrieved user-specific security policies from a policy server. It then passed them to the local policy handler, which deployed them on the

host. The security policies consisted of directly usable IPChains rules.

Gangadharan and Hwang [6], [8] made use of a distributed micro-firewall approach in order to protect nodes of a Linux cluster. The micro-firewall module was made up of three components – a packet filter (IPChains), an anomaly detector (LIDS) & access logging facility (LogCheck), and it was built on each node. Each cluster formed a policy domain managed by a policy manager. The individual anomaly detectors communicated with the policy manager using mobile agents, thus forming a DIDS (distributed intrusion detection system). When an intrusion was detected by the anomaly detector of the micro-firewall on an end-user host, it was reported to the policy manager, which generated a policy update that was sent to all cluster nodes using Java-based RMI. The policy manager also notified other policy managers so that the whole network got updated to prevent the spread of the threat.

As for hardware-based distributed firewalls, the known ones include the Distributed Embedded Firewall (EFW), the Autonomic Distributed Firewall (ADF) and the Network Edge Security Distributed Firewall.

The EFW [27]-[29] is a NIC-based host-OS-independent firewall that filtered IP traffic to/from a particular host. It was managed by a central policy server and all policy server/EFW-enhanced NIC communication was authenticated by 3DES. Each policy server managed exactly one policy domain, within which EFW NICs were grouped into device sets by virtue of their function. That helped ease manageability and policy deployment. The EFW NIC had a stateless packet filtering engine provided in the EFW-enhanced firmware of the NIC, which enforced the policy rules. The 3Com 3CR990 family of NICs was used for the EFW as these NICs have on-board processor, memory & cryptographic engine, which allowed the EFW to operate independently of the host OS.

The ADF [27], [30] implementation finds its origins in the same codebase as the EFW. EFW was created first and was commercialized by 3Com. ADF was later developed by Adventium Labs, with added capabilities like Virtual Private Groups (VPGs).

The Network Edge Security distributed firewall [31], as well, was derived from the same design as EFW & ADF.

A. System Evaluation

The stateful CSG-based distributed firewall model has been compared to the above-mentioned software-based and hardware-based distributed firewalls in order to determine the various desirable characteristics that it possesses and to see how well it fares with respect to the others:

- *Fine-grained security*

The CSG firewall nodes provide both robust Layer 2/3 packet filtering using Ebtables and IPTables respectively. Network Access Control is provided by switch MAC ACLs and port security. The Strongman, Smokey and Micro-firewall architectures too provide robust packet filtering and access control mechanisms. The hardware-based models, however, do not score high in this area as they have limited packet filtering capability due to limited processing power and memory on the NIC. The NIC can be easily overloaded by network traffic even when small firewall rulesets are used [27].

- *Firewall tamper resistance*

Since the firewall rulesets are not found on the end-user hosts but rather on dedicated CSG firewall nodes, malicious end-users cannot change or delete packet filtering rules. They can only comply with the security policy defined. Only the CSG-based and hardware-based distributed firewall models exhibit this characteristic. This is not the case for the other software-based distributed firewall models, where malicious end-users working with root privileges on end-user hosts can modify/delete firewall rules.

- *High scalability*

The Ebtables Layer 2 load balancing scheme allows additional end-user hosts to be added very easily to the CSG architecture. Firewall nodes, too, can be added easily. This helps in preventing the overloading of the firewall nodes as the end-user cluster grows in size. All the other models also provide high scalability.

- *Anti-spoofing*

In the CSG-based model, packet spoofing by end-users is prevented by using features like port security on switches, and the *--among-src* match of Ebtables on the firewall nodes. Any packet with spoofed MAC or IP addresses is dropped at the very cluster level. In the hardware-based models, anti-spoofing is achieved through the inaccessibility of the NIC to end-users. All the other models use some kind of user authentication mechanism. For instance, cryptographic certificates are used in the Strongman and the Micro-firewall models, while Smokey uses tickets, uids and secure authenticated channels in order to prevent user identity spoofing and the spoofing of entities like Policy Server and Policy Manager.

- *Anti-sniffing*

In the CSG-based model, switch MAC ACLs make it impossible for end-user hosts to send multicast/broadcast traffic. Moreover, end-users cannot sniff unicast traffic between two communicating hosts as switches make use of virtual circuits. In the case of the hardware-based distributed firewall models, end-users are unable to put the end-user host NIC in promiscuous mode in order to sniff multicast/broadcast traffic because the NIC operates independently of the host OS. As for the software-based implementations, sniffing is possible by malicious end-users working with root privileges on the host, and if the network in which they are deployed does not limit multicast/broadcast traffic.

- *Low overall network load*

In our distributed firewall model, the number of secure (IPsec) connections to be made is very low as compared to those in the other distributed firewalling schemes which have to establish secure connections with every end-user host present on the network. In those models, the strain on the overall network increases with the number of protected end-user nodes present, especially during policy update broadcasts. This can be deduced easily from the bandwidth estimates obtained for one IPsec connection in Tables II & III. In our model, secure connections are made only with CSMs, which amount to one per cluster. Since fewer IPsec connections and fewer copies of firewall updates are needed, the overall network load is much lower than with the other schemes.

- *Secure & rapid real-time updating*

In the CSG-based model, upon detection of a threat, the Network Administrator can rapidly create new firewall rules so as to counter it. These new rules are sent as firewall updates to the affected clusters or to all clusters in the network if required. All the models, except Smokey, support this feature. However, the CSG-based model goes one step further than all the other models in the sense that end-user nodes that were previously offline or that have freshly been added to the cluster do not need to fetch firewall updates from servers as they are instantly protected by the CSG firewall nodes, which are online and updated all the time. Moreover, increase in the number of protected end-user hosts in the CSG-based model does not affect the speed of the updating process, as is the case in the other models. This is because the number of firewall nodes remains the same.

- *Low convergence time*

Since we have few IPsec connections to establish and a small number of CSG firewall nodes per cluster, the convergence time of the firewalls across the network is much less compared to updating every single end-user host in the network as may be required by the other models. This becomes even more evident when we have to deploy network-wide security policies. In the other models, the convergence time of the network with regard to the re-synchronization of all the host-resident parts of the distributed firewall tends to be high and increases with the number of protected end-user hosts involved.

- *Low end-user host processing strain*

In our model, the packet filtering strain is removed from the end-user hosts. Only two Ebtables rules are placed on each end-user host for load balancing purposes. This characteristic is supported by the hardware-based models as well since the packet filtering process is carried out independently by the NIC's on-board processor. On the other hand, the end-user nodes in the other software-based models are heavily taxed by the packet filtering process, which is done directly on the end-user hosts themselves.

- *Context knowledge*

Context knowledge helps achieve even more robust traffic filtering, especially at OSI Application Layer level. The firewall nodes forming part of the CSG-based distributed firewall model do not possess end-user host context knowledge. This is because the packet filtering process is done independently of the end-user hosts, on dedicated firewall nodes. This also holds true for all the hardware-based models as the NIC operates independent of the host's operating system. Only the Strongman, Smokey and Micro-firewall models are able to use context knowledge during the packet filtering process as the latter is done directly on the end-user hosts themselves.

The results of this comparison are summarized in Table IV.

VII. CONCLUSION

This paper highlights a new approach to distributed cluster and network security – the stateful CSG-based distributed firewall architecture. This distributed security model successfully addresses the limitations plaguing distributed firewalls and adds a few more desirable characteristics of its own.

Future work will look at the creation of load balancing network interface cards (LB-NICs). In order to prevent the tampering of the Ebtables load balancing rules that are found on the end-user hosts by malicious users, the Ebtables MAC-based load balancing scheme can be incorporated into a

TABLE IV. COMPARISON OF THE STATEFUL CSG-BASED DISTRIBUTED FIREWALL WITH OTHER MAJOR DISTRIBUTED FIREWALL SCHEMES

Characteristics	Strongman	Smokey	M-F	EFW	ADF	NES	Stateful CSG-based
Fine-grained security	✓	✓	✓	×	×	×	✓
Firewall tamper resistance	×	×	×	✓	✓	✓	✓
High scalability	✓	✓	✓	✓	✓	✓	✓
Anti-spoofing	✓	✓	✓	✓	✓	✓	✓
Anti-sniffing	×	×	×	✓	✓	✓	✓
Low overall network load	×	×	×	×	×	×	✓
Secure real-time updating	✓	×	✓	✓	✓	✓	✓
Low convergence time	×	×	×	×	×	×	✓
Low end-user host processing strain	×	×	×	✓	✓	✓	✓
Context knowledge	✓	✓	✓	×	×	×	×

KEY: M-F = Micro-firewalls ; EFW = Distributed Embedded Firewall ; ADF = Autonomic Distributed Firewall ; NES = Network Edge Security

tamper-resistant network interface card with on-board processing engines. This approach adopts a similar line of thought as that used in the implementation of hardware-based distributed firewalls like EFW [28], [29] and ADF [30]. These load balancing cards will have to register with a central policy server first in order to be able to function. The central policy server will perform LB-NIC group management, where each LB-NIC group will consist of all the end-user node NICs forming part of a particular end-user cluster. Thus, each LB-NIC group will have an appropriate load balancing policy created for it. The load balancing policy will simply consist of the Ebtables load balancing rules that will indicate the firewall nodes to which traffic from end-user nodes of a particular cluster will be load balanced to. The potential advantages of this scheme include:

1. *Load Balancing rule tamper resistance*

Tamper resistance of the load balancing rules will be provided as the LB-NIC will be protected from direct manipulation. This can be ensured in the following manner:

- Only the policy server can add/delete/update the load balancing policy on the LB-NIC.
- Only the policy server can disable the LB-NIC.
- All policy server/LB-NIC communication is authenticated and secured by means of cryptography.

2. *Easy addition/removal of firewall nodes in the CSG*

The centralized management of LB-NICs will also allow new firewall nodes to be added/removed in the CSG very easily. Upon such changes, the policy server can update the load balancing policy on the LB-NICs by multicasting new load balancing rules to the appropriate LB-NIC groups.

3. *No need for a dedicated load balancer*

Since load balancing is performed by the LB-NICs themselves, there is no need for a dedicated software-based or hardware-based load balancer. Also, network latency decreases as the number of intermediate components decreases.

REFERENCES

[1] V. Ramsurrun, and K. M. S. Soyjaudah, "Efficient cluster security gateway architecture for per-packet load balanced IP filtering on switched clusters," in *2006 Proc. CSNDSP'06 Conf.*, pp. 256-261.

[2] S. M. Bellovin, "Distributed firewalls," ;login: magazine, special issue on security, 1999.

[3] S. Ioannidis, A. D. Keromytis, S. M. Bellovin, and J. M. Smith, "Implementing a distributed firewall," in *Proc. 7th ACM Conf. Computer and communications security*, Athens, 2000, pp. 190-199.

[4] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. Keromytis, "The KeyNote Trust-Management System Version 2," *RFC (Informational) 2704*, Internet Engineering Task Force, 1999.

[5] Y. Bartal, A. Mayer, K. Nissim, and A. Wool, "Firmato: A Novel Firewall Management Toolkit," in *Proc. IEEE Symposium Security and Privacy*, 1999, pp. 17-31.

[6] M. Gangadharan, and K. Hwang, "Micro-firewalls for dynamic network security with distributed intrusion detection," in *IEEE Int. Symposium Network Computing and Applications (NCA'01)*, 2001.

[7] W. Li (2000). Distributed Firewall [Online]. Available: http://citeseer.ist.psu.edu/li00distributed.html

[8] M. Gangadharan, and K. Hwang, "Intranet security with micro-firewalls and mobile agents for proactive intrusion response," in *IEEE Int. Conf. Computer Networks and Mobile Computing*, 2001.

[9] W. Odom, "CCENT/CCNA ICND1 Official Exam Certification Guide," 2nd ed. Indianapolis, USA: Cisco Press, 2008, pp. 253-256.

[10] W. R. Stevens, B. Fenner, and A. M. Rudoff, *Unix Network Programming: The Sockets Networking API Volume 1*, 3rd ed., Addison-Wesley, 2004, pp. 595-598.

[11] Keepalived website (2007). Keepalived for Linux - Linux High Availability [Online]. Available: http://www.keepalived.org/index.html

[12] Ebtables website (2007). Who's behind Ebtables? [Online]. Available: http://ebtables.sourceforge.net/about.html#behind

[13] Bridge website (2007). Bridge [Online]. Available: http://linux-net.osdl.org/index.php/Bridge

[14] P. Hollenback (2005). Improving Network reliability with Keepalived [Online]. Available: http://www.linuxdevcenter.com/lpt/a/6162

[15] L. Ortega (2007). GNU Mac Changer [Online]. Available: http://www.alobbs.com/macchanger/

[16] ClarkConnect website (2007). Howtos – Clustering with LVS [Online]. Available: http://www.clarkconnect.com/wiki/index.php?title=Howtos_-_Clustering_with_LVS

[17] P. N. Ayuso (2007). conntrack-tools: Connection tracking userspace tools for Linux [Online]. Available: http://people.netfilter.org/pablo/conntrackd/

[18] Cisco Systems Inc. (2005). Configuring Network Security with ACLs [Online]. Available: http://www.cisco.com/en/US/products/hw/switches/ps628/products_configuration_guide_chapter09186a00800d84c8.html

[19] R. Spenneberg (2007). IPsec HOWTO (Revision 0.9.96) [Online]. Available: http://www.ipsec-howto.org/

[20] M. Gates, A. Tirumala, J. Dugan, and K. Gibbs (2003). Iperf version 1.7.0: Iperf User Docs [Online]. Available: http://www.itl.ohiou.edu/iperf-doc/

[21] K. U. Rommel (2005). NETIO - Network Benchmark, Version 1.26 [Online]. Available: http://www.ars.de/ars/ars.nsf/docs/netio

[22] S. M. Specht, and R. B. Lee, "Distributed Denial of Service: Taxonomies of Attacks, Tools, and Countermeasures," in *2004 Proc. 17th Int. Conf. Parallel and Distributed Computing Systems*, pp. 543-550.

[23] P. Ferguson, and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing," *RFC 2827*, 2000.

[24] A. D. Keromytis, S. Ioannidis, M. B. Greenwald, and J. M. Smith, "The STRONGMAN Architecture," in *Proc. DARPA Information Survivability Conference and Exposition*, vol. 1, 2003.

[25] A. D. Keromytis, K. G. Anagnostakis, S. Ioannidis, M. B. Greenwald, and J. M. Smith, "Managing Access Control in Large-Scale Heterogeneous Networks," *in Proc. NATO C3 Symposium Interoperable Networks for Secure Communications (INSC'03)*, 2003.

[26] R. Rubin (2002). Smokey: A User-Based Distributed Firewall System [Online]. Available: http://www.cs.berkeley.edu/~daw/teaching/cs261-f02/reports/rubin.pdf

[27] M. Ihde, and W. H. Sanders, "Barbarians in the Gate: An Experimental Validation of NIC-based Distributed Firewall Performance and Flood Tolerance," in *2006 Proc. Int. Conf. Dependable Systems and Networks (DSN 2006)*, pp. 209-216.

[28] T. Markham, L. Meredith, and C. Payne, "Distributed embedded firewalls with virtual private groups," in *2003 Proc. 3rd DARPA Information Survivability Conf.*, vol. 2, pp. 81–83.

[29] C. Payne, and T. Markham, "Architecture and applications for a distributed embedded firewall," in Proc. 17th Annual Computer Security Applications Conference (ACSAC 2001), 2001, pp. 329-336.

[30] L. M. Meredith, "A summary of the autonomic distributed firewalls (ADF) project," in *2003 Proc. 3rd DARPA Information Survivability Conf.*, vol. 2, pp. 260–265.

[31] T. Markham, and C. Payne, "Security at the network edge: A distributed firewall architecture," in *2001 Proc. 2nd DARPA Information Survivability Conference and Exposition (DISCEX II)*.