Global Information Assurance Certification Paper

Designing a Secure Network for Fortune Cookies

Security Through Paranoia

GCFW Practical Version 4.0

Author: Frank Sweetser
Date: Friday 17th December, 2004

Contents

List of Figures

Abstract

The first part of this paper shall give an analysis of adding a wireless network to an already existing network from a security point of view. The rest shall give a policy for designing and configuring a network for GIAC, a company that deals in buying and selling fortunes for fortune cokies, followed by an actual configuration for the central firewall derived from the policy.

1 Wireless Integration

This section will give an analysis of adding wireless to the GIAC Enterprises network. We will assume that the infrastructure as described in Section 2 is already in place and operational. We also assume that management fully understands and has evaluated all of the risks and benefits discussed here, and decided that it is worth the investment (because, as we all know, upper management would *never* push through a bad IT decision based on poor or inadequate information).

1.1 Attacks

When constructing a traditional wired LAN, one can typically assume that ensuring the physical security of the facilities through the usual means (locks, security guards, alarm systems, etc) includes the physical security of the network. After all, routers, switches, and cabling can be safeguarded from physical access by locked doors just as well as any valuable physical asset in the offices. Once the physical security is ensured, access is restricted to known points of connectivity which can be secured using devices such as firewalls.

Then along came wireless networking. Wireless signals are invisible, pass through many solid barriers such as walls, and can be readily picked up or generated by anyone with the right equipment. The equivalent of a network cable has been invisibly extended right out from the data closet, into the cafeteria, the lobby, and possibly even the parking lot and your next door neighbors (6, p21). While you can try and stop attackers from successfully connecting to this invisible cable with various authentication methods, the only way to stop them from trying to connect or even just listen is to turn the wireless network off. Since that's not an option, we will start by reviewing the basic attack types, and then the methods of mitigating them.

1.1.1 Denial of Service

There are two main categories of denial of service (DoS) attacks. The first is the simple brute force method: turn up the power on a transmitter in the same frequency band, and transmit massive levels of random noise. The end result of this is that legitimate network traffic gets swamped out like whispers in the middle of a rock concert. Clients lose their association with the access point (AP), and all network traffic stops. With the right RF gear, the source of the noise can be located, and once stopped, the network should quickly return to normal functionality.

The other, more complex method of attacking WiFi involves exploiting weaknesses in the lower levels of the 802.11 protocol family. It is an unfortunate fact that 802.11 management frames are neither encrypted nor authenticated (6, p335). This makes it trivial for an attacker to generated spoofed management frames, such as Disassociation frames. By sending out such a spoofed frame with the source address of a given APand a destination address set to the broadcast address, an attacker can knock most wireless clients off of a given AP. Some newer cards are getting smarter, and will only respond to their own address, not the broadcast address. By continually generating a stream of these messages, the attacker can keep the wireless network in an unusable state for an extended length of time.

The one positive point to remember about DoS attacks is that while they may render the network unusable, they also should not grant the attacker any elevated level of privilege, or leak any sensitive information. While this may be small comfort in the middle of an attack that is busy shutting down all work in the warehouse, it does mean that there is unlikely to be any permanent damage to recover from once the active attack ceases.

1.1.2 Passive Traffic Interception

In a wired LAN, assuming that all hosts are secure, preventing an intruder from sniffing traffic is quite simple - keep him away from the network! Once wireless is brought into the picture, however, you've already done

half of the work for the attacker by shipping all traffic on the wireless LAN outside of your physical boundaries. Any attacker within range can easily intercept and store all traffic. Modern antennas only make this problem worse. At the time of writing, for around $50 USD, you could purchase a cheap directional antenna that claims a 12 dbi gain(23) - approximately 16 times the signal level you would expect to find with a typical omnidirectional antenna as found in most WiFi cards. In short, not matter how far you think your wireless network extends, you have to assume that an attacker can pick it up from even further away than that. Once the attacker can pick up and record those signals, any traffic on the wireless network is at risk of being decrypted and leaked.

1.1.3 Unauthorized Association

Once an attacker is able to receive signals from your wireless network, it is only safe to assume that the inverse is also true: that the attacker is able to transmit signals that will get received by your wireless network. Once the attacker can send those signals, he can attempt to associate with an AP, and if successful, become a fully connected client on the wireless network.

Once he is at this stage, the attacking computer is, at the IP layer, trusted as much as any other legitimate wireless client [1]. By virtue of that magic invisible network cable, the attacker has skipped right over all of the usual internet facing firewalls and sensors, and right into the soft chewy network core. Unless special measures are taken to distrust wireless clients, the only security left to stop the attacker is the end host security.

1.1.4 Man in the Middle

When an end user plugs into a wired drop, there is typically a high degree of confidence regarding where the other end of that cable goes. Unless an attacker has gained significant physical access, the connection almost certainly goes to a trusted piece of network equipment. With wireless, however, that pesky invisible cable proves to be as difficult for the client to reliably trace upstream as it is for the AP to trace downstream.

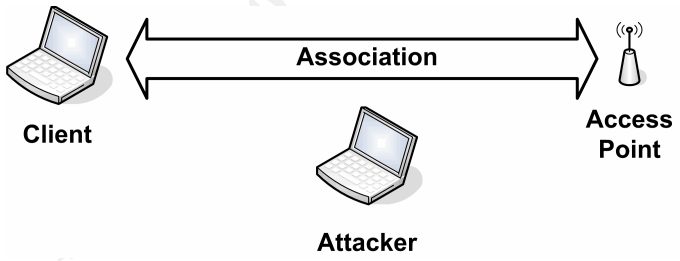

Figure 1: Attacker chooses victim

The ambitious attacker looking to try a man in the middle attack first selects a client to be the victim. The attacker, the victim, and the AP the victim is associated with must all be able to hear each other.

[1] While it is true that he may not be given a legitimate IP address, it is a fairly straightforward matter for the attacker to watch the network traffic for a bit, and then pick a valid address to use. Either way, the wireless network has been penetrated.

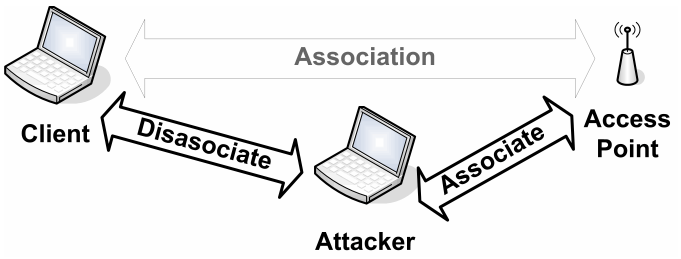

Figure 2: Attacker inserts himself between client and AP

The attacker first spoofs a disassociate message to the victim. Since 802.11 management frames are unauthenticated, the victim has no way to know the message was spoofed, and so it disconnects and begins searching for a new AP to associate with. At the same time, the attacker associates with the AP, either by impersonating the victim and thereby hijacking the clients session, or by creating a new one.

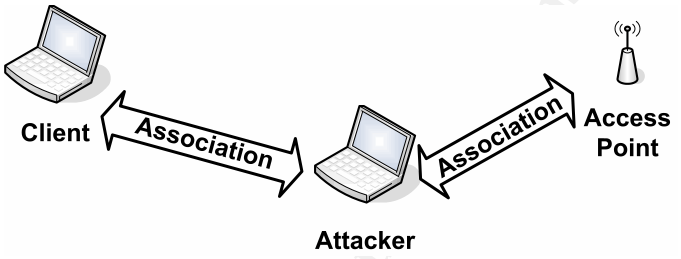

Figure 3: Successful MITM attack

If everything is successful, the attacker is now acting as a relay for all traffic between the AP and the victim client. Since performing this attack requires that the attacker already have all encryption keys, the attacker can now view, insert, and modify all traffic on the wire, without the victim or AP being aware anything is wrong.

1.2 Wireless Security Methods

Luckily, none of these attacks are completely new categories. While the world of wireless may give them some new twists, these attacks all have counterparts in traffic that flows across any uncontrolled network, such as the internet. Likewise, the same solutions can be adapted to bringing a modicum of security into the wireless world. All of the solutions presented here have the same three central goals.

Authentication Ensure that a given datagram is really from the entity it claims to be.

Confidentiality Ensure that an attacker cannot gain unauthorized access to data transmitted over the network. Since there is no reasonable way to prevent an attacker from receiving transmissions from a wireless network, this implies that the data must be encrypted to prevent the attacker from capturing the original data.

Integrity Ensure that an attacker cannot modify a message, either by modifying it in transit, or capturing and replaying the modified message, without the modification being detectable.

1.2.1 WEP

Bruce Schneier writes “There are two kinds of cryptography in this world: cryptography that will stop your kid sister from reading your files, and cryptography that will stop major governments from reading your files”(13, p19). While designed to give reasonably good protection, Wired Equivalent Privacy (WEP) has unfortunately ended up falling squarely into the kid sister category.

WEP was the first attempt at providing wireless layer encryption for WiFi networks. It uses the RC4 algorithm for encryption, originally with a 40 bit key size, though 104 bits is currently standard. Although vendors currently market WEP as 128 bit encryption, 26 of those bits are actually an Initialization Vector (IV) that is transmitted in the clear in each packet. This value is randomly generated on a per packet basis, appended to the secret key for encryption, and transmitted in the clear to the remote endpoint. This prevents two identical packets from being encrypted to identical ciphertext, making cryptanalysis that much harder. The remaining 104 bits (or 40 in the case of 56 bit encryption) is left as the actual secret.

Right out from the gate, WEP leaves much to be desired. It fails to define any key distribution method whatsoever, leaving administrators and end users to plow through the painstaking and error prone process of inputing long keys in manually. And even when everything worked right, 104 bit keys can be brute forced on modern computing hardware.

The obvious short term answer to this was to increase the key length. However, two papers successfully attacked the mechanics of the WEP design. First, in 2001 Smith(22) demonstrated that by analyzing packets with identical IV values, an attacker could build up a table of the key streams for each IV. This means that the attacker is now able to encrypt and decrypt any packet that has an IV that is in that table. Storing every possible IV value for a given WEP key would take about 23Gbytes of storage space - well within the realm of modern hard drive sizes(6, p98). Then, Fluhrer et al.(14) finished off what was left of WEP by detailing an algorithm capable recover the original shared key from duplicate IV values. Even worse, the algorithm scales linearly with the length of the key, meaning that lengthening the key does little to slow the attacker down. In the final evaluation, no one interested in anything more substantial than trivial, kid sister level security measures should rely on WEP to do the job.

1.2.2 Shared Key Authentication

An optional component of the original WEP design was to also use the shared secret for authentication via a simple challenge/response protocol. Unfortunately, WEP fails to do so in a secure fashion, with results every bit as dismal as its attempts at data privacy.

Shared key authentication starts when a client requests association from an AP. The AP sends back a random challenge string. The client is expected to XOR the string with a value derived from the shared WEP key using RC4, and send the encrypted result back to the AP, which compares it with the results of its own encryption. If the two match, access is granted. Unfortunately, in the process, the network has just given both the plaintext and the ciphertext of the message to any listening attackers. By XORing these two together and using the same IV, the attacker can trivially determine the value needed to encrypt *any* challenge string, and thereby pass the authentication.

As if this weren’t already enough, it is also worth noting that neither WEP nor shared key authentication include any kind of authentication on subsequent packets. This means that if an attacker is able to quickly knock a valid client that has already passed authentication off without letting the client explicitly disconnect, the attacker can hijack the authentication credentials of the client simply by spoofing the MAC address. Like the rest of WEP, shared key authentication is almost worthless from a security standpoint.

1.2.3 802.1X Authentication

One of the methods used in attempting to fix some of the problems of WEP was to implement 802.1X authentication in a wireless network(3). 802.1X was originally designed to allow the use of EAP authentication to control access to physical, wired switch ports. In short, the client send a request to the authenticator (the switch). The authenticator forwards the request along to an authentication server (typically a Radius server), and based on the reply, either allows or denies access. Although the actual specification includes no provision for transmitting key material, most implementations now include extensions for doing so. This solves the problem of reasonably robust access control and key management, and mitigates the vulnerabilities in WEP by allowing re-keying on a frequent basis to minimize the risk of IV reuse.

From the perspective of both administrators and end users, this is an immediate gain. There is no longer any need to manually rotate keys, as this is now handled automatically. The biggest limiting factor is that not all wireless clients will necessarily support 802.1X and an appropriate authentication back-end, such as TLS identity certificates.

Getting back to the technical side, there are still flaws that remain. The first is that there's nothing in any of the standards about how to generate and exchange key material. While it is true that vendors have implemented methods to do this, the lack of a standard opens the door for incompatibilities between different vendor products. The second, more serious problem, is that 802.1X only handles the initial authentication. Once access is granted, we are once again vulnerable to the man in the middle and 802.11 level denial of service attacks. And finally, it makes no change whatsoever to the actual encryption algorithm used. While it may lessen the likelihood of a successful cryptographic attack by drastically reducing the length of time that a given key is used, the original flaws still remain.

1.2.4 MAC Address ACLs

Many wireless access points support an administratively defined list of MAC addresses to be allowed to connect, with all other addresses denied. Unfortunately, all current 802.11 standards, including the new 802.11i standard, transmit the management layer headers unencrypted, including the transmitting MAC address. This has the unfortunate effect of making recovery of valid addresses trivial. Once a valid address is captured and selected, it is a simple matter to reconfigure an adapter via software to present itself with that MAC address, bypassing the ACL completely.

While ACLs are useful to prevent people from accidentally connecting to your network, it will do little more than slow down a determined attacker. In addition, employing this method requires that the administrators maintain an up to date list of all valid mac addresses, which can be very difficult and time consuming in a large enterprise network.

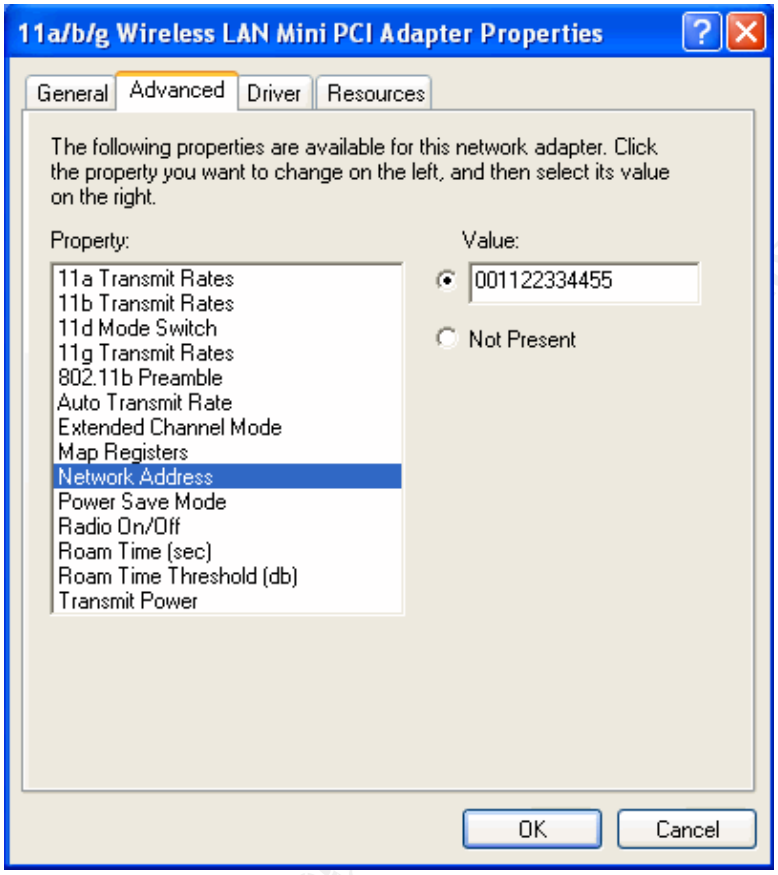

Figure 4: Modifying the MAC address under Windows XP

1.2.5 RSN

In response to the serious security flaws in the original WEP standard, the 802.11 working group recently approved the Robust Security Network (RSN). The three most fundamental architectural changes are the addition of the Advanced Encryption Standard (AES) cipher[2] as an alternative to RC4, a 48 bit nonce (equivalent to the IV in WEP), and instead of directly using the master key, generating a hierarchy of single purpose from the master key. Between these changes, all of the known critical attacks on WEP lans are nullified.

By adding AES, the 802.11 group acknowledged that while RC4 cipher can still be effective when carefully used, it is also possible to cause problems by misusing it, as happened with WEP. While switching to AES does require new hardware, this cost was seen worth the benefits of obtaining the latest and (hopefully!) most secure cipher currently available. In addition, it was expected that there would be a long, gradual transition period where older hardware would be replaced by newer hardware capable of both WEP and RSN operation, until enough had been replace that WEP could be shut off when no one was left using it.

The other major change was in how the master key is used(6, p199), and a larger nonce value. In the case of WEP, the master key is directly used in a number of operations, including encryption and decryption. This allows for an attacker to directly mount attacks on the master key. In RSN, the master key is never

[2] Also known as Rijndael, published as FIPS-197(7)

directly used on any data that is transmitted in plaintext. Instead, it is used as to generate a hierarchy of different keys, each of which is dedicated to a specific purpose, such as encryption or authentication, and is never reused for another purpose. In addition, the 24 bit IV in WEPhas been replaced by a 48 bit nonce, an increase which very nearly eliminates the possibility of duplicate nonce values for a given encryption key. To further reduce the odds, the encryption key is unique to each client, meaning that even if two stations use the same nonce will not give anything away to an attacker. This combination ensures that none of the devastating attacks against RC4 in WEP can easily be adapted to AES in RSN. The one notable exception is that management frames are still neither encrypted nor authenticated, leaving RSN still open to DoS attacks.

From an end user point of view, there is virtually no change beyond the strong encouragement of using 802.1X authentication. While RSN can be used with a static, manually distributed master key, this leaves the network open to a dictionary attack on the master key. Beyond that, the only drastic change is the unfortunate fact that all older hardware and drivers must be replaced with ones that support RSN. If suitable hardware can be obtained, however, this is far and away the most secure wireless encryption and authentication method currently available.

1.2.6 WPA

RSN was designed from the ground up, utilizing all new hardware, with the anticipation that a slow, leisurely upgrade from WEP would be a viable option. Then along came Fluhrer and tools like airsnort(20), which made it trivial for any attacker to quickly penetrate WEP with off the shelf hardware. As an interim fix, the core operating mode was extracted from RSN, modified to work with RC4 hardware, and rewritten into Temporal Key Integrity Protocol (TKIP). The mode of operation that utilizes TKIP became WiFi Protected Access (WPA). While TKIP still must utilize RC4 in order to run on existing hardware, it takes the best pieces of RSN to vastly increase its security over WEP.

First, it uses the same key hierarchy method as RSN. This stops most of the attacks against RC4 cold by preventing key reuse. Secondly, it increases the IV from 24 to 48 bits, and mandates that it be incremented rather than picking a random value per packet, to reduce the possibility of IV collision between multiple clients. Finally, it includes a reasonably secure message integrity checksum (MIC) in each packet to reduce the risks of replay and forging attacks. While there are a few known theoretical attacks against the MIC used in WPA, WPA ends up being only moderately less secure than RSN, and far more than WEP.

The end user will not see any visible difference in operation between RSN and WPA. Both strongly encourage the use of 802.1X to prevent dictionary attacks against static master keys, and both require explicit driver and operating system support, though support for WPA is far more widespread. The greatest advantage of WPA is that existing hardware can often be used with nothing more than a software update, preventing the need to find cutting edge hardware with AES support.

1.2.7 VPN

One viable option is to completely give up on wireless security altogether. Don't bother with trying to secure the wireless layer at all, and instead simply treat it as a thoroughly hostile environment filled with an unknown collection of hosts - in other words, just like an internet connection. And like an internet connection, the way to securely let selective hosts in is with an authenticated VPN solution.

In the case of an untrusted wireless link, the incoming firewall must be locked down very tight. Since only trusted hosts using the VPN are expected, everything that is not critical to the operation of the VPN must be denied. In the case of a typical IPSec based VPN, for example, all that must be passed are the appropriate IP protocols to the VPN server, UDP port 500 for IKE. DNS and DHCP services can be provided by bastion hosts in the same subnet as the wireless hosts.

The single greatest advantage of using a VPN solution is flexibility. Since authentication and encryption are not tied to a particular wireless specification, requiring specific hardware, the administrator can put in

place a commercial IPSec VPN, such as Nortel Contivity, an SSL based one such as OpenVPN, or in simple cases, allow permit encrypted protocols such as HTTPS and ssh out. The disadvantage is ensuring client support for all devices. While wireless encryption methods are explicitly standardized, there is a much wider range of incompatible VPN solutions. Embedded devices, such as handheld barcode scanners, will most likely prove to be difficult to ensure client compatibility on.

Even once client compatibility is handled, this option will have the greatest impact on from the point of view of the end users. Requiring the VPN client to be up means that the end users will have to perform an extra authentication step. How much of an impact this will make depends upon the technical competence of the end users, and how much training is available.

1.3 Conclusions

Wireless, like any other extension of the network, represents a trade off between utility and security. While the original design attempted to include a reasonable level of security in the form of WEP, this ended up being little more than an embarrassment to the designers. Other options have since appeared to compensate, including WPA, RSN, and VPN technologies, and though they are not perfect, can offer a much higher level of security. As with any such measure, a case by case analysis is necessary to weight the benefits against the risk, but with modern wireless security methods, all but the highest security needs can most likely be fulfilled with readily available hardware and software.

2 Security Architecture

This section will describe the overall architecture of the GIAC network. It describes all of the critical security components, how they interact, and a justification of each decision.

2.1 Design Principles

When designing something as complex as a secure network, it is always helpful to start with a few simple design principles. Any subsequent design decisions should then fulfill one or more of these principles (or at least not contradict them!).

Layered Defense No single layer of defense can make a network secure, short of turning it off. Put up a port filtering firewall, and applications run on random ports. Put up a layer 7 filtering router, and viruses and trojans start sneaking through with more sophisticated encoding and encryption. Block all traffic, and users will walk viruses right by the smartest firewall and virus scanner in the world as they bring their laptops and files in from home. Therefore, *every* component from the end station to the edge router must be selected and configured with security in mind.

Minimal Access When configuring any kind of access control, the default policy should be to deny all access. From this starting point, each "allow" clause added must have a clear business or technical justification, and should be made as specific as possible.

Support Primary Business While it may seem rather obvious, it is worth explicitly stating that security measures must not interfere with business critical operations. At best, any measures that do not will be bypassed by users to get their job done. At worst, they will stop one or more business processes, causing possible financial or legal financial repercussions.

Never Send Passwords in Cleartext No matter how much the network is secured, there is always the possibility that one or more systems will be compromised. Once under the control of an attacker, it is not uncommon to find such systems sniffing the local network looking for passwords. To mitigate this, passwords and other such highly sensitive information should always be transmitted over the network encrypted whenever practical.

Sanitize All Data The fundamental purpose of any security measure is to deny the "bad", while still allowing the "good". A typical firewall achieves this by comparing network traffic against a set of rules, and accepting or rejecting packets based on those rules. Many attacks are payload based, however. To help defend against those, all incoming data should be validated at all layers, from IP to the application layer.

Open Standards and Software Software, and the standards that govern their interfaces and file formats, are the fundamental blocks of IT. If a company were to get locked into a single vendor, or a single proprietary vendor standard, that company would be at the vendors mercy. By using open source software, or at least software that is limited to open standards, you gain the ability to pick and choose the vendor you would like to use, and switch vendors later without starting over from scratch.

2.2 Network Topology

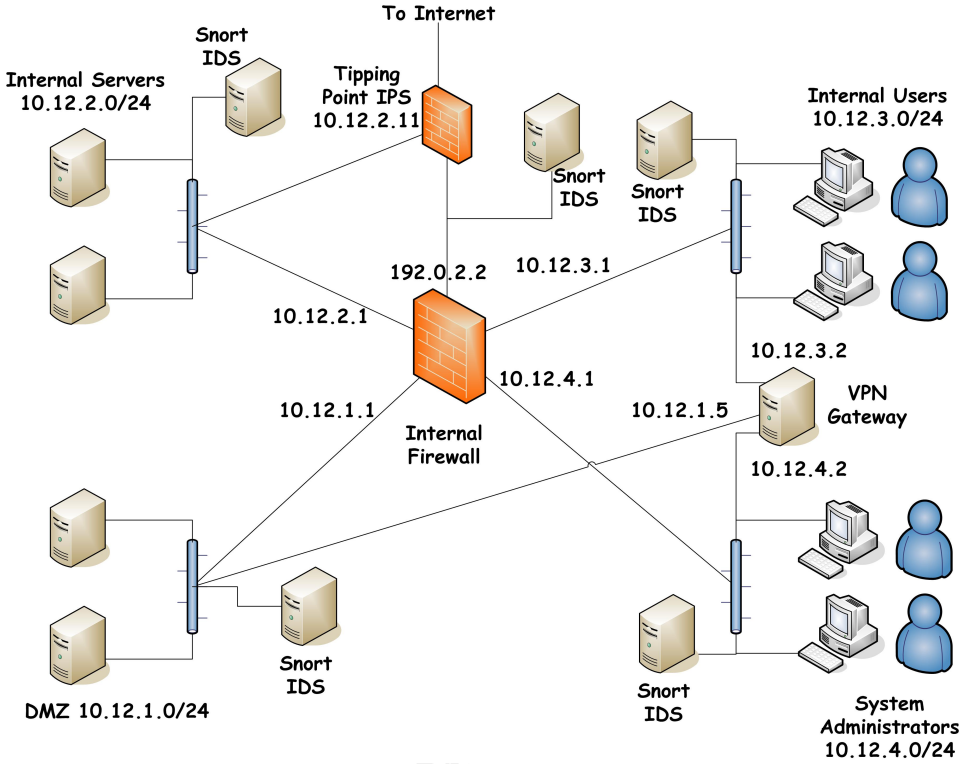

Figure 5: The GIAC Network

This network is built using a simple, straightforward topology. First in line from the internet is a router configured to filter out any traffic that is not destined to one of the outward facing IP addresses, or to any service not explicitly advertised. After that is a Tipping Point IPS(21) to handle application layer attacks. Finally is the central router and firewall system. This provides all connectivity between the different subnets, and is configured with a paranoid default deny policy.

There are four subnets behind the firewall. Each one is designated for a different purpose, and so has different rules in the firewall rule base.

2.2.1 Server Subnet

This subnet is designated for internal servers. This includes the majority of the most critical servers, including the application and database servers responsible for handling the actual fortune database. This subnet must be monitored particularly well, and is filtered from all incoming internet traffic. All outgoing traffic is proxied or relayed through a server in the dmz subnet.

2.2.2 DMZ Subnet

This is the subnet for hosts that must be visible to the internet. Each individual host is subject to bidirectional NAT by the firewall to provide connectivity from the internet. These systems have the greatest exposure, and so must be kept carefully locked down.

2.2.3 Users Subnet

This subnet is for standard, unprivileged workstations. This includes those used by administrative assistants, sales, marketing, management, development, etc. These machines are expected to be loaded with the widest range of software, and used by the least technically inclined users. As a result, this is the least trusted out of all of the subnets.

2.2.4 System Administrators Subnet

Since none of the machines in the users subnet are trusted enough to manage the servers, a different subnet is required. The admin subnet is designated for machines that are strictly limited to managing the server and other network infrastructure: ssh to servers, windows administration tools, nessus, etc. This means that these machines are not to be used for any casual web browsing, email, IRC and the like.

2.3 System Classifications

From a network security perspective, are 6 relevant classifications of systems.

2.3.1 External Customer and Partner Systems

Customers include anyone who is buying fortunes electronically, both small quantity sales from the general public and bulk sales from corporate customers. This includes translators, resellers, and suppliers. None of these groups are granted any special access to any private servers. Instead, all communication is limited to the public HTTP servers (in reality, the reverse proxies). The web applications are then responsible for authenticating the remote party, and granting the appropriate level of application level access based upon the provided identity. This may range from anonymous public access, such as a potential customer browsing the public web pages, to a corporate customer performing automated bulk purchases, to a translator who has extensive read and write access to the fortune database.

2.3.2 Outward Facing Servers

This category is defined as any server that must be directly reachable via a publicly routable IP address from the Internet. This includes public web, mail, and DNS. These machines, and these machines only, must be located in the DMZ subnet.

2.3.3 Internal Facing Servers

This category is defined as any server that does not need to be reachable from the Internet, only from internal clients and servers. This includes the internal mail server, cookie database servers, and file servers.

2.3.4 Non-Privileged User Workstations

This is any workstation used by a user who is not considered a system administrator. This includes sales, marketing, developers and their test labs - any user that does not require administrative access to any production servers.

2.3.5 System Administrator Workstations

This is any workstation used by a user who requires administrative access to servers. Note that any system that falls into this category should only be used for administrative tasks, and also be complemented by a non-privileged system for normal email, web browsing, and other day to day tasks. This subnet also includes one system with restricted dial in access for remote management in case of severe network outages during off hours.

2.3.6 VPN Users

Employees of GIAC can establish a VPN tunnel into the network. Once authenticated, they are classified by their user ID, and based on the results, the endpoint of their tunnel is placed into either the non-privileged user or system administrator subnet. This ensures that any given user has the same privileges when logged in through the VPN that they do when logged in locally.

2.4 Security Components

These are the devices that are intended primarily for security functions, such as filtering and monitoring.

2.4.1 External Router

The external gateway is the networks first line of defense. Rather than try and use this device as a full firewall, it is only set up to filter obvious known bad traffic, leaving the more advanced filtering to the dedicated firewall. This includes filtering traffic on the external interface with a source address of the internal networks, traffic on the internal network with a source address of anything other than one of the internal networks, and any RFC1918 traffic.

2.4.2 Tipping Point IPS

One of the principles used in designing this network was that all data should be sanitized. Unfortunately, routers and firewalls tend to be very bad at this. The best that most of them can be expected to do is to filter bad network and transport layer states, such as fragmentation attacks, from end hosts with buggy IP stacks. Any application layer attacks, such as directory traversal over buffer overflow attacks against CGIs, will go right through, so long as they are launched against a legitimately allowed port.

To mitigate this general problem, a Tipping Point UnityOne-50 has been installed between the primary firewall and the internet at large. By matching and filtering traffic against a large database of signatures from all layers, this helps to filter attacks that would slip through a normal firewall. This kind of protection is particularly helpful during the critical window between a vulnerability being discovered, and the vendor releasing a security patch. Instead of waiting for the vendor, an administrator can productively add a filter to block the attack, reducing the exposure of vulnerable hosts. Although it does not count as an excuse to skip patching altogether, it makes missing patches much less likely to be an immediate disaster.

2.4.3 Primary Firewall

For this network, it was decided to use a commodity x86 server running Fedora Core 3 Linux with the 2.6 kernel, netfilter, and Shorewall(5) to provide the central routing and firewalling capabilities. In this particular case, a Dell Poweredge 2650 server with three dual port Intel 1000MT gigabit ethernet adapters was chosen, though any Linux compatible hardware with the requisite ethernet interfaces would also work. While this may

appear to be an odd choice, compared to the usual array of Cisco and Juniper options, it does have a few advantages over those proprietary ones.

Choice of Hardware Platforms Linux runs on a wide variety of readily available platforms, providing the same advanced networking features on all of them. Changing vendors does not mean that the firewall would have to be re-architected at all.

Host IDS Most routers don't have any built in IDS capability to monitor their own operating system and configuration. Typically, the most one can expect is logging capabilities to monitor events on another system. With Linux, however, there is a whole range of such packages, such as Tripwire and AIDE to monitor system integrity.

Configuration Revision Control Shorewall is configured with a handful of flat text files. By putting these files under a standard version control system, such as Subversion or CVS, a complete history with accountability of all configuration changes can easily be maintained.

Powerful CPU Many modern routers are limited in their advanced capability by CPU power. Modern servers, equipped with multiple 3Ghz or faster CPUs and gigs of memory, can better afford to throw raw horsepower at problems like complex rule bases, huge state tables, and content based filtering.

It should also be pointed out that the single greatest limitation of using a commodity Linux system as a router is scalability. If the network expands beyond the handful of physical ports that are required here, a transition to an actual router platform will have to be made. Given the current size of GIAC, however, and the fact that it's primary commodity, fortunes, should require little bandwidth per transaction, the Linux router should prove to be more than adequate for the foreseeable future.

2.4.4 VPN Server

As already stated, the purpose of network filtering technology is to sort the good from the bad. One of the primary benefits of adding a VPN to a network is the ability to perform filtering not just on what the traffic is, but who requested it. This allows, for example, a system administrator to authenticate and gain the ability to open ssh sessions to servers, from anywhere on the internet, regardless of her source IP address. In addition, this traffic is typically encrypted and authenticated, ensuring data privacy for whatever goes over the VPN.

For this network, the OpenVPN(24) package running on a Linux server as been selected. It uses the stable and well tested OpenSSL library to provide encryption, including the AES cipher. It is capable of multiplexing multiple clients over a single UDP port. By using UDP instead of IPSEC, it avoids the issue of NAT breaking AH mode by modifying the IP headers. It also has clients for both Windows and UNIX platforms, including OSX.

2.4.5 Snort/Argus IDS

Unfortunately, no network can ever be said to be completely secure against all attacks. Instead, we must aim for the next best thing, which is to continually monitor the network to find any unexpected events, be they a hacker breaking in, a virus attempting to spread, or an ignorant user installing an unauthorized peer to peer application. To accomplish this, a Linux system is installed in each subnet and configured to passively monitor all traffic with the Snort and Argus applications.

Snort(12) is a signature based intrusion detection system. It is fast, lightweight, and has an extensive rule database that is written in a simple language that allows the local administrator to easily add custom

rules. It shall be configured to perform logging to the local filesystem in the unified output format, which is then processed and forwarded to a MySQL database server via Barnyard for analysis and reporting.

Argus(11) gathers and reports on IP flow statistics. While less useful for real time intrusion detection, it is invaluable in performing forensics. For example, if a server has been compromised, the Argus logs can be used to generate a list of all external IP addresses that have communicated with the host in a given time period. All Argus logs shall be periodically copied to a central server for storage and analysis as needed.

The central server is a Linux system responsible for collecting all snort events captured by the network of sensor probes. All events are stored in a MySQL database. This system also runs an Apache server configured to allow password protected access only from the System Administrator subnet, which allows administrators to view and analyze events using the Placid(4) CGI application.

2.5 Services

Even the most perfectly configured firewall in the world will not stop all attacks from getting through. Every business will have critical services such as web, email, and DNS that must be let through for the business to function. By letting these services through, the machines hosting them are now threatened by application layer attacks, such as buffer overflows or carefully corrupted requests. While this risk can be greatly mitigated by installing a filtering IPS upstream, the end station must also be configured to protect itself as best it can. This section will go over a few more the more critical serves that have a direct impact on the security of the network as a whole, and mention ways to configure them that complement the firewall policy.

2.5.1 HTTP Servers

This is the collection of systems that are responsible for handling all web traffic. The most important feature of this system, is that the external facing machines that customer web browsers connect to do not run the actual web servers or contain any content. Instead, the external systems run the Pound reverse filtering HTTP proxy(15). Pound has two very important features that add to network security.

First, it will verify incoming requests for correctness. While this does not guarantee that a given request is not malicious, it does limit the number of attacks that will ever make it to the real web server. While in general, more comprehensive content filtering is better done at the Tipping Point system, since the Pound proxy is configured to decrypt the HTTP payload before forwarding the request, it can validate requests in HTTPS sessions.

Second, Pound does not simply forward along HTTPS requests. Instead, it acts as an SSL endpoint, decrypting the requests, and passing the request along in plaintext to the back-end server pool. From a security standpoint, this constitutes a clear trade off. On the one hand, potentially sensitive information is now being transmitted in the clear, even though it is limited to the local network only. The advantage is that the plaintext is now available for inspection by the Snort IDS sensors. Without this extra decryption and relay, any application layer attacks against the web server or any CGI applications running on it would be completely invisible to anything but the server under attack.

After Pound has done its best to sanitize the HTTP requests, they are passed along to the back end servers. These shall be Linux servers running the Apache HTTP server. Since it is relatively well shielded from the external network, it should only have to worry about application layer attacks on any CGI applications it hosts.

2.5.2 DNS

External facing DNS servers can, if misconfigured, act as an information leak(1). They are configured with a complete list of valid system names and IP addresses. If an attacker can mine this information, it quickly

narrows down the potential targets to a list of valid, live IP addresses that are worth concentrating on. If a single server is configured with both the internal and external DNS views, this can provide the attacker with the internal IP topology and the mapping between internal and external addresses, further aiding the attack.

To minimize these risks, it shall first be configured only with records for systems that must be accessible from the Internet. Since this system is not intended to be queried by any internal systems, there is no need for this server to have the data in the first place. Keeping it off ensures that even if the system were compromised or a data leak were to later be discovered in the BIND software, internal addresses can not be disclosed from this system.

Second, zone transfers must be disallowed by default. Even though the records on this system are limited to ones considered “safe” for disclosure to the Internet, allowing zone transfers makes it that much easier for an attacker to generate a list of valid targets. Zone transfers must be restricted to a list of any off site secondary DNS servers, first by an IP address ACL, and then further secured by use of TSIG cryptographic signing on all zone transfer requests. The use of TSIG signing ensures that simple IP address spoofing will not enable an attacker to successfully initiate a zone transfer.

Finally, recursive queries must be disallowed. Recursive queries are only required for servers that will be answering requests for domains for which it does not already have authoritative data. Since this server is not intended for use by any end stations, but instead only for Internet users seeking data about public `giac.com` addresses, there is no reason to support recursion.

The internal DNS servers provide addresses that map to the internal network addresses suitable for internal use. Since this system will be providing service for internal clients, it will have recursive queries enabled. It should still have zone transfers disabled for everything except any slave servers.

2.5.3 Email Servers

The external SMTP service shall be provided by a Linux server, running either the latest version of sendmail (16) as provided with the Linux distribution, or, if necessary for security reasons, the latest version as released by the sendmail authors. It is responsible for handling incoming and outgoing mail from the internet. Firewall rules prevent any other system from creating outgoing or receiving incoming SMTP connections.

This system shall not be configured to perform any final delivery of any email. Instead, all received email that is addressed to a valid `giac.com` email address shall be relayed to `smtp.giac.com` for further processing and delivery.

All incoming mail will be run through the inline virus scanner ClamAV (2), and the spam analysis system Spamassassin (17). Virus updates should be automatically downloaded at least daily through the outgoing proxy. Static spam rules change more infrequently, and may be updated periodically at the discretion of the system administrator.

Internal email services shall be provided by a separate Linux server. This separation means that even if the externally facing email server is compromised, it is still limited to directly attacking only other hosts in the DMZ. In addition to accepting outgoing mail from internal machines via SMTP, this server will also provide POP3 and IMAP services to allow internal clients to read mail. This is the only system that is permitted by firewall rules to create or receive SMTP connections to and from the external mail server.

All outgoing mail will be run through the inline virus scanner ClamAV with up to date virus definitions.

2.5.4 Outgoing HTTP Proxy

In addition to the incoming proxy, there is a separate proxy required to make outgoing requests. There are several reasons for requiring end user workstations to use a password protected HTTP proxy. The first is to monitor the activities of employees. Examining proxy logs can help ensure that employee web browsing activity does not violate any relevant company policies. The second reason is that by enabling caching on the proxy, there is a potential for reducing bandwidth on the internet link.

The third reason is to attempt to slow down viruses and spyware. A significant number of malware attempt to contact a web site to download instructions or additional binaries, such as a number of the MyDoom variants(9). The assumption is that the malware will not be able to either contact the site directly, or use the proxy without prompting the user for the password, hence stopping or crippling the malware.

The proxy service shall be provided by running Squid(18) on a Linux server. It shall be configured only to allow connection attempts from machines within the GIAC network, and to require a valid username and password before permitting access. All requests shall be logged for analysis purposes.

2.5.5 NTP Server

Consistent time across all systems is critical in analyzing and managing any data distributed across multiple systems. If the time has drifted between servers, log files will be out of sync, and event correlation will be unreliable. In addition, many security protocols, such as validating SSL certificates and kerberos, depend upon synchronized clocks to run properly.

This is provided by a Linux server running NTP server software (10). Rather than syncing with external peers, however, it is equipped with a GPS receiver and a high resolution local oscillator. The GPS receiver provides a stratum 0 time source, and the local oscillator provides a low drift local source in case the GPS signal is lost. The software shall be configured to allow only non modifying queries from machines within the GIAC network.

2.6 DHCP Server

If an attacker is able to get a machine onto the local network, often times the machine will try to request an IP address via DHCP. This is much more of a concern if a wireless network is installed. To at not help the attacker out, the DHCP server should be configured only to hand out addresses to explicitly known hosts. More importantly, any requests from unknown hosts should be flagged as an anomalous event, and further investigated to determine whether it was a malicious machine that slipped onto the network or just somebody in sales forgetting to register their shiny new laptop.

2.7 DNS Views

To accommodate the fact that the external facing servers in the DMZ must be addressed by two different IP addresses based on whether the client is local or external, the DNS servers present two different views to clients. The external view is the list of A records, referenced by their public IP address, that are returned to clients from the internet, and only lists hosts that are in the DMZ. The internal view includes all hosts on the GIAC network by their private IP address.

2.7.1 External

Hostname	Description	IP
ns.giac.com	External DNS Server	192.0.2.129
www.giac.com	Reverse HTTP/HTTPS Proxy	192.0.2.130
smtp.giac.com	External SMTP Server	192.0.2.131
vpn.giac.com	VPN Server	192.0.2.132
client.giac.com	External Address for Masquerading	192.0.2.133

2.7.2 Internal

Hostname	Description	IP
ns-ext.giac.com	External DNS Server	10.12.1.2
www-ext.giac.com	External HTTP Server	10.12.1.3
smtp-ext.giac.com	External SMTP Server	10.12.1.4
vpn.giac.com	VPN Server	10.12.1.5
ns-int.giac.com	Internal DNS	10.12.2.2
cookie-db.giac.com	Cookie DB Server	10.12.2.3
cookie-app.giac.com	Cookie Application Server	10.12.2.5
smtp.giac.com	Internal SMTP Server	10.12.2.8
www-int.giac.com	Real HTTP/HTTPS Server	10.12.2.10
active-directory.giac.com	Windows AD Controller	10.12.2.11
dhcp.giac.com	DHCP Server	10.12.2.12
ntp.giac.com	NTP Server w/GPS Receiver	10.12.2.13
proxy.giac.com	Outgoing Web and FTP Proxy	10.12.2.14
snort.giac.com	Snort DB and Analysis Console	10.12.2.21

2.8 Routine Auditing

Once any network is set up according to the appropriate policies, it must be periodically checked to make sure it is still in compliance. For a firewall, the most critical element is making sure that no unauthorized traffic is passed. For example, according to the policy here, the only IP address that should be able to open connections to TCP port 25 on the internal SMTP server is that of the external SMTP server. If any other IP address is able to create this connection, that is a violation of the security policy, and must be both rectified and investigated to find out how the breach came about.The best way to do this is automated periodic portscans, both to and through the firewall.

To accomplish this, we leverage the existing snort grid. Each of the snort sensors is also running the nessus security scanner(19) listening on the interface in the `srv` subnet. On a scheduled basis, two scans are performed from each sensor. On the local subnet, including the local firewall interface, a vulnerability scan is performed, checking for unauthorized open ports, trojan backdoors, missing security patches, and other such problems. This verifies that the local machines are in a reasonably good state. On all other subnets, just a simple port scan is performed. This is not done to actually test the state of the remote machines themselves, but to verify that the firewall is filtering everything but explicitly authorized traffic.

All the results of these scans are sent back to the analysis console to be logged for auditing and alerting purposes.

3 Firewall Policy

The underlying firewall capabilities are provided by the Netfilter and iptables(8) facilities of the Linux 2.6 kernel. This provides such features as stateful firewalling, packet mangling, and NAT. To simply configuring iptables, Shorewall is used. Shorewall, aka the Shoreline Firewall, is a set of wrapper scripts around the iptables command. Using a collection of flat text files, interfaces and subnets are defined as high level objects. Rules are then defined allowing and denying various services between different hosts and zones, making management much simpler.

The Shoreline configuration files are flat text, one directive per line, with each directive composed of whitespace separated arguments. Each file configures a different section of the firewall policy.

3.1 Zones

Filename: `/etc/shorewall/zones`
Line Format:

```
Zone            Display           Comment
```

Field Name	Meaning
Zone	The label for this zone
Display	The human readable name for this zone
Comment	An administrative comment

This file defines the administrative labels for each “zone”. Each zone provides a label that gets referenced in all other configuration files, obviating the need to explicitly reference actual IP addresses and interfaces names in further rules.

In this case, the zones are quite simple. There is one zone for each of the internal subnets, plus a fifth zone for the internet.

```
net     Net             Internet
dmz     DMZ             Public Servers
srv     Servers         Internal Servers
users   Users           Unprivileged Users
admin   Administrators  System Administrators
```

In addition, there is the special zone, the `fw` zone, which matches any IP address that belongs to the firewall itself, and `all`, which matches all defined zones.

3.2 Interfaces

Filename: `/etc/shorewall/interfaces`
Line Format:

```
Zone Interface Broadcast   Options
```

Field Name	Meaning
Zone	The zone this interface is in
Interface	The device name for this interface
Broadcast	The broadcast address of this interface
Options	Any special flags relevant to this interface

Flag Name	Description
tcpflags	Check for invalid TCP flags
blacklist	Traffic on this interface is checked against a real-time blacklist
norfc1918	Incoming traffic to or from an rfc1918 address is filtered
nobogons	Traffic is checked against the reserved list
nosmurfs	Check for broadcast and multicast source packets
routefilter	Reject packets with our own source addresses

This file defines the firewall specific parameters for each network interface. The first column determines which zone each interface will be mapped into. The second column determines which interface is being configured. Note that while in this case, each zone has a dedicated physical interface, multiple lower traffic zones could easily be configured on a single interface with 802.1q VLAN tagging enabled.

The third column defines the broadcast address of each interface. While this can be set to auto-detect the actual broadcast address of each interface based on the actual settings, hard coding them allows the firewall rules to be brought up before any IP addresses are configured on any interfaces. This guarantees that there will be no race conditions where a service is briefly left unprotected by firewall rules.

The fourth and final column declares any special options. The dmz zone needs no special flags. The `users`, `admin`, and `srv` zones require the `dhcp` flag to allow DHCP clients on the `users` and `admin` zones to communicate with the DHCP server on the `srv` subnet. The `net` zone, corresponding to the internet, has the most restrictive filtering.

```
net   eth0      192.0.2.255 tcpflags,blacklist,norfc1918,nobogons,nosmurfs,routefilter
dmz   eth1      10.12.1.255
srv   eth2      10.12.2.255 dhcp
users eth3      10.12.3.255 dhcp
admin eth4      10.12.4.255 dhcp
```

3.3 Default Policy

Filename: `/etc/shorewall/policy`
Line Format:

```
Source Dest  Policy
```

Field Name	Description
Source	Source zone
Dest	Dest zone
Policy	Policy for traffic from Source to Dest

This file is used for configuring broad policies for inter-zone traffic. Since security is our prime concern, we begin with a default deny policy. Specific allow rules will be added later in the `rules` file.

```
all     all   REJECT
```

3.4 Rules

Filename: `/etc/shorewall/rules`
Line Format:

```
ACTION SOURCE          DEST            PROTO   DEST PORT(S)
```

Field Name	Description
Action	What action matching the rule will trigger
Source	What source addresses to match on
Dest	What destination addresses to match on
Proto	What IP protocols to match on
Dest Ports	What destination TCP/UDP ports to match on

Action can be any one of `ACCEPT`, which adds an allow rule, `DROP`, which adds a rule to silently drop traffic matched by the rule, `REJECT`, which will generate a protocol specific rejection to the request, or a predefined `Action` rule. `SOURCE` and `DEST` specify the zone, and optionally an IP address within that zone.

While `ping` is not a very safe protocol to allow across the internet, it is far too useful of a debugging tool to completely filter internally. It should also note that the `AllowPing` rule enables not `only` ping requests and replies, but also traceroute functionality. Since we are only enabling this rule internally, this should still be safe.

```
AllowPing !net          !net
```

Allow internet access to the external DNS server. Note that from the internet, only UDP DNS queries are allowed. This will prevent zone transfers from leaking out onto the internet, even if the server is misconfigured to tricked into allowing them.

```
ACCEPT  net             dmz:10.12.1.2   udp     domain
```

Allow internal users to contact internal the DNS server.

```
ACCEPT  users           srv:10.12.2.2   udp     domain
ACCEPT  admin           srv:10.12.2.2   udp     domain
ACCEPT  fw              srv:10.12.2.2   udp     domain
ACCEPT  dmz             srv:10.12.2.2   udp     domain
```

Allow the internet HTTP and HTTPS access to the Pound reverse proxy. The `users` zone is also allowed access so developers can see the same view as customers for testing and development purposes.

```
AllowWeb  net             dmz:10.12.1.3
AllowWeb  users           dmz:10.12.1.3
```

Allow only the Pound proxy to contact the back end HTTP server. All web requests, whether from internal or external, *must* first go through the Pound proxy first.

```
ACCEPT  dmz:10.12.1.3   srv:10.12.2.9   tcp     http
```

Allow internal clients to connect to the internal proxy to make outgoing HTTP and FTP requests. This means that all clients that use the protocols, including web browsers, ftp clients, automatic software update systems, etc must be configured to use this proxy.

```
ACCEPT   users          srv:10.12.2.14  tcp     8080
ACCEPT   dmz            srv:10.12.2.14  tcp     8080
ACCEPT   admin          srv:10.12.2.14  tcp     8080
ACCEPT   srv            srv:10.12.2.14  tcp     8080
```

And on the other end, allow the proxy to make the actual HTTP, HTTPS, and FTP connections to the external hosts. It is worth noting here that no special rules are required in the firewall configuration to make FTP work properly. Instead, shorewall by default loads the `ip_contrack_ftp` and `ip_nat_ftp` Linux kernel modules. These modules snoop on the control stream of FTP sessions, and modify the state table to allow the data streams through. All that must be explicitly allowed is the initial control connection. Any other future protocols that are determined to be safe to allow outbound must be added here.

```
AllowWeb srv:10.12.2.14 net
AllowFTP srv:10.12.2.14 net
```

Allow only the incoming external SMTP server to send and receive mail to and from the internet. Both directions must be explicitly defined because the server both sends and receives email to and from the internet.

```
AllowSMTP    net             dmz:10.12.1.4
AllowSMTP    dmz:10.12.1.4   net
```

Allow the internal SMTP server to relay emails to the external one. Note that there is no explicit rule defined here to allow the external server to deliver mail to the internal one, as that is covered in the next set of rules.

```
AllowSMTP    srv:10.12.2.8   dmz:10.12.1.4
```

Allow internal users to send and read email. Note that user machines are not allowed to submit email except by the `submission` protocol, which requires the user authenticate first.

```
ACCEPT  users           srv:10.12.2.8   tcp     submission
ACCEPT  users           srv:10.12.2.8   tcp     pop3s
ACCEPT  users           srv:10.12.2.8   tcp     imaps
```

Allow administrative systems and servers to send mail only. Note that POP3 and IMAP are not allowed from any of these systems. The SMTP hole is intended for any email alerts that are generated. All reading of mail should be done by a system in the `users` zone. Email based viruses are far too common to safely allow email to be read on any truly important systems.

```
AllowSMTP    admin          srv:10.12.2.8
AllowSMTP    dmz            srv:10.12.2.8
AllowSMTP    fw             srv:10.12.2.8
```

Allow incoming access to the OpenVPN server. Once connected to the VPN server, clients are routed from either the `admin` or `users` subnets based upon the user credentials provided, so not further filtering rules are required for managing end station traffic.

```
ACCEPT  net             dmz:10.12.1.5 udp     1194
```

Allow `admin` zone systems to ssh to all other local systems for management purposes.

```
AllowSSH     admin      !net
```

Allow all local systems to synchronize to the central NTP server.

```
AllowNTP     !net       srv:10.12.2.13
```

Allow the wide range of ports necessary for the various Windows networking and Active Directory protocols to function, including DNS and LDAP.

```
AllowDNS     users           srv:10.12.1.11
AllowSMB     users           srv:10.12.1.11
AllowSMB     srv:10.12.1.11  users
Allow        users           srv:10.12.1.11  tcp    389
AllowDNS     admin           srv:10.12.1.11
AllowSMB     admin           srv:10.12.1.11
AllowSMB     srv:10.12.1.11  admin
Allow        admin           srv:10.12.1.11  tcp    389
```

3.5 Bidirectional NAT

Filename: `/etc/shorewall/nat`
Line Format:

```
EXTERNAL       INTERFACE         INTERNAL      ALL INTERFACES     LOCAL
```

Field Name	Meaning
External	External IP address
Interface	External interface to use
Internal	Internal range
All Interfaces	Whether to restrict translation
Local	NAT the firewall or only clients

Since all systems are configured with RFC1918 class addresses, any system that needs to talk directly to the internet must be subject to NAT. Any such machines are required to be in the `dmz` subnet.

The first field is the external IP address that internal systems should be mapped to. The second field is the interface the translation is performed on. By specifying a physical interface plus a colon and index number, shorewall will create a virtual interface with the specified external IP address. The third field is the internal address that should be translated. The fourth field determines whether NAT will be performed from all interfaces, or only the specified external interface. The fifth and final field determines whether the NAT will be performed on traffic originating from the firewall itself. Since neither the firewall nor any other local systems should need to address the external IP addresses of any of these systems, they should both be configured to “no” for all NAT rules.

NAT the external DNS server.

```
192.0.2.129  eth0:1     10.12.1.2   no              no
```

NAT the external HTTP/HTTPS proxy.

```
192.0.2.130  eth0:2     10.12.1.3   no              no
```

NAT the external SMTP server.

```
192.0.2.131  eth0:3     10.12.1.4   no              no
```

NAT the VPN server.

```
192.0.2.132  eth0:4     10.12.1.5   no              no
```

3.6 Client Masquerading

Filename: `/etc/shorewall/masq`
Line Format:

```
INTERFACE SUBNET          ADDRESS
```

Field Name	Meaning
Interface	Interface to NAT on
Subnet	Subnet to NAT
Address	Different address to NAT behind

Only systems that must accept incoming connections from the internet are configured with dedicated external IP addresses and static mapping rules. All other systems that are only expected to make outgoing connections are masqueraded behind a single IP address.

In this case, the IP address to masquerade clients behind is not specified in the third column. Instead, the interface field is specified as both the physical interface, and an IP address to use. This will trigger the creation of the IP address as an alias on that interface, and use that IP for client masquerading.

```
eth0:192.0.2.133   10.12.2.0/24
eth0:192.0.2.133   10.12.3.0/24
eth0:192.0.2.133   10.12.4.0/24
```

3.7 Firewall Disable Routes

Filename: `/etc/shorewall/routestopped`
Line Format:

```
INTERFACE          HOST(S)
```

Field Name	Meaning
Interface	Interface to leave open
Hosts	Which hosts should be allowed in

When shorewall is in the stopped state, all rules are flushed and a global default deny policy is put in place. Since such a state makes managing the firewall system rather awkward, this file allows an interface and subnet that is still allowed to communicate with the firewall. In this case, we only allow the `admin` subnet in, including the one system with dial in access.

```
eth4        10.12.4.0/255
```

Glossary

Access Point (AP)	A dedicated piece of hardware which bridges multiple wireless clients onto a wired LAN.
Advanced Encryption Standard (AES)	Current reccomended encryption standard by NIST. Also known as FIPS-197.
Ciphertext	The encrypted result of combining a key with plaintext through a cryptographic algorithm.
Denial of Service (DoS)	A form of attack in which the victim is activly denied access to a service such as network connectivity.
Initialization Vector (IV)	A random value used in conjunction with the encryption key to prevent multiple identical packets from encrypting to identical ciphertext.
Message Integrity Checksum (MIC)	A cryptographically secure fingerprint of a packet that allows an endpoint to determine if the packet has been tampered with by a third party.
Plaintext	Unencrypted data.
RC4	A fast simple and reasonably secure encryption cipher. Used in a flawed manner in WEP.
Robust Security Network (RSN)	The latest and most secure encrypted wireless encapsulation. Uses AES as the underlying cipher. Also known as 802.11i or WPA2.
Temporal Key Integrity Protocol (TKIP)	Security protocols used in WPA.
WiFi Protected Access (WPA)	A new encrypted wireless encapsulation method that still runs on older RC4 hardware.
Wired Equivalent Privacy (WEP)	The original RC4 based encryption standard for wireless LANs.

Works Cited

[1] Paul Albitz and Cricket Liu. *DNS and BIND*. O'Reilley, fourth edition edition, 2001.

[2] Clam AV Development Team. Clam antivirus [online]. 2004. Available from World Wide Web: http://www.clamav.net.

[3] J. Philip Craiger. 802.11, 802.1x, and wireless security. 2002. Available from World Wide Web: http://www.giac.org/practical/GSEC/Phillip_Craiger_GSEC.pdf.

[4] Phil Deneault. Placid [online]. 2004. Available from World Wide Web: http://speakeasy.wpi.edu/placid/.

[5] Thomas M Eastep. Shoreline firewall [online]. 2004. Available from World Wide Web: http://shorewall.sourceforge.net/.

[6] Jon Edney and William A. Arbaugh. *Real 802.11 Security*. Addison Wesley, 2004.

[7] National Institute of Standards and Technology. Federal information processing standards publication 197. 2001. Available from World Wide Web: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

[8] Netfilter Development Team. Netfilter/iptables [online]. 2004. Available from World Wide Web: http://www.netfilter.org/.

[9] Networks Associates Technology, Inc. Virus profile: W32/mydoom.ab@mm [online]. 2004. Available from World Wide Web: http://us.mcafee.com/virusInfo/default.asp?id=description&virus_k=128357.

[10] NTP Development Team. Ntp [online]. 2004. Available from World Wide Web: http://www.ntp.org/.

[11] QoSient, LLC. Argus [online]. 2004. Available from World Wide Web: http://www.qosient.com/argus/.

[12] Marty Roesch and Brian Caswell. Snort [online]. 2004. Available from World Wide Web: http://www.snort.org/.

[13] Bruce Schneier. *Applied Cryptography*. John Wiley & Sons, second edition edition, 1996.

[14] Itsik Mantin Scott Fluhrer and Adi Shamir. Weaknesses in the key scheduling algorithm of rc4. 2001. Available from World Wide Web: http://www.drizzle.com/~aboba/IEEE/rc4_ksaproc.pdf.

[15] Robert Segall. Pound reverse proxy [online]. 2004. Available from World Wide Web: http://www.apsis.ch/pound/index_html.

[16] Sendmail Consortium. Sendmail [online]. 2004. Available from World Wide Web: http://www.sendmail.org.

[17] Spamassassin Development Team. Spamassassin [online]. 2004. Available from World Wide Web: http://spamassassin.apache.org/.

[18] Squid Development Team. Squid proxy server [online]. 2004. Available from World Wide Web: http://www.squid-cache.org/.

[19] Tenable Network Security. Nessus open source vulnerability scanner project [online]. 2004. Available from World Wide Web: http://www.nessus.org/.

[20] The Shmoo Group. Airsnort homepage [online]. 2004. Available from World Wide Web: http://airsnort.shmoo.com.

[21] TippingPoint Technologies. Tipping point intrusion prevention systems [online]. 2004. Available from World Wide Web: http://www.tippingpoint.com/.

[22] Jesse R. Walker. Unsafe at any key size; an analysis of the wep encapsulation. 200. Available from World Wide Web: http://www.dis.org/wl/pdf/unsafe.pdf.

[23] Wireless Garden, Inc. Cantenna wifi booster antenna [online]. 2003. Available from World Wide Web: http://www.wirelessgarden.com/catalogue/SCB10.html.

[24] James Yonan. Openvpn [online]. 2004. Available from World Wide Web: http://openvpn.sourceforge.net/.