IBM

Linux on IBM @server zSeries and S/390: Best Security Practices

Security topics for z/VM

Securing Linux guests

Firewalls on Linux for zSeries

Gregory Geiselhart
Ami Ehlenberger
Darius Fariborz
Jerry Lam
Neville Mendes
Carlos Ordonez
Luiz Carlos Santos
Karl-Erik Stenfors

Redbooks

ibm.com/redbooks

International Technical Support Organization

Linux on IBM @server zSeries and S/390: Best Security Practices

May 2004

SG24-7023-00

Note: Before using this information and the product it supports, read the information in "Notices" on page vii.

First Edition (May 2004)

This edition applies to z/VM Version 4 Release 4 and multiple Linux distributions. SUSE LINUX Enterprise Server 8 is used for the examples in this book.

Contents

Notices

Trademarks

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

AIX®
DB2®
developerWorks®
DirMaint™
e(logo)server®
@server®
@server®
FICON®
HiperSockets™
IBM®
ibm.com®
MVS™
OS/390®
PR/SM™
RACF®
Redbooks™
Redbooks (logo) ™
S/390®
SecureWay®
Tivoli®
VM/ESA®
WebSphere®
z/OS®
z/VM®
zSeries®

The following terms are trademarks of other companies:

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Preface

This IBM® Redbook discusses best security practices for running Linux as a z/VM® guest on IBM @server® zSeries® and S/390® machines. This publication is intended for system administrators and IT architects responsible for deploying secure Linux servers running under z/VM. We consider both z/VM and Linux security topics.

We examine the unique security and integrity features zSeries offers for consolidating a large number Linux servers under z/VM. We discuss virtual machine isolation and command privileges assigned to VM guests. Security configuration options for z/VM Version 4.4 are explained.

In this book, we also discuss Linux security topics. We examine options for hardening a Linux installation. Securing Linux network traffic using Secure Sockets Layer and Secure Shell is considered. We look at implementing a virtual private network using FreeS/WAN. Commercial firewall technology and implementation using the StoneGate firewall for zSeries is discussed. We examine using IBM Tivoli® Access Manager in conjunction with an LDAP server running on z/OS® to authenticate Linux users against a RACF® running on z/OS.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, Poughkeepsie Center.

Gregory Geiselhart is a Project Leader for Linux on zSeries at the International Technical Support Organization, Poughkeepsie Center.

Ami Ehlenberger is a Staff Software Engineer and has been with IBM for the past four years. Ami was hired into the OS/390® development team in 1999 and has since worked in the areas of integration test, solution design, and services. She has a B.S. in Computer Science from Indiana University of Pennsylvania and an MBA in e-business from the University of Phoenix. Her expertise resides in LDAP, IBM WebSphere®, and IBM Tivoli Access Manager. Ami currently works for the Custom Technology Center, an IBM @server services organization.

Darius Fariborz has been a Network Architect in IBM Global Services since 1995. He joined IBM U.K. Laboratories in 1985 and holds an MSc in Telecommunications and Micro-Electronics from the University of Surrey. Prior to joining IBM, Darius worked for International Aeroradio Limited as a software engineer from 1982 to 1984 and as hardware design engineer for Thorn EMI Datatech from 1979 to 1982. His main interests are in network connectivity and security.

Jerry Lam is a Data Security Specialist in Prudential Securities, a subsidiary of Wachovia in the United States. He has 12 years of experience in Wall Street financial institutions, concentrating mainly on information security. He holds a B.S. in Computer Science from New York Institute of Technology. His areas of expertise include C programming, system auditing, network vulnerability testing, and security administration. This is his first redbook. He is a strong advocate for open-source tools for security on all system platforms.

Neville Mendes is a Senior Network Security Engineer based at the EMEA headquarters of Stonesoft in Helsinki, Finland. He covers StoneGate for zSeries classes worldwide in the areas of StoneGate firewall VPN and security. He also travels extensively implementing customer environments.

Carlos Ordonez is a Senior Systems Engineer at the Poughkeepsie IBM Laboratory in the United States. Carlos joined IBM in 1997. He has more than 25 years of experience in mainframe technologies and holds a B.S. degree in Computer Science from Florida International University. His areas of expertise include programming, systems programming, and Linux on S390/zSeries. He is the Linux Technical Lead for the zSeries New Technology Center. He has participated in previous ITSO residencies.

Luiz Carlos Santos is a support analyst for EDS in Brazil. He has been involved professionally in system administration and support since 1994. He has five years of experience in UNIX® on several platforms. For more than five years, he has been responsible for providing customer support in UNIX installation and customization. He holds a degree in Computing Science from University Paulista. His areas of expertise include AIX®, Sun Solaris, HP True64, and Linux for zSeries. This is his first redbook.

Karl-Erik Stenfors is a Senior IT Specialist in the Product and Solutions Support Centre (PSSC) in Montpellier, France. He has more than 30 years of experience in the large systems field, both as a systems programmer and consultant with IBM customers, and for the last 17 years as an IBM employee. His areas of expertise include zSeries hardware and operating systems. He teaches at numerous IBM user group and IBM internal conferences. Before joining the PSSC in 2000, he was an employee of IBM, Norway. While an employee in Norway, he held two international assignments, both in the large systems field. He has participated in earlier ITSO residencies.

Thanks to the following people for their contributions to this project:

Terry Barthel, Dave Bennin, Roy Costa, Al Schwab
International Technical Support Organization, Poughkeepsie Center

Frank J. De Gilio
IBM Design Centers for e-business infrastructure

Jack Hoarau, Gerard Laumay
IBM PSSC, Montpellier, France

Erich Amrehn, Ulrich Boche, Manfred Gnirss
IBM Germany

Cliff Laking
IBM United Kingdom

Alan Altmark
IBM z/VM Development

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

`ibm.com/redbooks/residencies.html`

Comments welcome

Your comments are important to us!

We want our Redbooks™ to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- Use the online **Contact us** review redbook form found at:

  ibm.com/redbooks

- Send your comments in an Internet note to:

  redbook@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYJ Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

1

Introduction

In this chapter, we provide an introduction to security concepts. We discuss:

- Security goals
- Elements of security
- System installation and backup

1.1 Security goals

Strictly speaking, no computer system can be considered absolutely secure. The primary goal for a security administrator is to establish policies and procedures that minimize risk to a computer system to an acceptable level. To determine the acceptable risk level, it is important to understand:

- What needs to be protected
  Higher security measures (such as encryption and strong authentication) are required to protect highly sensitive data.
- What components are vulnerable
  Systems connected to public networks (such as the Internet) are inherently more at risk than systems connected to a secured intranet.
- What steps can be taken to reduce the risk
  A risk assessment considers assets requiring protection and the vulnerabilities to those assets. From the risk assessment, appropriate policies can be developed to minimize security risk.

1.1.1 Security policy

The first step to secure a system is to establish a security policy. The security policy defines access methods to system objects. It details the privileges assigned to users and processes. In addition, the policy should define the mechanism for users to acquire access to higher privileges (for example, in emergencies or for control access). Components of a security policy can include:

- Acceptable use
- User accounts and authentication
- Access control
- Network control
- Encryption
- Logging and accounting
- Intrusion detection

In general, the cost of protecting an assets should not be higher than the cost of recovering the assets (should the threat become a reality). A security policy should change with technology and cover all components of security. Figure 1-1 on page 3 illustrates the basic principle of security.

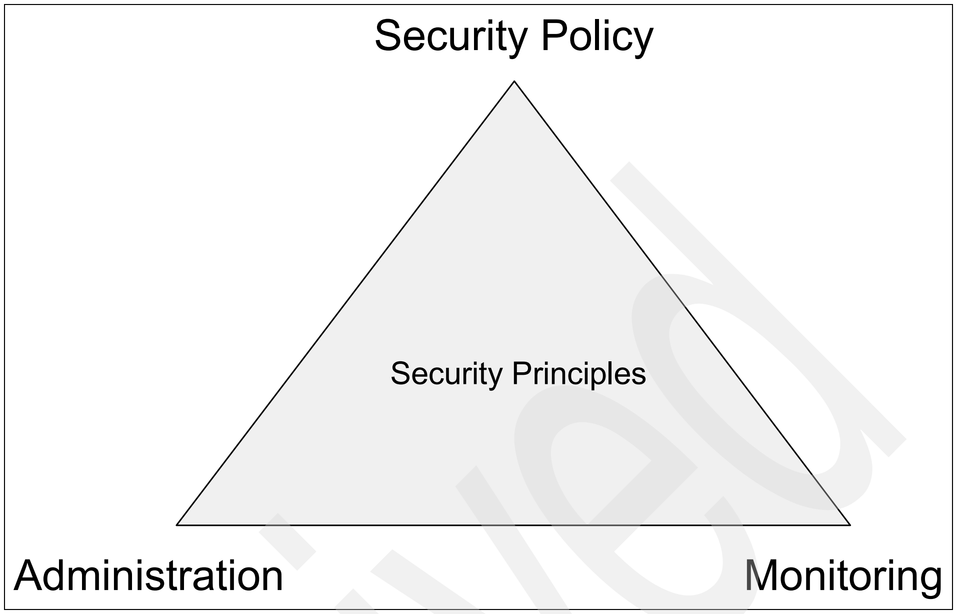


Figure 1-1   Security principles

The security policy relies on security administration and monitoring. Over time, a security policy can change as new administration tools and principles are put in place, or as security monitoring reveals new or unforeseen vulnerabilities.

1.2 Elements of security

In this section, we examine some of the security aspects to consider when developing a security policy.

1.2.1 Physical security

The first and most important element to consider is the security and integrity of physical locations from which the system can be accessed. Any security procedure or process can be subverted if an attacker gains access to the system console. At a minimum, access to the physical system should be restricted to authorized personnel (using badge access, for example). Entry into the restricted areas should be monitored and tracked.

1.2.2 System security

Procedures should be in place limit and monitor user access to the system. Some suggested procedures include:

- Authenticate users from a central repository (such as LDAP or RACF) to lower maintenance overhead.
- Maintain user password standards and expiration policies.
- Limit the number of users authorized to access the system.
- Limit the number of running services.
- Install and upgrade RPM packages from trusted sources.
- Logging to a central log server and log all system accesses.
- Apply recommended security patches to Linux hosts.
- Restrict the number of hosts running X Windows.
- Do not allow programming languages and compilers on production machines.
- Include the acceptable use policy as part of the system logon screen.

1.2.3 Network security

Securing network access is critical for hosts connected to a public network. However, it is equally important to secure networks on internal, corporate networks. Typically, the majority of network security incidents originate from within an internal network. Some points to consider include:

- For untrusted hosts, use a firewall. Access to these hosts should be limited. Examples of untrusted hosts include:
  - Hosts on the demilitarized zone (DMZ)
  - Hosts on external networks
  - Hosts for vendor systems
- The number of trusted hosts should be limited and tightly monitored. Examples of trusted hosts include:
  - Hosts within an internal or corporate network
  - Hosts accessed over a virtual private network
- Remote connections should be encrypted with a industry-approved encryption algorithm equal to or exceeding 1024 bits.
- Use server certificates for secure HTTP traffic.
- Access to security servers (such as a firewall or log server) should be strictly limited to authorized security personnel.

1.3 System installation and backup

It is important to ensure that no malicious or insecure code is installed on the system. When initially installing Linux, always use images from a supported distributor (preferably from a secure medium such as the distributor-provided CD-ROM). Take regular backups to ensure that recovery to a secure install is possible in the event the system is compromised.

1.3.1 Verifying the RPM package

RPM packages should always be authenticated before installation. This ensures that the package contents have not been tampered with. Use the **`rpm -Kvv`** ***`package`*** command to check the author's digital signature. Example 1-1 illustrates how to authenticate a package signature.

Example 1-1 Verifying an RPM package

```
# rpm -Kvv tripwire-1.2-385.s390.rpm
D: New Header signature
D: Signature size: 156
D: Signature pad : 4
D: sigsize        : 160
D: Header + Archive: 204491
D: expected size   : 204491
tripwire-1.2-385.s390.rpm:
MD5 sum OK: 63b2c0ea8302f6db86b02ec025b913e0
gpg: Warning: unsafe permissions on directory "/usr/lib/rpm/gnupg"
gpg: Warning: unsafe permissions on file "/usr/lib/rpm/gnupg/pubring.gpg"
gpg: Signature made Tue Nov  5 19:09:00 2002 PST using DSA key ID 9C800ACA   1
gpg: Good signature from "SuSE Package Signing Key <build@suse.de>"          2
gpg: Warning: unsafe permissions on file "/usr/lib/rpm/gnupg/trustdb.gpg"
gpg: WARNING: This key is not certified with a trusted signature!            3
gpg:          There is no indication that the signature belongs to the owner.
Fingerprint: 79C1 79B2 E1C8 20C1 890F  9994 A84E DAE8 9C80 0ACA
```

In this example, we note:

1. The RPM package contains a Pretty Good Privacy (PGP) digital signature.
2. The signature belongs to SUSE, and it appears to be valid (that is, it has not been tampered with).
3. Although the signature appears valid, it has not been registered with a central authority. In this case, the RPM package came from an official SUSE CD, and therefore can be trusted.

Note: If a signed RPM package is modified, the check listed in the second point will fail. It is possible for an RPM package to be signed with a forged certificate. This can be detected if the original signer registers their certificate with a trusted third party (in check three above).

In Example 1-2, we show an example of verifying an unsigned RPM package.

Example 1-2 Example of an unsigned RPM package

```
# rpm -Kvv tcpshow-1.0-2.src.rpm
D: New Header signature
D: Signature size: 68
D: Signature pad : 4
D: sigsize        : 72
D: Header + Archive: 16064
D: expected size   : 16064
tcpshow-1.0-2.src.rpm:
MD5 sum OK: 7c5753f34a8c6b50e1ade11ade311c1a
```

In this case, no signature information is displayed. Packages from unknown or questionable sources should not be installed. Use only packages from an authorized Linux distributor.

2

z/VM integrity and security

In this chapter, we examine security for Linux guests executing in a z/VM environment. Topics include:

- zSeries and z/VM system integrity
- zSeries network security
- Securing your z/VM system
- CP privilege classes
- The z/VM SYSTEM CONFIG file
- The z/VM user directory
- Directory Maintenance Facility
- RACF for z/VM

2.1 zSeries and z/VM system integrity

We begin by first examining system integrity features offered by running Linux guests under z/VM. If we consider Linux executing in an LPAR (that is, native mode without z/VM), it is very much a pure Linux environment. However, Linux benefits from the inherent values of the z-Architecture itself:

- The isolation between LPARs ensures as secure an environments as executing on physically separate servers, certified by Evaluation Assurance Levels (EAL) 5 Common Criteria.
- Linux can access zSeries-unique hardware features such as:
  - HiperSockets™
  - zSeries I/O devices such as OSA cards and FICON® channels
  - zSeries specialized cryptographic hardware
- Extremely high availability of the hardware platform.
  Mean-Time-Between-Failure (MTBF) is estimated to be more than 30 years, unique in the industry.

This section also addresses the unique security and integrity benefits z/VM adds to Linux. In this context, system integrity refers to the ability to run multiple systems on a single physical zSeries server and to ensure that one system cannot compromise the operation of any other system.

Two levels of system integrity are enforced when running on zSeries:

- Integrity at logical partition level
  Each logical partition (LPAR) is isolated from all other LPARs.
- Integrity provided by z/VM
  Each virtual machine (guest) running under z/VM is isolated from all other z/VM guests.

System integrity provided by both z/VM and zSeries ensures Linux guests are isolated from all other operating systems and guests running on the same physical zSeries server.

2.1.1 LPAR integrity

The ability to partition the zSeries Central Processing Complex (CPC) into multiple LPARs is provided by the Processor Resource/System Manager (PR/SM™). PR/SM is a hardware facility that enables sharing resources across LPARs. It is the Start Interpretive Execution (SIE) instruction that enables context switches between LPARs. When a context switch occurs between LPARs, the current state of the LPAR is saved. All resources are cleared or reset before control is passed to another LPAR. When control is returned to an LPAR, the saved state of the LPAR is restored before execution of the SIE instruction.

System memory is always dedicated to a specific LPAR. PR/SM performs dynamic address translation when a context switch occurs between LPARs. Because address is a hardware feature, this ensures that software running in one LPAR cannot access or compromise an address space running in a different LPAR. It is possible to reconfigure LPAR memory allocation. However, memory is cleared before it is reassigned to another LPAR.

Note: Details about the interpretive execution architecture can be found in "ESA/390 interpretive-execution architecture, foundation for VM/ESA," by D. L. Osisek et al., available at:

`http://www.research.ibm.com/journal/sj/301/ibmsj3001E.pdf`

LPAR security certification

PR/SM for the IBM @server zSeries 900 has achieved Common Criteria for Information Security Evaluation EAL5 level certification. This provides a high degree of assurance that operating systems running in different LPARs on the same z900 are isolated from each other. SUSE LINUX Enterprise Server 8 with Service Pack 3 has achieved Controlled Access Protection Profile compliance under the Common Criteria (referred to as CAPP/EAL3+). For more information, see:

`http://www.ibm.com/servers/eserver/zseries/security/certification.html`

2.1.2 Integrity provided by the z/VM Control Program

The z/VM Control Program (CP), the hypervisor, is primarily a real-machine resource manager. The CP provides each user with an individual working environment known as a virtual machine. Each virtual machine is a functional equivalent of a real system, sharing the real processor function, storage, console, and input/output (I/O) device resources. When you first log on to z/VM, the CP controls the working environment. Many of the facilities of z/VM are immediately available to you. For example, you can use the CP commands to do various system management tasks. However, most of the work done on z/VM requires the Conversational Monitor System (CMS) or a guest operating system, such as z/OS, to help with data processing tasks and to manage work flow. The CP provides connectivity support that allows application programs to exchange information with each other and to access resources residing on the same z/VM system or on different z/VM systems.

Note: For details about running virtual machines under z/VM, consult *z/VM V4R4.0 Virtual Machine Operation*, SC24-6036, and *z/VM V4R4.0 Running Guest Operating Systems*, SC24-5997.

In the same manner as the SIE instruction enables context switches between LPARs, the CP uses the SIE instruction when dispatching virtual machines. When dispatched, a virtual machine runs until its time slice expires, or until it executes an instruction that cannot be virtualized (for example, an I/O instruction). At that time, control returns to the CP. The CP then executes instructions on behalf of the virtual machine, or dispatches another virtual machine. This mechanism enables the CP to limit many software or hardware failures encountered when running on the real processor. If the error is isolated to a virtual machine, only that virtual machine fails without affecting either the CP itself or any other virtual machine running under z/VM.

z/VM security certification

In a press release jointly issued by SuSE and IBM on August 5, 2003 titled "IBM and SuSE Earn First Security Certification of Linux," IBM makes the following statement:

> In addition to IBM's ongoing commitment to accelerate the development and certification of Linux, the company will continue to invest in ongoing certifications for new and existing IBM products. Common Criteria certification is anticipated for z/VM in the upcoming year, along with IBM's suite of middleware products on Linux. IBM Directory has just completed evaluation under the Common Criteria, WebSphere Application Server and Tivoli Access Manager are currently undergoing evaluation, and several other Software Group products are being prepared to enter the process.

2.2 zSeries network security

zSeries networking facilities offer a unique security advantage. Queued direct input/output (QDIO) architecture uses direct memory access (DMA) to transfer network data from the address space of the sender to the address space of the receiver. For HiperSocket networks, this means that network traffic cannot be "sniffed" by third parties (other z/VM guests or other LPARs). For OSA-Express, network traffic is moved directly to/from the adapter to the address space of the send/receiver. For details about zSeries networking, see *zSeries HiperSockets*, SG24-6816, available at:

http://www.ibm.com/redbooks/abstracts/sg246816.html

Note: Although a Linux guest cannot sniff QDIO traffic that is not directed to or originating from that guest, it is possible for one process running in the Linux guest to view traffic directed to another process running in the same guest. Similarly, network traffic directed to an OSA-Express adapter can be sniffed by connected hosts external to the zSeries server.

2.3 Securing your z/VM system

Before securing your Linux environment, it is important to first secure the z/VM system itself. An insecure z/VM system cannot provide security to Linux guests. For example, preventing unauthorized access inside a Linux virtual machine cannot prevent an intruder from gaining access to the z/VM console (and thereby compromising the entire system integrity).

In addition to z/VM itself, two other IBM software products can be installed and used to enhance integrity and security:

- IBM Directory Maintenance for z/VM and VM/ESA® (DirMaint™)
  DirMaint provides a safe and simple interface to define and manage the z/VM directory. Features such as minidisk allocation, virtual machine management, and auditing of transactions help system administrators create and maintain a secure environment on which to deploy Linux virtual guest machines.
- The Resource Access Control Facility (RACF)
  RACF is an External Security Manager (ESM) to further enhance z/VM system security. RACF provides the ability to system resources, as well as virtual machine resources. RACF also provides a comprehensive set of auditing facilities. RACF can be used to verify logon passwords and control access to minidisks, network nodes, shared segments, spool files, and system commands.

An overview of z/VM security and integrity features can be found in *z/VM Security and Integrity*, by Alan Altmark and Cliff Laking,.

Note: For details about how to secure the z/VM TCP/IP stack, consult *z/VM V4R4.0 TCP/IP Planning and Customization*, SC24-6019.

2.3.1 System integrity statement for z/VM

System integrity is an important characteristic of z/VM. This statement extends the previous statements of IBM on system integrity in the z/VM environment. IBM has implemented specific design and coding guidelines for maintaining system integrity. Procedures have also been established to make the application of these design and coding guidelines a formal part of the design and development process.

Because it is not possible to certify that any system has perfect integrity, IBM continues its efforts to enhance the integrity of z/VM and to respond promptly when exposures are identified. IBM will accept APARs that describe:

- z/VM system integrity exposure
- Exposures to system integrity caused by a program running in a virtual machine, provided the problem has not been introduced by a customer-controlled authorization mechanism

z/VM system integrity definition

z/VM system integrity ensures that an authorized virtual machine cannot:

- Circumvent or disable Control Program real or auxiliary storage protection.
- Access resources protected by RACF. Protected resource can include virtual machines, minidisks, and terminals.
- Access a Control Program password-protected resource.
- Obtain control in real supervisor state, or with privilege class authority greater than those it is assigned.
- Circumvent system integrity of any other z/VM guest.

With real storage protection, a virtual machine is isolated from all other virtual machines. The CP accomplishes this using:

- Dynamic Address Translation (DAT) implemented at the hardware level
- The Start Interpretive Execution instruction
- The Set Address Limit facility

With auxiliary storage protection, the CP ensures that a virtual machine cannot access a minidisk or virtual disk (VDISK) to which it is not authorized. This protection is implemented using channel program translation.

Customer responsibility

Although protection of the customer's data remains the customer's responsibility, data security continues to be an area of vital importance to IBM. The IBM commitment to z/VM system integrity (as described in this statement) represents a significant step to help customers protect their data.

Note: For more information about z/VM security and integrity, refer to *z/VM V4R4.0 General Information*, GC24-5991.

2.4 CP privilege classes

CP commands are grouped according to the level of authorization required to execute the specific command. The privilege class assigned to z/VM users determines what commands any user is authorized to execute. Table 2-1 lists the CP command privilege classes, the type of user intended to be authorized for the privilege class, and a description of the functions provided by the privilege class.

Table 2-1 CP command privilege classes

Class	User	Function
A	System operator	Controls the z/VM system. System operator is responsible for system availability and resources.
B	System resource operator	Controls all real resources except those controlled by system operator or spooling operator.
C	System programmer	Updates or changes system-wide parameters.
D	Spooling operator	Controls spool files and system real reader, punch, and printer.
E	System analyst	Examines and saves system operation data.
F	Service representative	Obtains and examines detailed system data from I/O devices. Class F is reserved for IBM use.
G	General user	Controls functions associated to the user's own virtual machine.
Any		Commands available to any user regardless of privilege class.
H		Reserved for IBM use.
I-Z 1-6	User defined	Classes available to be customized by each installation.

Users are authorized for specific privilege classes based on their user directory entry, as discussed in 2.6, "The z/VM user directory" on page 20. If a user attempts to execute an unauthorized command, the CP will ignore the command or return with an error message. User-defined privilege classes can be created using the procedure outlined in 2.5.6, "Redefining a command privilege class" on page 19.

Tip: To see the commands for which a user is authorized, execute the CP QUERY COMMAND command when logged on as that user.

Privilege classes help maintain z/VM system integrity. Commands issued by general (class G) users cannot affect the operation of the CP or of any virtual other machine other that of the executor. Be aware that privilege classes A, B, and C have sufficient authority to bypass security controls.

Important: Never assign a higher privilege class to a user than required. For Linux virtual machines, assign a privilege class no higher than class G.

2.5 The z/VM SYSTEM CONFIG file

The system configuration file (SYSTEM CONFIG) stores configuration information of the system itself. This file contains system and I/O definitions parameters. In traditional z/VM systems, the HCPRIO ASSEMBLE and HCPSYS ASSEMBLE files are used for the same purpose. In this section, we identify and discuss selected SYSTEM CONFIG statements. Details about specific statements can be found in *z/VM V4R4.0 CP Planning and Administration*, SC24-6043.

Note: In order for changes made to the SYSTEM CONFIG to take effect, z/VM must be re-IPLed.

2.5.1 Enabling journaling

The CP can be enabled to journal invalid passwords supplied to the CP LOGON or LINK commands. Use the JOURNALING statement to record invalid password events. See Example 2-1.

Example 2-1 JOURNALING statement in the SYSTEM CONFIG file

```
Journaling,
  Facility On ,                             1
  Set_and_Query On ,                        2
  Logon ,
    Message after 3 attempts to operator ,  3
    Account after 2 attempts ,              4
  Link ,
    Message after 3 attempts to operator ,  5
    Disable after 4 attempts                6
```

Example 2-1 illustrates the use of the JOURNALING statement:

1. The journal facility is enabled.
2. The CP SET JOURNAL and QUERY JOURNAL commands are enabled.

Note: The QUERY JOURNAL command reports the journaling status. The SET JOURNAL command enables or disables journaling.

3. After three logon attempt failures, a message is sent to OPERATOR.
4. After two logon attempt failures, an accounting record is generated.
5. After three link attempt failures, a message is sent to OPERATOR.
6. After four link attempt failures, the CP LINK command is disabled for the user.

2.5.2 System features

The FEATURES statement sets certain system attributes at initialization. Options of interest from a security perspective include:

- PASSWORDS_ON_COMMANDS
  The PASSWORD_ON_COMMANDS feature controls whether the CP prompts users for passwords when using the CP AUTOLOG, LINK, or LOGON command. If specified as **NO**, users must supply passwords when prompted by the CP. If specified as **YES**, users can specify passwords as parameters to the respective commands (passed as cleartext).
- CLEAR_TDISK
  The CLEAR_TDISK feature instructs the CP to clear all temporary disk space when detached by the user.
- SET_PRIVCLASS
  This feature enables users the authority to use the CP SET PRIVCLASS command. The privilege class can also be set in the SYSTEM CONFIG file, as discussed in 2.5.3, “Defining privilege classes” on page 16.

Example 2-2 FEATURES statement in the SYSTEM CONFIG file

```
Features ,
  Disable ,
    Set_Privclass ,    1
  Enable ,
    Clear_TDisk   ,    2
  Passwords_on_Cmds ,
    Autolog  no ,      3
    Link     no ,
    Logon    no
```

Example 2-2 illustrates the use of the FEATURES statement:

1. The CP SET_PRIVCLASS command is disabled.
2. The CP is to clear temporary disks when detached.

3. Users are to be prompted for passwords when using the CP AUTOLOG, LINK, or LOGON command.

2.5.3 Defining privilege classes

The PRIV_CLASSES statement is used to change privilege classes authorizing:

- Logging on as the system operator
- Access to the IOCP utility
- Recording intensive error logging
- The default user class

Example 2-3 PRIV_CLASSES statement in the SYSTEM CONFIG file

```
Priv_Classes  ,
   Operator     AC , 1
   User_Default G    2
```

Example 2-3 illustrates the use of the PRIV_CLASSES statement:

1. The primary system operator is authorized for privilege classes A and C.
2. The default class for general users is G.

2.5.4 z/VM virtual networking

Virtual networks provide connectivity between z/VM guests on the same z/VM host system without using dedicated hardware. Virtual networks can be created from virtual CTCA devices; however, these only offer point-to-point connectivity. Every network link must be defined and configured, increasing overall management complexity.

VM Guest LAN

A more easily managed option is the Guest LAN feature. A global LAN segment is defined. Each virtual machine in the Guest LAN need only define a virtual network interface card (NIC) and connect to the Guest LAN segment. A Guest LAN can connect to a physical LAN by routing through an OSA adapter, as shown in Figure 2-1 on page 17.

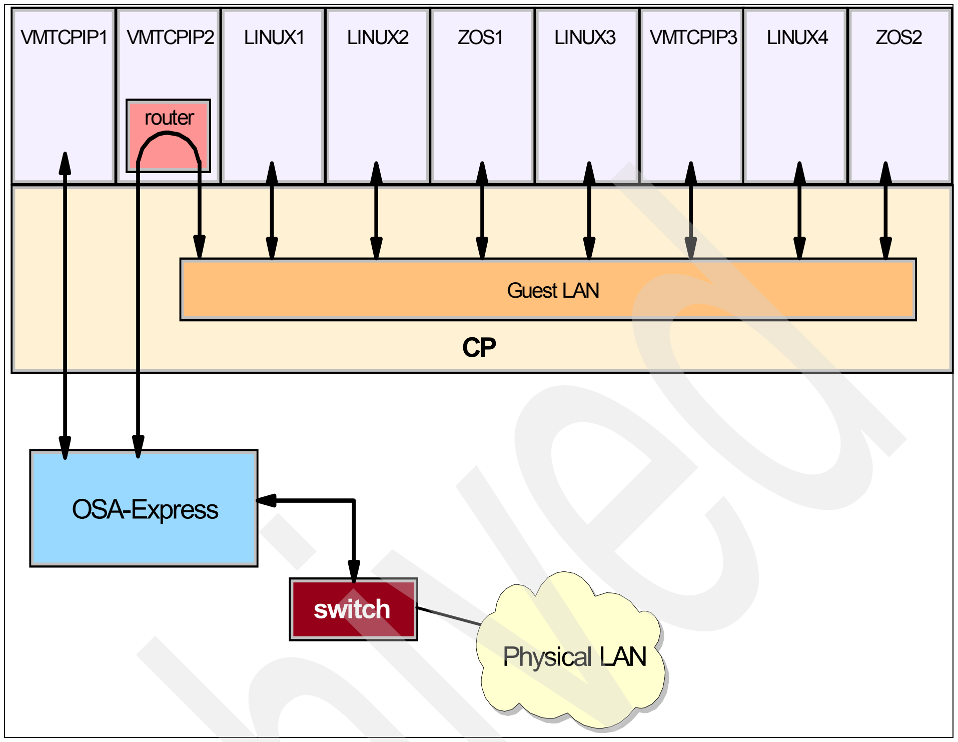


Figure 2-1 Example Guest LAN configuration

The z/VM Virtual Switch

With z/VM Version 4.4, a new type of virtual network is available. The z/VM Virtual Switch (VSWITCH) offers the ability to connect a virtual network directly to an external physical LAN, *without requiring a z/VM guest to act as a router*. Virtual guests acting as routers consume processor cycles to move packets between the virtual and physical networks, and routing adds latency to network traffic. Guests connected to a VSWITCH can be defined in the subnet of the external network to which they are connected.

In addition, a VSWITCH can participate in a IEEE 802.1Q VLAN. VLANs offer the following management and security improvements over physical network topologies:

- VLAN subnets (and the hosts on those VLANs) can be defined according to the logical organization of network traffic. Broadcast domains are defined by the VLAN and not limited to the physical LAN.

- VLAN administration can simplify network management. For example, when moving hosts on a VLAN, the network configuration can be changed at the switch (instead of each individual host).

An example VSWITCH configuration is shown in Figure 2-2.

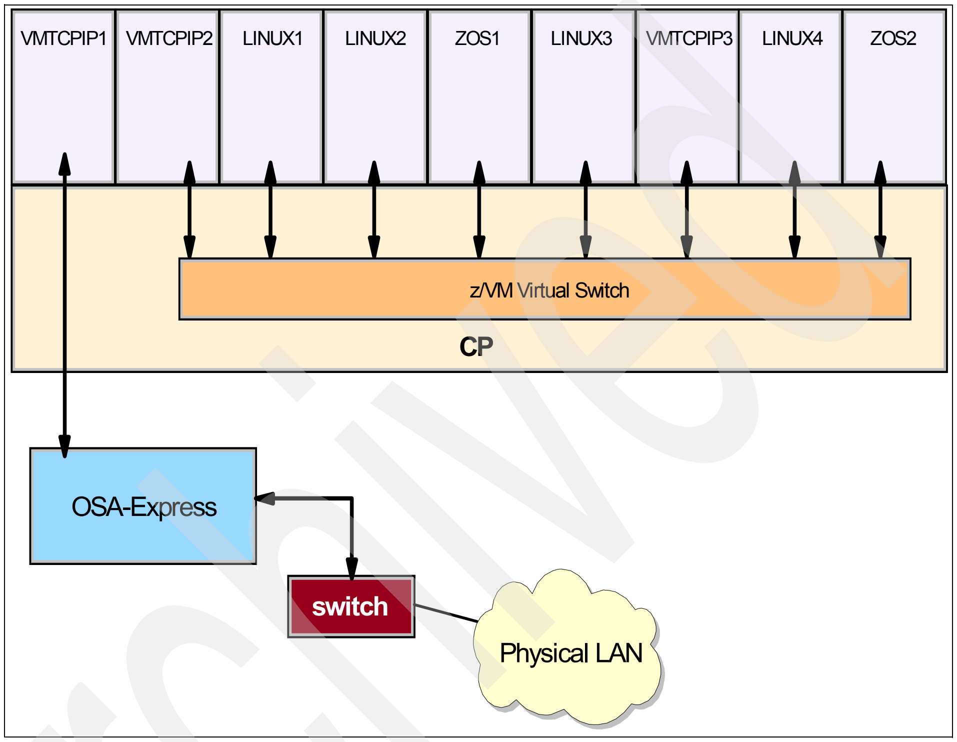


Figure 2-2 Example VSWITCH configuration

Note: VSWITCH is not available on IPV6 networks; access to external networks is limited to OSA-Express adapters. For information about VLAN concepts, see:

`http://net21.ucdavis.edu/newvlan.htm`

In a similar fashion, the HiperSockets Accelerator feature of z/OS can be used to reduce complexity and strengthen network security, as described in 7.1, “z/OS HiperSockets Accelerator” on page 138.

2.5.5 Configuring virtual networks

VM Guest LANs and z/VM Virtual Switches can be configured in the SYSTEM CONFIG file:

- DEFINE LAN
  Use the DEFINE LAN statement to create a VM Guest LAN at system initialization. To change the attributes of an existing VM Guest LAN, use the MODIFY LAN statement.
- DEFINE VSWITCH
  The DEFINE VSWITCH statement creates a z/VM Virtual Switch at system initialization. To change the attributes of an existing Virtual Switch, use the MODIFY VSWITCH statement.
- VMLAN
  The VMLAN statement sets global attribute defaults for VM Guest LANs.

Example 2-4 Defining virtual networks in the SYSTEM CONFIG file

```
VMLAN Limit Transient 0                                 1
Define LAN ,
  Type QDIO VMQDIO OwnerID System Maxconn 4 ,
  Grant USER1, USER2, USER3, USER4                      2
```

Example 2-4, illustrates how to configure virtual networks using the SYSTEM CONFIG file:

1. No transient Guest LANs are allowed. This prohibits general users from creating Guest LAN segments.
2. Create a persistent VM Guest LAN segment named VMQDIO. The Guest LAN is of type QDIO; connections to the Guest LAN is restricted to VM users USER1, USER2, USER3, and USER4. The maximum number of connections to the Guest LAN is limited to four.

Important: By default, a general user can create a transient Guest LAN. Limiting the number of transient Guest LANs to zero has the effect of removing the ability of general users to create Guest LANs.

2.5.6 Redefining a command privilege class

The MODIFY COMMAND statement can be used to redefine the privilege class assigned to a CP command. This is useful to make a command unavailable to users in the privilege class normally assigned to the command.

As an example, the CP SET SECUSER command (which associates a secondary user to a virtual machine) is normally authorized for privilege classes

A and C. In Example 2-5, we use the MODIFY COMMAND statement to remove authorization for SET SECUSER and authorize the command for the user-defined privilege class L.

Example 2-5 MODIFY COMMAND statement in the SYSTEM CONFIG file

```
Modify Command ,
  Set SubCmd SecUser ,
    IBMClass C ,
    PrivClass L
```

After it is re-IPLed, the SET SECUSER command will be authorized for privilege classes A and L.

Note: The CP COUPLE and DEFINE CTCA commands are authorized to class G users. This can allow two virtual machines to cooperatively establish a virtual network without system administrator authorization. To remove this potential, redefine authorization to the COUPLE or DEFINE CTCA commands.

Using the VMLAN statement in the SYSTEM CONFIG file (as discussed in 2.5.4, “z/VM virtual networking” on page 16) ensures that only authorized VM Guest LANs can be created.

2.6 The z/VM user directory

The z/VM user directory defines to the CP the characteristics of each virtual machine. The user directory exists on a CP-formatted disk in two forms:

- Source format that describes the user directory
- Compiled version (created from the source) of the directory used by the CP

The source format of the user directory can exist as a monolithic file that contains the entire user directory, or as several files to define the user directory. In either case, the compiled format is created using the DIRECTXA utility. Example 2-6 on page 21 illustrates the source format for a sample user directory entry.

Example 2-6 z/VM USER directory entry

```
USER USER1 USER1PW 128M 1G *                                  1
   INCLUDE IBMDFLT                                            2
   IPL 190 PARM AUTOCR                                        3
   LOGONBY USER2                                              4
   MACHINE ESA
   MDISK 0191 3390 21 5 440U1R
   MDISK 0201 3390 26 1087 440U1R WR READ WRITE MULTIPLE      5
   MDISK 0202 FB-512 V-DISK 5000 WV
   LINK TCPMAINT 592 592 RR                                   6
   DEDICATE 2800 2800                                         7
```

Notable security-related statements include:

1. The USER statement starts the user directory entry. Specific security-related operands are discussed in 2.6.1, "The USER directory entry statement" on page 21.
2. Additional statements common to multiple VM user directories can be included, as discussed in 2.6.2, "The INCLUDE statement" on page 23.
3. The IPL statement (discussed in 2.6.3, "The IPL statement" on page 23) specifies a device of Named Shared Segment (NSS) to boot at logon.
4. It is possible to define surrogate VM users, as discussed in 2.6.4, "The LOGONBY statement" on page 23.
5. User minidisks are defined using the MDISK statement; see 2.6.5, "The MDISK statement" on page 23.
6. Minidisks can be shared among virtual machines using the LINK statement, as discussed in 2.6.6, "The LINK statement" on page 24.
7. Devices can be dedicated to specific virtual machines, as discussed in 2.6.7, "The DEDICATE statement" on page 24.

2.6.1 The USER directory entry statement

The USER statement specifies the virtual name, password, virtual machine size, and CP command privilege class. User names are one to eight characters in length. Two virtual machine size operands are recognized; the first specifies address space size at logon, the second specifies the maximum address space size the user can specify using the CP DEFINE STORAGE command.

Specifying user passwords

The user password field specifies the one to eight character password required at logon. Special password options can be specified in the password field:

- NOLOG
  Specifying a password value of NOLOG prevents the user from logging on. NOLOG virtual machines are useful for managing shared VM resources. For example, assigning shared minidisks to a virtual machine that cannot log on is a mechanism to easily identify those minidisks.
- NOPASS
  If specified, the NOPASS option allows the user to log on *without* providing a password.

  > **Important:** NOPASS virtual machines should be considered a security exposure and should not be permitted in the VM user directory.

  If an ESM (such as RACF for z/VM) is installed, passwords might still be required if the NOPASS password option is specified.
- AUTOONLY
  The AUTOONLY password option specifies that the virtual machine can only be logged on using the CP XAUTLOG command. Terminal logons are disabled.
- LBYONLY
  The LBYONLY password option specifies that the virtual machine can only be logged on to by a surrogate user. A surrogate user is another VM user entitled to log on to this virtual machine.

  To log on as a surrogate user, use the CP LOGON command specifying the BY option (where the BY operand is the surrogate user ID). When prompted, the surrogate user must specify *their* specific password, not the password of the logon virtual machine. An LBYONLY virtual machine must have at least one surrogate user defined, as discussed in 2.6.4, “The LOGONBY statement” on page 23.

  > **Note:** Using the LBYONLY option, it is possible for many VM users to share access to a common virtual machine *without* sharing passwords. With journaling enabled, it is possible for a system administrator to audit activity.

Specifying privilege class

Virtual machine privilege class is assigned in the USER statement. Up to 32 single characters can be specified (each representing a defined command privilege class, as discussed in 2.4, “CP privilege classes” on page 13).

Note: If not specified or specified as “*”, the privilege class default privilege class or classes specified are assigned to the VM user (see 2.5.3, “Defining privilege classes” on page 16).

2.6.2 The INCLUDE statement

The INCLUDE statement names a profile entry to be included as part of a user definition. When constructing a user entry, ensure that statements in any included profile do not violate security policy for Linux virtual machines.

2.6.3 The IPL statement

The IPL statement is used to boot a virtual machine at logon. In a traditional z/VM environment, CMS is typically IPLed at logon. For Linux virtual machines, the device containing the Linux root file system can be specified to automatically load Linux at logon.

2.6.4 The LOGONBY statement

The LOGONBY statement defines up to eight VM users that can act as surrogate users. When logging on, VM expects the surrogate user to provide their specific password. Surrogate users are discussed in 2.6.1, “The USER directory entry statement” on page 21.

2.6.5 The MDISK statement

The MDISK statement defines minidisks and VDISKs owned by the virtual machine. It is possible for the user directory entry to specify passwords for minidisk sharing. MDISK passwords are one to eight characters in length; three types are recognized:

- Read mode password
- Write mode password
- Multiple write mode password

VM users linking to a minidisk (using the CP LINK command) are expected to provide the password appropriate for the desired access mode (read, write, or multiple write). For details about the disk access modes, consult the MDISK statement in *z/VM V4R4.0 CP Planning and Administration*, SC24-6043.

Passwords can be specified as ALL, in which case attempts to link to the minidisk succeed *without* requiring a password.

Important: To ensure a secure z/VM system, *always* require non-trivial passwords on shared minidisks; disallow use of the ALL password option. In addition, always require password prompting (disallow passwords passed as part of CP commands) by disabling the PASSWORDS_ON_COMMANDS feature, as discussed in 2.5.2, "System features" on page 15.

2.6.6 The LINK statement

The LINK statement allows a virtual machine to access a minidisk owned by another virtual machine. Read-only or read-write access can be granted. By using the LINK statement, you open the door for cross-virtual-machine access. Our recommendation is that you exercise extreme care if you decide to use LINK statements to authorize your Linux virtual machines to have access to other virtual machines' minidisks.

2.6.7 The DEDICATE statement

The DEDICATE statement assigns sole ownership of a real device to a virtual machine. For tape devices, DEDICATE assigns serial, shared access the device with other virtual machines. A DEDICATED device is owned and controlled by the virtual machine. Be selective when dedicating real devices to Linux virtual machines.

2.6.8 The OPTION statement

Special characteristics can be assigned to a virtual machine using the OPTION statement. Most options give the virtual machine higher authorization than they would normally receive. Consult *z/VM V4R4.0 CP Planning and Administration*, SC24-6043, for details about the OPTION statement.

Note: In keeping with the policy that Linux virtual machines should run with no higher than general user authorization, the OPTION statement should not be present in the directory entry for a Linux guest.

2.6.9 The SPECIAL statement

The SEPCIAL statement is used to define a virtual network interface card (NIC) to the virtual machine and connect the NIC to a virtual network (such as a Guest LAN segment). For Guest LANs, the SPECIAL statement is equivalent to performing the CP DEFINE NIC and COUPLE commands in order to connect the guest to a virtual network. The SPECIAL statement typically connects a virtual machine to a persistent Guest LAN.

2.7 Directory Maintenance Facility

Maintaining the user directory can be simplified using the automation and command interface provided by the Directory Maintenance Facility (DirMaint). Every directory entry statement is implemented as a DirMaint command.

Note: DirMaint is a separately licensed z/VM feature. With z/VM Version 4.4, DirMaint is shipped with the system installation media as a convenience. You must obtain a separate license to enable DirMaint.

DirMaint provides the following facilities to ensure system integrity when creating or updating the user directory:

- Error checking is performed to ensure only valid changes are applied to the user directory.
- Authorization to modify specific directory statements can be delegated to z/VM users.
- Modifications to the user directory are performed by DirMaint service machines (thus reducing the need to grant elevated authorization to other z/VM virtual machines).
- DirMaint can maintain the user directory for both local and remote z/VM systems (again reducing the need to grant elevated user authorization).

2.7.1 DirMaint security features

DirMaint supports the z/VM security strategy:

- Access to user IDs is password controlled. DirMaint maintains user passwords, with customer choice of administration control or user control. DirMaint also supports the use of an ESM for password control.
- Access to minidisks is controlled either by password or explicit link authorization (as determined by the minidisk owner). Minidisk passwords are optional for controlling minidisk directory links. DirMaint supports control of minidisk links by an ESM.
- VM system services are used to identify the originating user ID (and node ID) for all requests (local and remote). By default, requests are authenticated using the logon password for each DirMaint transaction. Unless prohibited by the system administrator, users can request suspension of authentication for requests made from their user ID. Remote requests never require authentication, and surrogate requests always require authentication.

- All DirMaint commands involving the DirMaint service machines (DIRMAINT, DATAMOVE, DIRMSAT) are auditable. A few DirMaint commands (CHECK, DEFAULTS, EXECDROP, GLOBALV, and HELP) are completely processed in the user’s virtual machine and are therefore not auditable by DirMaint. They might, however, be auditable by an ESM.

All messages generated by the DirMaint service machines are auditable. An exit routine can support customer-tailorable filtering of unnecessary audit details.

Note: For more information about DirMaint security, refer to *z/VM V4R4.0 General Information*, GC24-5991

2.8 RACF for z/VM

As mentioned earlier, there is a RACF product available on z/VM, and we recommend the use of RACF to secure the z/VM system itself. Refer to *z/VM Security and Integrity*, by Alan Altmark and Cliff Laking, for a discussion of this. For the Linux environment executing on z/VM, we demonstrate how the various operating system environments on the zSeries platform can interoperate using non-proprietary, open interfaces. We choose to use LDAP for directories, in this case having the LDAP server executing on a z/OS image. We use the same z/OS image to show how LDAP and RACF on z/OS can interoperate to create a secure environment.

Note: RACF for z/VM is a separately licensed feature. With z/VM V4.4, RACF is shipped with the system installation media as a convenience. You must obtain a separate license to enable RACF.

The value of using RACF as a back end for LDAP is that all your existing RACF users can be given access to Linux machines, using the user ID and password that they use for their z/OS-related work. Secondly, the password can never be compromised; it is always encrypted in the RACF database. Conversely, if you decide to have the root user ID in RACF, all your root passwords for all your Linux machines will have to be the same. If that single password is then compromised, access will be given to your entire set of Linux machines.

If you decide to have the root user ID managed from outside of your Linux machine, we strongly recommend that you do not use RACF on z/OS for that purpose, but instead leave the control in LDAP on z/OS.

For more information, refer to Chapter 7, “Using z/OS features in a Linux environment” on page 137.

3

Hardening a Linux installation

In this chapter, we examine options for creating a more secure Linux installation. We use the term hardening to indicate processes and procedures taken to reduce the likelihood of system compromise. Topics discussed include:

- Linux system logging
- Pluggable Authentication Modules
- Delegating superuser authority with sudo
- Securing Internet services with TCP_wrappers
- Securing Linux using Bastille

3.1 Linux system logging

Linux logging is controlled by the syslog utility. The syslogd daemon accepts incoming log messages from a variety of sources, such as:

- The Linux kernel
- System services (such as mail, cron, and PAM)
- Applications programs

In addition to the message text, incoming log messages indicate:

- The service or application generating the message (the *facility*)
- The severity of the message (the *level*)

The syslog utility adds a time stamp, host identifier, and process identifier to incoming messages, and then processes the messages according to the specification found in the /etc/syslog.conf configuration file. Collectively, the message facility and level are referred to as the message *selector*.

3.1.1 Configuring syslogd

The /etc/syslog.conf file is used to determine the disposition of incoming messages. Entries in /etc/syslog.conf are of the form:

```
facility.level action
```

Defined syslog facilities include:

auth	Authentication messages (deprecated)
authpriv	Authentication messages
cron	Messages from cron
daemon	Messages from system services (daemons)
kern	Kernel messages
lpr	Printing service messages
mail	Mail service messages
mark	Internal syslog time stamp messages
news	News service messages
syslog	Syslog messages
user	User application messages
uucp	uucp messages
local0 - **local7**	Messages from locally-defined services

Defined syslog levels include:

emerg	Fatal system error
alert	Errors that require immediate attention
crit	Errors that disable a system service
err	Errors that disable a system service component

warning	Warning message
notice	Normal message
info	Informational message
debug	Debug message

The action indicates where syslog is to direct incoming messages. Syslog can report messages to a:

- Log file
  Action specifies the log file name. Typically, the /var/log/messages file is specified for all syslog messages. If a file name is prefixed with the "-" character, syslog will not sync the file after each write.
- Remote host
  Action can specify a remote host name or IP address. In this case, the "@" precedes the IP address or host name. Log messages are then directed to the syslog running on that host. We discuss logging to a remote host in 3.1.2, "Using a central log server" on page 30.
- Console
  To log to the console, specify /dev/console as the action.
- Logged on user
  To send log messages to a user, specify the user name as the action. A list of users can be specified (each user name separated by a comma ","). If the action is specified as "*", log messages are directed to all logged on users.
- Named pipe
  To direct log messages to a named pipe, specify the "|" prefix before the pipe name.

Example 3-1 illustrates a sample syslog configuration file.

Example 3-1 Sample /etc/syslog.conf file

```
# /etc/syslog.conf - Configuration file for syslogd(8)
#
# Print most on /dev/console
kern.*;*.warn;news.emerg;mail.alert;authpriv.none        /dev/console
# all email-messages in one file
mail.*                                                  -/var/log/mail
# Warnings in one file
*.=warn;*.=err                                           /var/log/warn
# save the rest in one file
*.*;mail.none;news.none                                  /var/log/messages
# Log to a remote host
*.*                                                      @192.168.10.1
```

3.1.2 Using a central log server

The advantages of directing syslog messages to a central log server include:

- Simplified log administration
  With a central log server, log files from all Linux machines can be viewed and backed up from one location.
- Enhanced log file integrity
  If an intrusion occurs on a Linux server, the attacker can attempt to remove any record of the intrusion from the log files. A central log server can be hardened and tightly controlled. In the event a local log file is modified, the central log server can maintain an unedited version.

To configure syslog to accept incoming log messages, use the **-r** option with **syslogd**. For SLES, add the option to the SYSLOG_PARAMS variable in the /etc/sysconfig/syslog file:

```
#
# if not empty: parameters for syslogd
# for example SYSLOGD_PARAMS="-r -s my.dom.ain"
#
SYSLOGD_PARAMS="-r"
```

To make the change take effect, syslog must be restarted. When restarted, syslog records the change to the log file (/var/log/messages):

```
Mar 24 10:35:31 lnxsu2 syslogd 1.4.1: restart (remote reception).
```

Note: To configure remote hosts to log to the central log server, use the @ syntax in the host's /etc/syslog.conf file (specify the central log server IP address or host name).

To test the new configuration, use the **logger** command. For example, on the remote host, issue the command:

```
logger -p syslog.warning "test[123]: testing remote logging"
```

This generates a message in the log file of the central log server:

```
May 24  10:56:56 lnxsu3 root: test[123]: testing remote logging
```

Note: The **-p** option to the logger command specifies the facility and priority of the message.

3.2 Pluggable Authentication Modules

The default mechanism used for user authentication in Linux uses Pluggable Authentication Modules (PAM). PAM provides centralized authentication for common services such as login, FTP, Telnet, and SSH. PAM is implemented using dynamic load libraries (DLLs). Authentication can be customized for PAM-aware services using PAM configurations files. Figure 3-1 illustrates PAM operation.

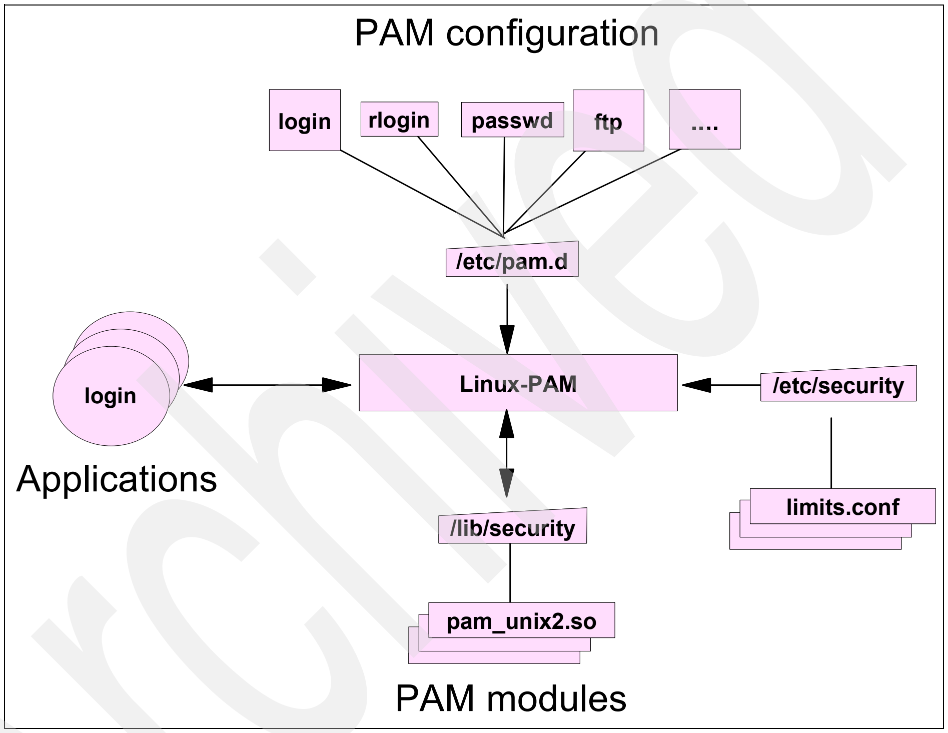


Figure 3-1 PAM architecture

Authentication for each PAM-aware application is configured by a configuration file in the /etc/pam.d directory (the file name indicates the service it configures). PAM modules implement user authentication; by default, PAM modules are located in the /lib/security directory. Some PAM modules read global configuration files located in the /etc/security directory.

3.2.1 PAM configuration files

A PAM configuration file consists of lines in the form:

```
module-type   control-flag   module-path   arguments
```

Each field is described in the following sections.

PAM module types

Four types of authentication mechanisms (referred to as module types) are supported. Each module type is described in Table 3-1.

Table 3-1 PAM module types

Type	Description
auth	Used to authenticate a user based on user name and password
account	Used to grant or deny access based on account policies (such as time of day)
password	Used to authorize user password updates
session	Used to apply settings before a session is established

Note: More than one line can be specified for any PAM module type. This is referred to as *module stacking*. In this case, each module is executed in first-to-last order. Final authentication depends on the PAM control flags specified.

PAM control flags

The PAM control flag field determines how authentication proceeds based on a module return code. Table 3-2 lists the four possible flags.

Table 3-2 PAM control flags

Flag	Action
required	Required modules must return a success code for authentication to succeed. In every case, all stacked modules are executed, and final authentication succeeds only if these also succeed.
requisite	Requisite modules terminate immediately on authentication failure. Stacked modules are executed on success.
sufficient	Indicates a successful return code is sufficient for authentication. If a previous *required* module has failed, authentication will fail.
optional	Optional modules are not considered crucial. PAM ignores success or failure codes.

PAM module paths

The module path field specifies the DLLs name used to implement the PAM check. Names can be specified as absolute paths or relative paths (in which case, the modules are assumed to reside in the /lib/security directory).

PAM module arguments

PAM module arguments are generally optional and recognized by specific PAM modules. Table 3-3 shows some typical PAM arguments.

Table 3-3 PAM arguments

Argument	Description
debug	Use the syslog call to log debugging information to the system log.
audit	Use verbose debugging.
nullok	Allows the use of blank passwords. The default is to require non-blank passwords.
no_delay	By default, if authentication fails, the module delays 1 second before prompting for a new password. This argument causes the module to immediately prompt for a new password.
no_warn	Instructs the module not to pass warning messages to the application.
use_first_pass	The module should not prompt for a password. Instead, it should use the password obtained from the preceding auth module. If that password fails, the user is not authenticated. This option is intended for auth and password modules only.
try_first_pass	The module should attempt authentication with the password obtained from the preceding auth module. If that password fails, the user is prompted for a password. This option is intended for auth modules only.

PAM configuration for the login service

To illustrate PAM configuration, we examine the default SUSE LINUX Enterprise Server Version 8 (SLES8) PAM configuration for the login service in Example 3-2 on page 34.

Example 3-2 The default /etc/pam.d/login configuration for SLES8

```
#%PAM-1.0
auth      requisite  pam_unix2.so     nullok     #set_secrpc           1
auth      required   pam_securetty.so                                  2
auth      required   pam_nologin.so                                    3
auth      required   pam_mail.so                                       4
account   required   pam_unix2.so
password  required   pam_pwcheck.so   nullok                           5
password  required   pam_unix2.so     nullok use_first_pass use_authtok
session   required   pam_unix2.so     none     # debug or trace
session   required   pam_limits.so                                     6
```

PAM modules in this example allow for:

1. Standard UNIX password checking is provided by pam_unix2.so.
   The pam_unix.so module supports all four PAM modules types (auth, account, password, and session). Password checking for pam_unix2.so depends on the contents of the /etc/nssswith.conf file.

   When used as a password module type, pam_unix2.so can require users to change expired passwords. Its actions are determined by the following passed arguments:

   – The nullok argument permits changing an empty password (normally, empty passwords cause an account to be locked).

   – The use_first_pass argument locks the selection of the old and new password to the ones established by the previous stacked module (in this case, pam_pwcheck.so).

   – The use_authok argument sets the new password to the value passed by the previous stacked module (again, pam_pwcheck.so in this example).

   When used as a session module type, pam_unix2.so simply logs the user and service type to syslog. It does this at the beginning and end of the session.

2. Superuser login is restricted to secure terminals.
   The pam_securetty.so module enables superuser login to be limited to secure terminals. This is discussed in 3.2.2, “Limiting superuser login to secure terminals” on page 35.

3. Normal users can be denied login access.
   The pam_nologin.so module supports auth and account PAM module types and is discussed in 3.2.3, “Restricting user login” on page 35.

4. Mail notification can be sent using PAM.
   The pam_mail.so module only supports the auth module type. It provides the “You have new mail” service to the user. It also sets the MAIL environment variable to the user’s mail directory.

5. New user passwords can be tested before being set.
   The pam_pwcheck.so module supports only the password module type. It checks passwords when initially set or changed. It does not change the password; rather, it calls the cracklib routine to check password strength. Other password checks can be set in the /etc/login.defs file.
6. System resource limits can be set based on user.
   The pam_limits.so module supports only the session module type. This modules sets the limits of systems resources a user can use in a session. Superuser is not affected by this module. This module depends on the configuration file /etc/security/limits.conf.

3.2.2 Limiting superuser login to secure terminals

Superuser access should always be restricted to a secure terminal. Network login by root should be denied. This can be enforced using the pam_securetty.so module. If enabled when a superuser logs in, the pam_securetty.so modules denies access unless the login originates from a terminal device listed in the /etc/securetty file. The default secure device defined for SLES8 is shown in Example 3-3 (the ttyS0 device is the VM console).

Example 3-3 The SLES8 default /etc/securetty file

```
# This file contains the device names of tty lines (one per line,
# without leading /dev/) on which root is allowed to login.
ttyS0
```

Important: When logging on from the VM console, passwords are not hidden on the terminal. Be careful to avoid exposing passwords (especially the root password) to anyone looking at the login terminal.

If superuser authority is required for a network session, log in as a non-privileged user and then issue the `su` command to switch to superuser. Alternatively, consider using sudo to delegate authority as discussed in 3.3, "Delegating superuser authority with sudo" on page 37.

The pam_securetty.so module is not enabled for SSH in the default SLES8 installation. It might be desirable to deny superuser SSH network access even though the data is encrypted. This can help prevent an attacker from guessing the root password.

3.2.3 Restricting user login

At times, it might be useful to restrict non-superuser logins (for example, when performing system maintenance). The pam_nologin.so module can be used to deny access to the system. When enabled, pam_nologin.so checks for the existence of the /etc/nologin file. If found, pam_login.so denies access and displays the contents of the file to the user. An alternate nologin file can be specified by supplying the `file=`*`pathname`* PAM argument.

Note: The nologin mechanism does not affect superuser login. Superusers can log in even if the /etc/nologin file exists.

3.2.4 Mandatory access control

Currently, Linux relies on a Discretionary Access Control (DAC) mechanism when granting access to system objects (such as files and directories). With DAC, user authorization is based solely user identity and file ownership. In a DAC environment, there are only two types of users:

- Normal users
- Superuser

This model is referred to as discretionary, because individual users have the ability to specify authorization to files that they own. In addition, a user can delegate permission to change access control. Programs executed by a user inherit permissions granted to the user and are free to change access to the user’s files. This can create a security hole for malicious software.

The preferred security mechanism for enterprise computing is Mandatory Access Control (MAC). In a MAC environment, users cannot specify access control to the objects they own. Instead, file and directory authorization is granted based on a system policy. Each user has a role defined by the system security administrator, and file access is granted based on that role. Roles are enforced at the kernel level and cannot be circumvented by users.

3.2.5 Linux Security Module (LSM)

This section contains an edited excerpt from "Linux Security Modules: General Security Support for the Linux Kernel," by Chris Wright et al., in *2002 USENIX Security Symposium*, available at:

`http://www.usenix.org/events/sec02/wright.html`

At the Linux Kernel 2.5 summit in 2001, the National Security Agency (NSA) presented their work on Security-Enhanced Linux (SELinux), an implementation of a kernel-based access control architecture. Linus Torvalds appeared to accept that a general access control framework was needed, but favored a more general approach that should be:

- Generic (security models implemented as loadable modules)

- Conceptually simple, minimally invasive, and efficient
- Able to support existing POSIX.1e capabilities as a loadable module

Based on these principles, the Linux Security Modules (LSM) project was created. LSM is a general purpose, access control framework for the Linux kernel that implements access control mechanisms as loadable kernel modules. LSM support is available in the 2.6 kernel (announced in July 2003). A number of existing access control implementations (including SELinux) have adapted the LSM framework. LSM architecture mediates access to internal kernel objects (for example, tasks, inodes, and open files) using kernel hooks, as shown in Figure 3-2.

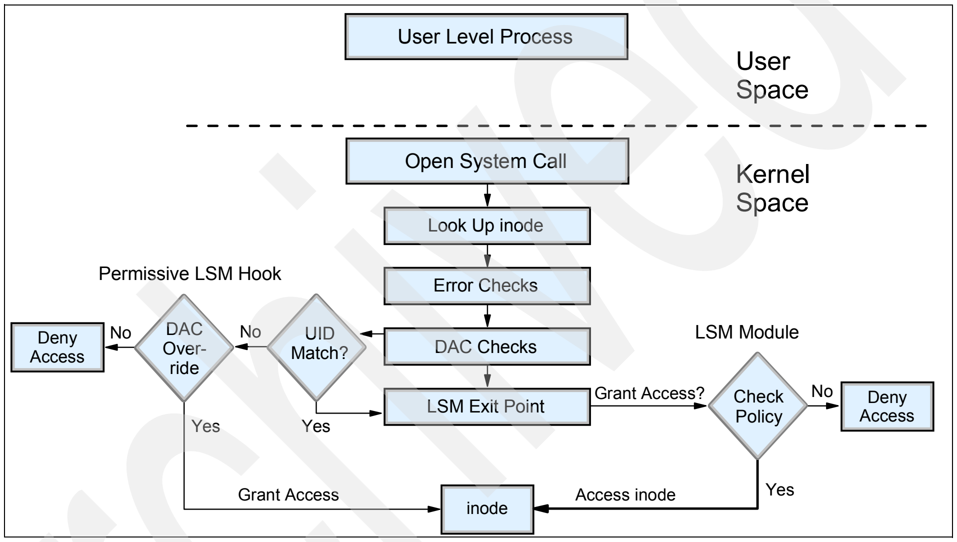


Figure 3-2 LSM architecture

With LSM, user processes execute system calls traversing existing kernel logic. However, before a kernel object is accessed, an LSM module must first grant access.

Note: There are no current plans to support LSM in Linux for zSeries.

3.3 Delegating superuser authority with sudo

Sudo is an open source security tool distributed with SLES8 that enables administrators to delegate authority to specific commands. The sudo home page is located at:

http://www.courtesan.com/sudo

With sudo, the system administrator can allow non-privileged users to execute commands that would normally require higher authorization. The benefits of using sudo include:

- Delegation of administrative tasks.
  Authorization to specific privileged commands can be delegated based on the role assigned to a non-privileged user.
- Sharing passwords is not required.
  When a sudo-authorized command is executed, the non-privileged user is prompted for their own password (not the root password).
- Commands executed under sudo are logged.
  In addition to logging incorrect password attempts, sudo logs all command executions (thus creating an audit trail).
- Commands can be authorized to execute under any user (not just root).
  This allows administrators to finely control command authorization.
- Configuration is centralized.
  A single configuration file can manage sudo authorization for both local and remote hosts.

Important: Caution should be exercised when delegating root authority. It is possible to introduce security vulnerabilities using sudo (as discussed in 3.3.4, “Security considerations with sudo” on page 42).

3.3.1 Configuring sudo

The /etc/sudoers file is used to configure sudo. This file should be writable only by root and must be edited using the **visudo** command.

Note: The **visudo** command ensures that the /etc/sudoers file is safely edited. If more than one editor attempts to open the file, the visudo command will fail:

```
# visudo
visudo: sudoers file busy, try again later.
```

The /etc/sudoers file contains three types of entries:

- Aliases
  Aliases are name/value pairs in the form:

  ```
  Alias_type NAME = item1, item2, ...
  ```

  Four types of aliases are recognized:

  – Host_Alias
    Host aliases specify host names, IP addresses, or network addresses.

  - User_Alias
    User aliases specify users or group names or numeric IDs.
  - Cmnd_Alias
    Command aliases specify commands to execute.
  - Runas_Alias
    Run as aliases specify a user under which to execute commands.
- Defaults
  Configuration options can be overridden from their default values using the Defaults specification.

  Tip: To see the default configuration options, execute the **`sudo -V`** command as root.

- User privileges
  The user privileges specification defines which users are authorized to execute specific commands.

  Users privileges are specified by lines of the form:

  ```
  user-spec host-spec = (runas-spec) cmd-spec
  ```

  Where:

`user-spec`	Specifies users to be granted authorization. User name, group name, and User_Alias specifications are accepted.
`host-spec`	Specifies the hosts on which authorization is to be permitted. Host name, IP addresses, and Host_Alias specifications are accepted.
`runas-spec`	Specifies the user ID with which to execute the command. User name and Runas_Alias specifications are accepted. By default, commands are executed by root under sudo.
`cmd-spec`	Specifies a command list. Commands and Cmnd_Alias specifications are accepted.

Note: The /etc/sudoers file grammar has extensive regular expression and wildcard expansion capabilities. Details can be found in the man page for sudoers.

To illustrate a sudo configuration, we consider the /etc/sudoers file shown in Example 3-4 on page 40.

Example 3-4 The /etc/sudoers file

```
# sudoers file.
#
# This file MUST be edited with the 'visudo' command as root.
#
# See the sudoers man page for the details on how to write a sudoers file.
#

# Host alias specification
Host_Alias      SUBNET1=172.22.10.0/255.255.255.0         1
# User alias specification
User_Alias      NETADMIN=user1                            2
User_Alias      HTTPADMIN=user2
# Cmnd alias specification
Cmnd_Alias      NETCMDS=/sbin/ifconfig hsi2 *             3
Cmnd_Alias      HTTPCMDS=/usr/sbin/apachectl
# Defaults specification

# User privilege specification
root            ALL=(ALL)      ALL                        4
NETADMIN        SUBNET1=NETCMDS
HTTPADMIN       ALL=HTTPCMDS
```

This configuration file specifies:

1. We define a host alias specifications based on network subnet assignment:
   – SUBNET1 specifies all host on the 172.22.10.0 network.
2. We define two user aliases:
   – The NETADMIN group contains user1.
   – The HTTPADMIN group contains user2.
3. We define two command aliases:
   – NETCMDS for the **`/sbin/ifconfig`** command operating on the hsi2 interface.
   – HTTPCMDS for the **`/usr/sbin/apachectl`** command.

   Note: We explicitly restrict the NETCMDS command group to operate *only* on the hsi2 interface. In contrast, the HTTPCMDS command group permits any operand to the **`/usr/sbin/apachectl`** command.

4. Privilege specification indicates which users are authorized to execute what commands:
   – The sudo reserved word ALL indicates that the root user is authorized to execute any command, as any user, on any host under sudo.

– NETADMIN users are granted authorization to NETCMDS only if executed on a host in SUBNET1.

– HTTPADMIN users are granted authorization to HTTPCMDS on any host.

3.3.2 Using the sudo command

Sudo is invoked using the **sudo** command. In Example 3-5, we see user1 is not normally authorized to execute the **/usr/sbin/apachectl** command. Using the **sudo** command, however, user1 is granted authorization to execute the command.

Example 3-5 Executing the sudo command

```
$ /usr/sbin/apachectl configtest
Ouch! ap_mm_create(1048576, "/var/lib/httpd/mm.4071") failed
Error: MM: mm:core: failed to open semaphore file (Permission denied): OS: No such file or ..
$ sudo sudo /usr/sbin/apachectl configtest
Password:
[Wed Mar 24 08:54:55 2004] [warn] module ssl_module is already loaded, skipping
[Wed Mar 24 08:54:55 2004] [warn] module mod_ssl.c is already added, skipping
Syntax OK
```

In order to execute a command under sudo, the user is first prompted for their password. After the user is authenticated, sudo remembers the user password using a ticket lasting for five minutes. While the ticket is valid, sudo does not prompt for a user password.

Command line options for sudo

Table 3-4 lists some of the command line options used by the sudo command.

Table 3-4 Common options to the sudo command

Option	Description
-u *user*	Execute the command as the specified user.
-k	Expire the user's ticket. User is required re-enter password next time the **sudo** command is executed.
-v	Validate the user's ticket for the amount of time (the default is five minutes). If required, the user must re-enter the user password.
-l	List the authorized commands for the user on the current host.

Example 3-6 on page 42 illustrates the use of the **-l** option to list authorized commands.

Example 3-6 Listing authorized sudo commands

```
$ sudo -l
Password:
User user1 may run the following commands on this host:
    (root) /usr/sbin/apachectl
```

3.3.3 Command logging with sudo

Commands executed under sudo are logged to syslog using the “local2” facility by default. All sudo commands are logged, both when successfully authenticated, and when authentication is denied. By default, successful authentications are logged using priority “notice.” Unsuccessful authentications are logged at priority “alert.” Example 3-7 shows example log messages generated for both successful and unsuccessful authentication.

Example 3-7 Logging commands executed under sudo

```
Aug 24 11:15:22 lnxsu2 sudo:   user1 : TTY=pts/1 ; PWD=/home/totibm ; USER=root ; COMMAND=list
Aug 24 11:16:05 lnxsu2 sudo:   user1 : 3 incorrect password attempts ; TTY=pts/1 ;
PWD=/home/user1 ; USER=root ; COMMAND=/usr/sbin/apachectl
```

3.3.4 Security considerations with sudo

When executing commands, sudo attempts to minimize security risks. For example, sudo will clear unsafe environment variables such as IFS and ENV and variables used for dynamic loading (the LD_* variables). When searching for commands, sudo checks the current directory (“.”) last when specified in the PATH environment variable.

Note: The PATH variable is *not* modified when passed to the executed command.

It is important to be aware that sudo configuration can introduce security exposures. For example, allowing access to the **vi** command to a non-privileged user can allow the user to break out into a root shell (by simply executing the **:shell** subcommand). To reduce security risks:

- Limit the number of users granted enhanced authority with sudo.
- Limit the number of commands available to sudo-authorized users.
- Do not authorize access to shell or interpreted commands that allow escapes to a shell.

3.4 Securing Internet services with TCP_wrappers

Internet services provided by the inetd daemon (such as Telnet, FTP, rlogin, and rsh) are inherently insecure. These services pass unencrypted data over the network (including user IDs and passwords). By default, Internet services are disabled in most Linux distributions. Before enabling the inetd daemon, consider using an alternate secure service such as Secure Shell (discussed in Chapter 4, "Secure Sockets Layer and the Secure Shell" on page 51).

Note: Although Internet services are typically disabled, most distributions include TCP_wrappers in the default /etc/inetd configuration file.

Some sites might choose to enable specific Internet services (for example, to run an anonymous FTP server). In these cases, it is important to take precautions to minimize security exposures. A common facility to secure Internet services is TCP_wrappers. TCP_wrappers provide the ability to filter incoming connections from Internet services (such as FTP, Telnet, or SSH). Filtering can be applied without changing the underlying server software. Most TCP/IP applications implement the standard client/server model:

1. A client initiates a service request (a connection) on a socket.
2. The server listening on the socket responds to the connection request.

TCP_wrappers are implemented by the tcpd dameon, which interposes an additional layer (or wrapper) between the client and server. Clients requests are examined and access controls applied to determine whether to allow or deny the connection. TCP_wrappers provide the ability to log incoming client requests to the system logger (the syslogd daemon).

Access control is implemented using two configuration files:

- The /etc/hosts.allow file
  This file contains the specification for allowed client access.
- The /etc/hosts.deny file
  This file contains the specification for denied client access.

Note: TCP_wrappers provide an access control mechanism based on the examination of incoming TCP/IP packets. TCP_wrappers cannot protect against man-in-the-middle attacks, or situations where a network router has been compromised. In addition, all unencrypted network data is open for inspection. TCP_wrappers should only be considered as part of a "defense-in-depth" strategy.

3.4.1 TCP_wrappers access control specification

The /etc/hosts.allow and /etc/hosts.deny files determine the restrictions imposed on services managed by TCP_wrappers Entries in these files have the form:

```
service:hostname:domain
```

The domain name is optional. Service can specify a specific Internet service ("ftp" for example), or the "ALL" keyword can specify all Internet services. The host name can specify a particular host or a wildcard to match several hosts. Wildcard specifications are shown in Table 3-5.

Table 3-5 Wildcard specifications in the /etc/hosts.allow and /etc/hosts.deny files

Wildcard	Description
ALL	Matches all hosts.
LOCAL	Matches any hosts specified with a host name without a domain name. This is used to match hosts in the local domain.
UNKNOWN	Matches users or hosts whose a name or address is unknown.
KNOWN	Matches users or hosts whose a name or address is known.
PARANOID	Matches hosts whose host name does not match its IP address.
EXCEPT	An operator that provides exceptions to matches. It takes the form: `list1 EXCEPT list2`. Hosts matched in both `list1` and `list2` are excluded.

Note: The KNOWN and UNKNOWN wildcards are subject to the availability of Domain Name Systems (DNS). If DNS is unavailable, these wildcards might not match properly. When using PARANOID, the host name and addresses will not match if DNS is unavailable and connections will be refused.

The operation of TCP_wrappers is illustrated in Figure 3-3 on page 45.

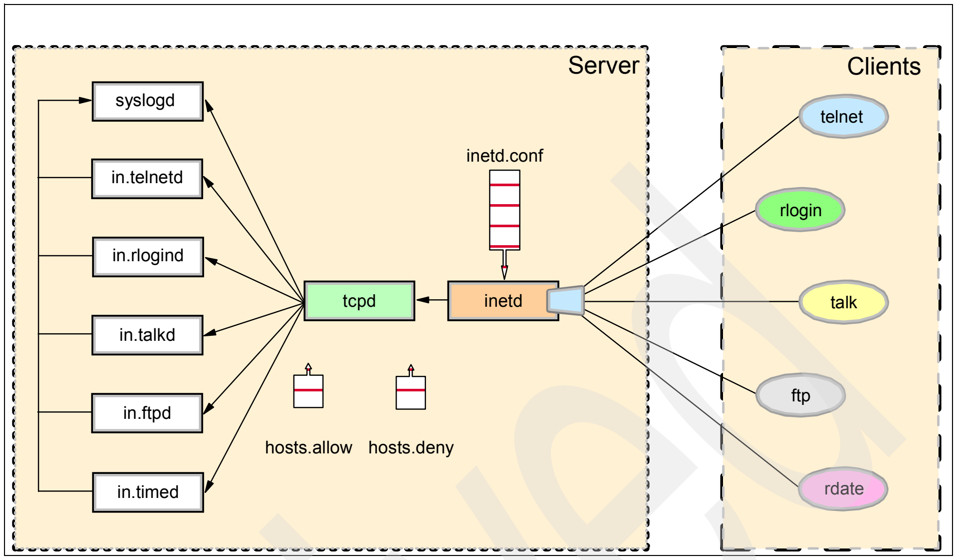


Figure 3-3 TCP_wrappers operation

When a client request is received, the inetd daemon runs the wrapper program (tcpd) passing to it the service process name (for example, in.ftpd). This name is used for both logging and to determine the access control to the service (based on the contents of the /etc/hosts.allow and /etc/hosts.deny files).

3.4.2 Configuring TCP_wrappers

TCP_wrappers is enabled in the /etc/inetd configuration. Example 3-8 on page 46 illustrates enabling TCP_wrappers for the Very Secure FTP daemon (vsftpd).

Example 3-8 Sample /etc/inetd.conf file

```
# <service_name> <sock_type> <proto> <flags> <usr> <server_path> <args>
# These are standard services.
#
# ftp   stream  tcp     nowait  root    /usr/sbin/tcpd  in.ftpd
ftp   stream  tcp     nowait  root    /usr/sbin/tcpd  vsftpd
#
# If you want telnetd not to "keep-alives" (e.g. if it runs over a ISDN
# uplink), add "-n".  See 'man telnetd' for more details.
# telnet        stream  tcp     nowait  root    /usr/sbin/tcpd  in.telnetd
# nntp  stream  tcp     nowait  news    /usr/sbin/tcpd  /usr/sbin/leafnode
# smtp  stream  tcp     nowait  root    /usr/sbin/sendmail    sendmail -L
sendmail -Am -bs
```

In the example, TCP_wrappers is invoked from the inetd daemon to service FTP requests. When an FTP request is received, the actual service daemon name (vsftpd) is passed to TCP_wrappers. Based on the access control specified in the /etc/host.allow and /etc/host.deny files, TCP_wrappers will either reject the request, or invoke vsftpd to process an allowed request.

Note: When modifying the /etc/inetd file, be sure to restart the inetd daemon in order to make changes take effect.

Access control specification

To illustrate access control specification, we consider the /etc/hosts.allow file shown in Example 3-9.

Example 3-9 The /etc/host.allow file

```
#wrapper allow list
ftp:9.12.9.35
ftp:9.12.9.41
```

In the example, we see FTP requests are permitted from two client machines (9.12.9.35 and 9.12.9.41). Example 3-10 illustrates the /etc/hosts.deny file.

Example 3-10 The /etc/hosts.deny file

```
#wrapper deny list
in.rshd: ALL
```

You can allow access to all but a few specific hosts. Put in your hosts.allow file ALL for a service, but list the ones you are denying access to in the hosts.deny file. The entry ALL:ALL opens your system to all hosts for all services.

Note: Using IP addresses instead of host names is more secure, because host names can be compromised through the DNS records by spoofing attacks, where an attacker pretends to be another host.

3.5 Securing Linux using Bastille

Bastille Linux is an open source project to “harden” a Linux installation by adjusting various system settings. The Bastille home page is available at:

`http://www.bastille-linux.org`

Bastille is a useful tool for system administrators to discover and tighten Linux security settings. A graphical user interface enables administrators to choose security settings appropriate for their environment. Bastille automatically implements changes based on selections provided by the user.

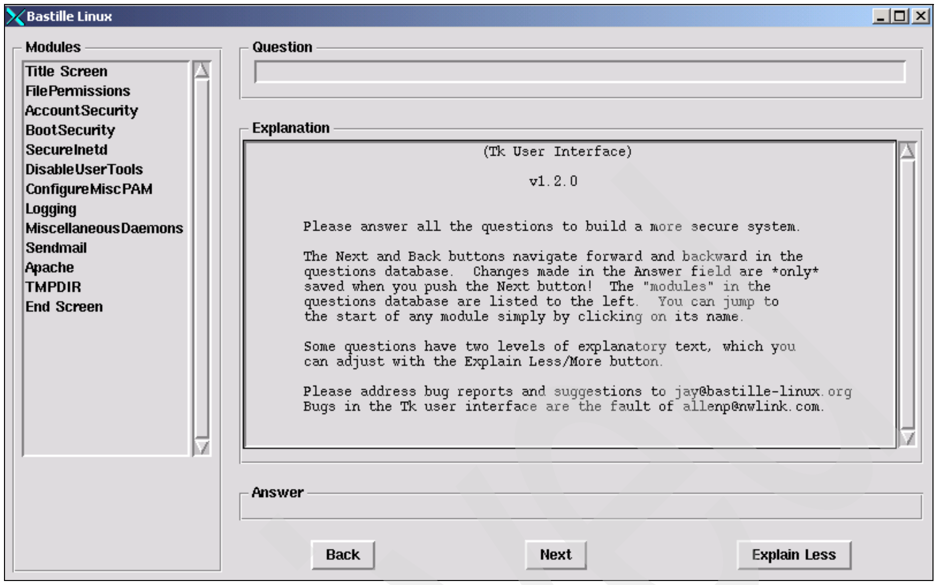
Note: The version of Bastille provided with SLES8 (1.2.0) does not work on Linux for zSeries. However, Bastille Version 1.3.0 will work if installed from the source code distribution (because Bastille is written in Perl, no compilation is required). Bastille Version 1.3.0 can be obtained from the Rpmfind.net server, available at:

`http://rpmfind.net`

Choose the Mandrake 1.3.0 package.

3.5.1 Configuring security settings with Bastille

To configure security settings with Bastille, use the **`InteractiveBastille`** command. This initiates the Bastille configuration interface shown in Figure 3-4 on page 48. Be aware that you need an X Windows server in order to run the application.



```
                    (Tk User Interface)

                          v1.2.0


       Please answer all the questions to build a more secure system.

       The Next and Back buttons navigate forward and backward in the
       questions database.  Changes made in the Answer field are *only*
       saved when you push the Next button!  The "modules" in the
       questions database are listed to the left.  You can jump to
       the start of any module simply by clicking on its name.

       Some questions have two levels of explanatory text, which you
       can adjust with the Explain Less/More button.

       Please address bug reports and suggestions to jay@bastille-linux.org
       Bugs in the Tk user interface are the fault of allenp@nwlink.com.
```

Figure 3-4 Bastille main entry screen

Security settings are configured using Perl modules (displayed on the left side of the screen). Bastille selects security settings based on responses from a series of questions posed to administrators. Bastille provides modules to customize settings for:

- File Permissions
  This module enables you to set restrictive file permissions on administrative utilities.
- Account Security
  This module is used to adjust user account settings.
- Boot Security
  This module adjusts Linux boot settings (such as the Crtl-Atl-Del key stroke behavior).
- SecureInetd
  This module configures TCP_wrappers settings.
- DisableUserTool
  This module configures user access to system tools (such as the gcc compiler).
- ConfigureMiscPAM
  This module configures PAM authentication settings.
- Logging
  This module configures system log settings.

- MiscellaneousDaemons
  This module can disable daemons that are often unnecessary and pose potential security risks.
- Sendmail
  This module enables you to disable daemon mode for the sendmail daemon. Bastille will ask you if you would like to run sendmail every few minutes to process the queue of outgoing mail.
- Apache
  This module can adjust security settings for the Apache Web server.
- TMPDIR
  Many programs use the /tmp directory dangerously. This module can provide more secure alternatives to the /tmp directory (using the TMPDIR or TMP environment variables).

Note: Bastille offers a simple method to change system settings. Be aware however that some settings can deny yourself (or other users) access to a command or service.

3.5.2 Reverting changes

Bastille has the ability to revert system settings using the **`UndoBastille`** command. Information about how to undo changes is stored in the /var/log/Bastille/undo directory. Before committing changes to system settings, Bastille copies existing configuration files to the appropriate location in the I/var/log/Bastille/undo/backup directory. A record of all actions performed by Bastille is saved in the /var/log/Bastille/action-log file. Executing the **`UndoBastille`** command restores configuration files from the backup directory.

Important: Changes cannot be undone by simply removing Bastille. To restore system settings *and* remove Bastille, execute **`UndoBastille`** before removing Bastille.

Before committing changes, Bastille logs settings to the /etc/Bastille/config file. Therefore, we recommend that you back up the config file before running Bastille. Consider doing a complete system backup as well.

3.5.3 Copying the Bastille setup to other hosts

If you have a number of similarly configured hosts large, it is possible to automate Bastille configuration:

1. Run the **`InteractiveBastille`** command on the first host, choosing the desired security settings.
2. Install Bastille on the other hosts. Copy the /etc/Bastille/config configuration file to each host.
3. Run the **`BastilleBackEnd`** command on each new host to commit the changes.

4

Secure Sockets Layer and the Secure Shell

In this chapter, we describe using Secure Sockets Layer and Secure Shell to encrypt network data. Topics include:

- Introduction to Secure Sockets Layer
- Enabling OpenSSL in Apache
- Using hardware acceleration with OpenSSL
- Secure Shell overview
- Secure network access using SSH
- File transfer and remote command execution
- Authentication without passwords
- Secure tunneling using port forwarding
- X forwarding
- Securing VNC using port forwarding

4.1 Introduction to Secure Sockets Layer

Secure Sockets Layer (SSL) is a network protocol developed by Netscape Communications. Originally conceived to secure HTTP communications, because SSL operates below the application layer and above TCP/IP, it can be used to secure other application protocols (such as LDAP). SSL offers encryption and authentication using digital certificates.

OpenSSL is an SSL implementation available for Linux on zSeries. The OpenSSL Web site is available at:

http://www.openssl.org

In this section, we look at using OpenSSL to secure HTTP transactions using the mod_ssl Apache module.

4.2 Enabling OpenSSL in Apache

The mod_ssl package is distributed in RPM format and is available with SLES8:

```
# rpm -qa | grep -i ssl
openssl-0.9.6g-20
perl-Net_SSLeay-1.18-27
sslwrap-2.10-336
mod_ssl-2.8.10-36
```

OpenSSL provides key and certificate generation functions. The mod_ssl module provides strong encryption for the Apache web server. These two features can be obtained from:

http://www.modssl.org

To configure mod_ssl, we:

1. Creating SSL keys.
2. Generating an SSL certificate.
3. Activating mod_ssl.
4. Configuring mod_ssl.

4.2.1 Creating SSL keys

We first create an RSA key pair using the **openssl genrsa** command, as shown in Example 4-1 on page 53.

Example 4-1 Generating an RSA key pair

```
# openssl genrsa -des3 -out /etc/httpd/ssl.key/server.key 1024
Generating RSA private key, 1024 bit long modulus
..............................++++++
.........................++++++
e is 65537 (0x10001)
Enter PEM pass phrase:********
Verifying password - Enter PEM pass phrase:
```

This generates keys using 1024 bits. The **-des3** option specifies the generated private key is to be encrypted using Triple DES. To encrypt the private key, the command prompts for a Privacy-enhanced Electronic Mail (PEM) passphrase. This value must used whenever the private key is accessed.

Note: Passphrases can be encrypted using DES (option **-des**), Triple DES (option **-des**), or IDEA (option **-idea**). If none of these options are supplied, the generated private key is stored *unencrypted*.

Public and private keys can be displayed by using the **openssl rsa** command, as shown in Example 4-2.

Example 4-2 Display RSA key information

```
# openssl rsa -noout -text -in /etc/httpd/ssl.key/server.key
read RSA key
Enter PEM pass phrase:
Private-Key: (1024 bit)
modulus:
    00:9d:1d:82:4d:9b:4e:e4:cb:3b:41:07:c5:e2:7c:...
publicExponent: 65537 (0x10001)
privateExponent:
    00:99:c3:86:82:03:4c:5b:f8:56:3e:57:a8:1f:f7:...
prime1:
    00:d0:1b:cf:20:d9:8f:19:00:6b:11:31:f2:fa:8b:...
prime2:
    00:c1:45:8e:44:61:5f:73:d7:87:f3:91:b8:b1:bc:...
exponent1:
    00:8c:43:22:00:67:01:de:7c:fc:ac:b3:38:ac:0b:...
exponent2:
    00:85:3e:6c:cc:9b:cc:f0:d2:40:ba:1d:56:95:c2:...
coefficient:
    08:b1:36:dc:c8:bb:ce:4b:bd:1d:b1:a0:1f:dd:a8:...
```

Supply the passphrase used to encrypt the private key.

4.2.2 Generating an SSL certificate

To generate an X.509-compliant certificate, use the **openssl req** command, as shown in Example 4-3.

Example 4-3 Generating an SSL certificate

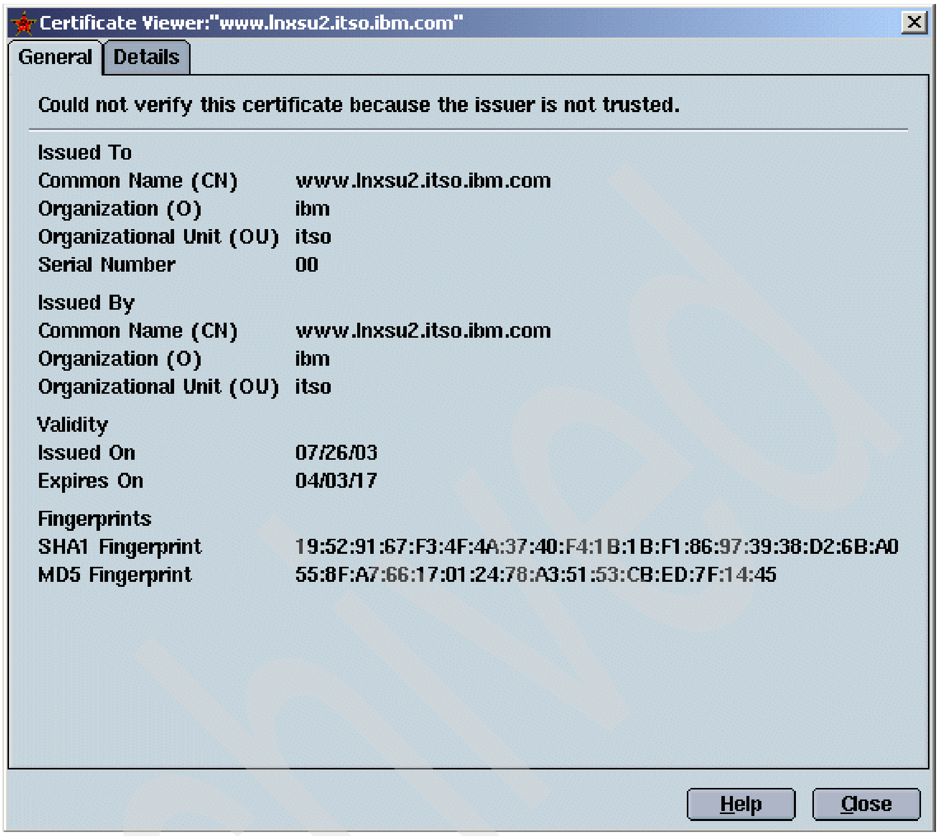
```
# openssl req -new -key /etc/httpd/ssl.key/server.key -x509 -days 5000 \
-out /etc/httpd/ssl.crt/server.crt
Using configuration from /etc/ssl/openssl.cnf
Enter PEM pass phrase:********
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:us
State or Province Name (full name) [Some-State]:nys
Locality Name (eg, city) []:poughkeepsie
Organization Name (eg, company) [Internet Widgits Pty Ltd]:ibm
Organizational Unit Name (eg, section) []:itso
Common Name (eg, YOUR name) []:www.lnxsu2.itso.ibm.com
Email Address []:tot131@itso.ibm.com
```

The generated certificate information can displayed using the **openssl x509** command, as shown in Example 4-4 on page 55.

Example 4-4 Displaying an SSL certificate contents

```
# openssl x509 -noout -text -in /etc/httpd/ssl.crt/server.crt
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 0 (0x0)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: C=us, ST=nys, L=poughkeepsie, ....
        Validity
            Not Before: Jul 26 14:43:22 2003 GMT
            Not After : Apr  3 14:43:22 2017 GMT
        Subject: C=us, ST=nys, L=poughkeepsie, ....
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:9d:1d:82:4d:9b:4e:e4:cb:3b:41:07:c5:e2:7c:...
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                F6:9F:AE:44:F7:3D:79:8E:31:51: ....
            X509v3 Authority Key Identifier:
                keyid:F6:9F:AE:44:F7:3D:79:8E: ....
                DirName:/C=us/ST=nys/L=poughkeepsie ...
            X509v3 Basic Constraints:
                CA:TRUE
    Signature Algorithm: md5WithRSAEncryption
        54:13:46:b0:5f:e7: ....
```

When accessed by a Web browser, the generated certificate is displayed, as shown in Figure 4-1 on page 56.


Certificate Viewer:"www.lnxsu2.itso.ibm.com"

General | Details

Could not verify this certificate because the issuer is not trusted.

Issued To	
Common Name (CN)	www.lnxsu2.itso.ibm.com
Organization (O)	ibm
Organizational Unit (OU)	itso
Serial Number	00
Issued By	
Common Name (CN)	www.lnxsu2.itso.ibm.com
Organization (O)	ibm
Organizational Unit (OU)	itso
Validity	
Issued On	07/26/03
Expires On	04/03/17
Fingerprints	
SHA1 Fingerprint	19:52:91:67:F3:4F:4A:37:40:F4:1B:1B:F1:86:97:39:38:D2:6B:A0
MD5 Fingerprint	55:8F:A7:66:17:01:24:78:A3:51:53:CB:ED:7F:14:45

Figure 4-1 Accessing the certificate from a Web browser

4.2.3 Activating mod_ssl

To activate the mod_ssl module, we add the following lines to the /etc/sysconfig/apache configuration file:

```
HTTPD_SEC_MOD_SSL=yes
HTTPD_START_TIMEOUT=60
```

Setting the HTTP timeout to 60 seconds allows enough time to enter the PEM passphrase when starting Apache. Activate the changes using the **`/sbin/SuSEconfig`** command.

4.2.4 Configuring mod_ssl

Edit the /etc/httpd/httpd.conf file to configure the mod_ssl model. We edited this file in several places. Those changes are summarized in Example 4-5.

Example 4-5 Configuring mod_ssl in the /etc/http/httpd.conf file

```
# Note: the host name is set by SuSEconfig according to the setting of the
# FQHOSTNAME variable in /etc/sysconfig/network/config!
ServerName www.lnxsu2.itso.ibm.com

## SSL Virtual Host Context

<VirtualHost www.lnxsu2.itso.ibm.com:443>

#  General setup for the virtual host
DocumentRoot "/srv/www/htdocs/private"
ServerName www.lnxsu2.itso.ibm.com
ServerAdmin tot131@itso.ibm.com
ErrorLog /var/log/httpd/error_log
TransferLog /var/log/httpd/access_log

#   SSL Engine Switch:
#   Enable/Disable SSL for this virtual host.
SSLEngine on

#   Server Certificate:
SSLCertificateFile /etc/httpd/ssl.crt/server.crt

#   Server Private Key:
SSLCertificateKeyFile /etc/httpd/ssl.key/server.key
```

4.3 Using hardware acceleration with OpenSSL

Encryption is CPU intensive and can lead to longer client response times. To improve the situation, OpenSSL has support for hardware cryptographic acceleration. On zSeries, CPU delegates the encryption process to a specially developed piece of hardware commonly referred to as the crypto engine. Off-loading encryption to hardware improves zSeries performance, as well as providing improved client response times.

To use the zSeries crypto engine:

1. Installing the crypto engine.
2. Creating a crypto device node.
3. Configuring mod_ssl to use the crypto engine.

4.3.1 Installing the crypto engine

The z90crypt90.o module is the device driver for the crypto accelerator. We install the crypto engine device driver, and query the detected crypto accelerators, as shown in Example 4-6.

Example 4-6 Installing and checking the crypto engine

```
# insmod z90crypt
Using /lib/modules/2.4.19-3suse-SMP/kernel/drivers/s390/misc/z90crypt.o
# cat /proc/driver/z90crypt

z90crypt version: 1.1.1
Cryptographic domain: 0
Total device count: 1
PCICA count: 1
PCICC count: 0
requestq count: 0
pendingq count: 0
Total open handles: 0


Mask of online devices: 01 means PCICA, 02 means PCICC
    0000001000000000 0000000000000000 0000000000000000 0000000000000000
    0000000000000000 0000000000000000 0000000000000000 0000000000000000


Mask of waiting work element counts
    0000000000000000 0000000000000000 0000000000000000 0000000000000000
    0000000000000000 0000000000000000 0000000000000000 0000000000000000
```

The device driver detected a PCI bus crypto accelerator (PCICA) hardware interface.

Note: The PCI bus crypto controller (PCICC) is a software encryption module.

4.3.2 Creating a crypto device node

The device node for z90crypt was created as follows:

```
# mknod /dev/z90crypt c 254 0
```

Where:

`c`	Specifies a character device is to be created.
`254`	Specifies the device major node number.
`0`	Specifies the minor node number.

We ensure that the z90crypt is writable, as shown in Example 4-7.

Example 4-7 Making z90crypt writable

```
# ls /dev/z90crypt -l
crw-r--r--    1 root     root     254,   0 Aug  7 16:18 /dev/z90crypt
# chmod 666 /dev/z90crypt
# ls /dev/z90crypt -l
crw-rw-rw-    1 root     root     254,   0 Aug  7 16:18 /dev/z90crypt
```

4.3.3 Configuring mod_ssl to use the crypto engine

To configure mod_ssl to use the crypto engine, add the highlighted line in Example 4-8 to the /etc/httpd/httpd.conf Apache configuration file.

Example 4-8 Insert crypto support in the /etc/httpd/httpd.conf file

```
<IfDefine SSL>
AddType application/x-x509-ca-cert .crt
AddType application/x-pkcs7-crl    .crl
</IfDefine>

<IfModule mod_ssl.c>
SSLCryptoDevice ibmca
```

Restart Apache for the changes to become effective. On startup, we are prompted to provide the PEM passphrase, as shown in Example 4-9.

Example 4-9 Apache with crypto engine enabled

```
# rcapache start
Starting httpd [ Mailman PERL PHP4 Python SSL ]Apache/1.3.26 mod_ssl/2.8.10
(Pass Phrase Dialog)
Some of your private key files are encrypted for security reasons.
In order to read them you have to provide us with the pass phrases.

Server www.lnxsu2.itso.ibm.com:443 (RSA)
Enter pass phrase:

Ok: Pass Phrase Dialog successful.
done
```

4.4 Secure Shell overview

The Secure Shell (also known as SSH) is a secure network access protocol. Traditional network access protocols (Telnet, FTP, and the r-command suite, such as rlogin, rcp, and rsh) must be considered inherently insecure. Unencrypted data (including user IDs and passwords) is transmitted over the network and can be captured by malicious third parties. SSH provides a secure alternative for remote access, file transfer, and remote command execution.

SSH was originally developed by Tatu Ylönen (a researcher at the Helsinki University of Technology) as a method to prevent password-sniffing attacks. His initial implementation (known as SSH1) was released as free software. The SSH-1 protocol was formalized as an Internet Engineering Task Force (IETF) Draft. Subsequent improvements resulted in a new SSH-2 protocol originally implemented as the SSH2 commercial product available from SSH Communications Security, Ltd. (founded by Ylönen in 1995). OpenSSH is an open source implementation of Secure Shell based on the last free version of SSH1. It supports both the SSH-1 and SSH-2 protocols. OpenSSH is provided as part of most distributions of Linux for zSeries (including SLES8 and RHEL3). Details about OpenSSH can be found at their Web site:

http://www.openssh.com

Note: When we use the term SSH in the remainder of this chapter, we are referring specifically to the OpenSSH implementation.

Some of the security benefits provided by SSH include:

- Strong encryption
  Network traffic is encrypted. SSH uses a variety of both symmetric and asymmetric encryption mechanisms:
  - Symmetric encryption is used for bulk data encryption. Supported algorithms include 3DES, Blowfish, and IDEA.
  - Asymmetric encryption is used for both host identification and for exchanging symmetric encryption keys. SSH can use Rivest-Shamir-Adleman (RSA) and Digital Signature Algorithm (DSA) public/private key encryption.
- Digital authentication
  Public keys are used to authenticate SSH hosts.
- Data integrity checking
  Hash functions (CRC-32, MD5, and SHA-1) are used to ensure packets have not been modified in transit.

4.5 Secure network access using SSH

One of the most prevalent uses for SSH is as a secure replacement for Telnet. The command syntax to securely connect to a remote host is:

```
ssh -l user host
```

Where:

user	Specifies the remote user name.
host	Specifies the remote host.

The command may be abbreviated using the syntax:

```
ssh user@host
```

If the remote user name is omitted, the local login name is used by default. To establish a remote SSH, the user is prompted to provide the remote user name password.

Note: User names and passwords are encrypted before being sent across the network. In 4.7, "Authentication without passwords" on page 65, we discuss a mechanism to authenticate SSH clients without providing a password.

4.5.1 Known hosts

SSH employs a known host mechanism to prevent a man-in-the-middle attack. When a client initiates an SSH connection, the server returns its host public key prior to user authentication. The client then checks that key against its list of known server public keys. If the sent key does not match the saved public key for the server, the SSH client assumes that a third party has redirected the connection (to a false host impersonating the intended server) and terminates the connection.

The known host mechanism is an additional security protection (a Telnet session has no protection against man-in-the-middle attacks). Known host public keys are saved in the ~/.ssh/known_hosts file on the SSH client in the format shown in Example 4-10.

Example 4-10 The ~/.ssh/known_hosts file

```
192.168.200.101 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAA...
9.12.9.38 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAIEAt+S...
9.12.9.37 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAIEAmhB...
172.22.10.2 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAIEAt...
192.168.200.102 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAA...
192.168.200.100 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAA...
9.12.6.137 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAIEAwM...
```

Host keys are generated automatically when SSH is installed. Host keys are specific to an SSH server and stored in the /etc/ssh directory. Three types of host keys are supported:

- RSA keys for SSH1
  The ssh_host_key and ssh_host_key.pub files contain the respective RSA private and public keys host identity keys used for SSH1.
- RSA keys for SSH2
  The ssh_host_rsa_key and ssh_host_rsa_key.pub files contain the respective RSA private and public keys host identity keys used for SSH2.
- DSA keys for SSH2
  The ssh_host_dsa_key and ssh_host_dsa_key.pub files contain the respective RSA private and public keys host identity keys used for SSH2.

To create host keys, use the **ssh-keygen** command:

```
ssh-keygen -t type -b 1024 -f /etc/ssh/key-file -N ''
```

Where:

type	Specifies the type of file to create: `rsa1` for SSH1 RSA keys `rsa` for SSH2 RSA keys `dsa` for SSH2 DSA keys

key-file	Specifies the private key file:
	`ssh_host_key` for type `rsa1`
	`ssh_host_rsa_key` for type `rsa`
	`ssh_host_dsa_key` for type `dsa`

The **-b** option specifies a 1024-bit key length is to be used. The **-N** option specifies an empty passphrase (therefore, the key files are stored unencrypted on the file system).

Note: The passphrase is used to encrypt the key files and must be supplied when decrypting a key file. Passphrases are not passed across the network.

Known hosts are added to the list when the client first contacts an unknown host and the user opts to accept that server's public keys, as shown in Example 4-11 on page 63.

Example 4-11 Accepting a server public key

```
# ssh 9.12.6.153
The authenticity of host '9.12.6.153 (9.12.6.153)' can't be established.
RSA key fingerprint is 23:76:e8:be:06:5b:92:5c:bc:f7:3f:1d:8d:b2:f7:ef.
Are you sure you want to continue connecting (yes/no)? yes
12424: Warning: Permanently added '9.12.6.153' (RSA) to the list of known
hosts.
root@9.12.6.153's password:
Last login: Thu Feb 19 13:37:53 2004 from 9.12.9.36
```

If the server public key does not match, SSH terminates, as shown in Example 4-12.

Example 4-12 Known hosts public key mismatch

```
# ssh 9.12.6.153
12502: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
12502: @    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
12502: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
12502: IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
12502: Someone could be eavesdropping on you right now (man-in-the-middle
attack)!
12502: It is also possible that the RSA host key has just been changed.
12502: The fingerprint for the RSA key sent by the remote host is
83:ad:28:b4:9b:61:4c:58:7e:c8:b5:92:c1:8d:39:c2.
12502: Please contact your system administrator.
12502: Add correct host key in /root/.ssh/known_hosts to get rid of this
message.
12502: Offending key in /root/.ssh/known_hosts:1
12502: RSA host key for 9.12.6.153 has changed and you have requested strict
checking.
12502: Host key verification failed.
```

Note: One possible reason for receiving this message is that the keys on the remote host have been recreated using the **`ssh-keygen`** command.

4.5.2 SSH access control

By default, any user account can be accessed using SSH. Access control can be imposed using the /etc/ssh/sshd_config SSH server configuration file. Various types of access control can be applied:

- User account access control
  The `AllowUsers` and `DenyUsers` directives permit or deny access to specific users accounts.

- Host access control
  The `AllowHosts` and `DenyHosts` directives permit or deny access from specific client hosts.
- Group access control
  The `AllowGroups` and `DenyGroups` directives permit or deny access to user accounts based on group membership.
- Root access control
  The `PermitRootLogin` directive controls SSH access to the root user. To deny access to the superuser, specify “`PermitRootLogin No`”.

4.6 File transfer and remote command execution

SSH provides two secure FTP replacements: the **`scp`** command and the **`sftp`** command. SSH also provides the ability to execute commands on a remote host. File transfers and remote command execution are authenticated and encrypted using a secure SSH channel.

4.6.1 The scp command

The **`scp`** command uses SSH to authenticate and securely transfer files over a network. The command syntax is:

```
scp source destination
```

Where *`source`* and *`destination`* are the source and destination file specifications, respectively. File specifications are of the form:

```
user@hostname:path-to-file
```

File specification indicates whether the file is on the local host, or on a remote server:

- If the host name is omitted, the file is assumed to be located on the local host.
- If no user is specified, the user ID executing the **`scp`** command is used.
- The path-to-file is an absolute or relative path (relative to the server on which the file is located). Relative path-to-file for remote servers are relative to the HOME directory of the login user.
- The source path-to-file can include standard shell wildcards (such as “*”, “?”, or “~”).

For example, to copy the xyz.txt file from the /home/user1 directory on remote host 192.168.1.1 to an existing temp directory in the current working directory, use the command:

```
scp user1@192.168.1.1:~/xyz.txt xyz.txt temp
```

Before transferring the file, SSH prompts for the user1 password on the 192.168.1.1 host.

4.6.2 The sftp command

The **`sftp`** command provides an FTP-like alternative to the **`scp`** command. The **`sftp`** command syntax is similar to FTP:

```
sftp user1@192.168.1.1
```

If no user is specified, the user ID executing the command is used.

4.6.3 Remote command execution using SSH

SSH can execute a command on the remote host using the syntax:

```
ssh user@remote-host command
```

For example, to list all files in the user1 home directory on remote host 192.168.1.1, use the command:

```
ssh user1@192.168.1.1 ls -al
```

4.7 Authentication without passwords

By default, SSH is configured to authenticate using passwords provided by the user. Using RSA authentication, it is possible for SSH to authenticate using public key exchange. To configure RSA authentication:

1. Ensure that RSA authentication is enabled on the server.
   Add the `RSAAuthentication yes` directive to the /etc/ssh/sshd_config file.

   Note: The default for SLES8 is to allow RSA authentication.

2. Generate RSA public/private keys to use for authentication.
   Use the command **`ssh-keygen -t rsa`** to generate RSA keys on the server. Execute the command as the user used for SSH access. You are prompted for:
   – The location of the private key file (~/.ssh/id_rsa by default)

   – A passphrase to access the key file

   The public key is stored in the ~/.ssh/id_rsa.pub file.

3. Distribute the public key to clients requiring RSA authentication.
   Copy the server public key file (~/.ssh/id_rsa.pub) to the clients wanting to authenticate without passwords. The contents of the id_rsa.pub file should be appended to the ~/.ssh/authorized_keys file for the user accessing the SSH server.

4.8 Secure tunneling using port forwarding

SSH provides the ability to create a secure TCP tunnel using a feature know as *port forwarding*. With port forwarding, client/server applications communicate over a normally unencrypted network using an SSH tunnel. The tunnel transparently encrypts data exchanged between the application client and server.

Note: Port forwarding is available only to TCP socket protocol applications. Be aware that port forwarding is only possible if an SSH server is running on *both* the application client and application server machines.

4.8.1 Local port forwarding

With local port forwarding, an SSH tunnel is initiated from the client host. SSH then binds to a port on the local host. Connections to this port are proxied over the SSH tunnel to the SSH server running on the remote host (where they are passed to the application server). The concept is illustrated in Figure 4-2 on page 67.

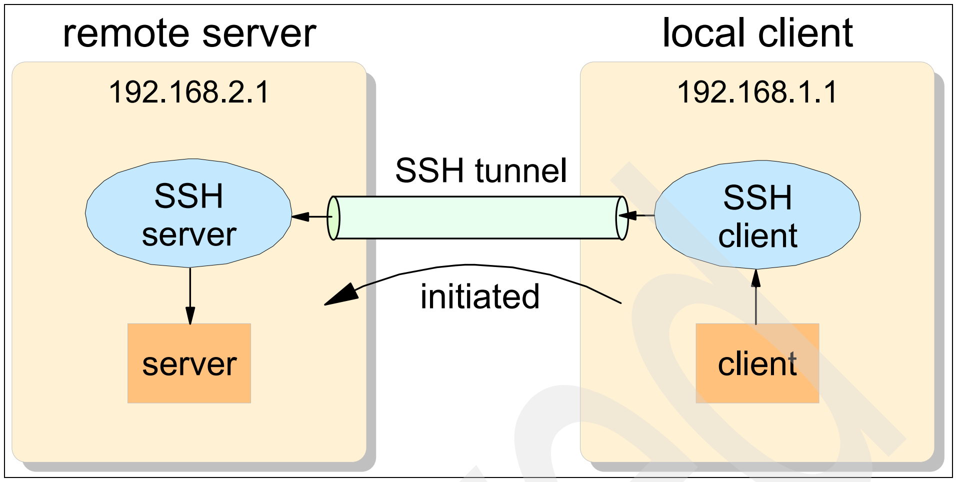


Figure 4-2 Local port forwarding to a remote server

As a simple example, consider an echo server running on the 192.168.2.1 host. The server could be accessed directly from a Telnet client running on the 192.168.1.1 host, but this requires cleartext data to be passed over the network.

Note: An echo server typically runs under inetd control. It listens on TCP port 7, and simply returns the data it is sent. The Telnet client can connect to an echo server by specifying the echo port number. Use the command:

```
telnet 192.168.2.1 7
```

To encrypt network traffic over the SSH tunnel, we use local port forwarding. On the 192.168.1.1 host, we issue the command:

```
ssh -L 4321:localhost:7 192.168.2.1
```

The **-L** option specifies port 4321 on the local client is to be forwarded to port 7 on the remote server. The command logs on to the remote server and tunnel remains open as long as the SSH connection is in place. To access the echo server over the SSH tunnel, we simply connect to the 4321 port on the 192.168.1.1 host (from another command terminal). We illustrate this procedure in Example 4-13 on page 68.

Example 4-13 Illustrating local port forwarding

```
# netstat --listening --numeric --tcp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:1024            0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:4321          0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:111             0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
tcp        0      0 ::1:4321                :::*                    LISTEN
tcp        0      0 :::22                   :::*                    LISTEN
# lsof -n -P -i 4tcp@127.0.0.1:4321
COMMAND PID USER   FD   TYPE DEVICE SIZE NODE NAME
ssh     546 root    5u  IPv4   3318       TCP 127.0.0.1:4321 (LISTEN)
# telnet localhost 4321
Trying ::1...
Connected to localhost.
Escape character is '^]'.
This will be forwarded to a remote echo server over local port 4321
This will be forwarded to a remote echo server over local port 4321
^]
telnet> quit
Connection closed.
```

Note: These commands are executed on the local client (192.168.1.1). Using the **netstat** command, we see a server bound to the loopback interface (localhost) listening on port 4321. The **lsof** command indicates the SSH client is listening on the port.

We connect to the remote echo server using the **telnet** command to port 4321. The response indicates the connection is forwarded to an echo server listening on the remote server (192.168.2.1).

GatewayPorts

In the previous example, we use port 4321 on the localhost interface to forward application client connections (as specified by the **-L 4321:localhost:7** option to the **ssh** command). By default, SSH listens only on the loopback interface for forwarded ports as a security precaution.

Connections to a loopback interface can only be made by clients running on the local host. If a client attempts to connect to a server listening on the loopback interface of a remote host, the connection is refused (typically resulting in a message such as “`Connection refused`”).

Forwarded ports can be opened to remote clients, as shown in Figure 4-3 on page 69.

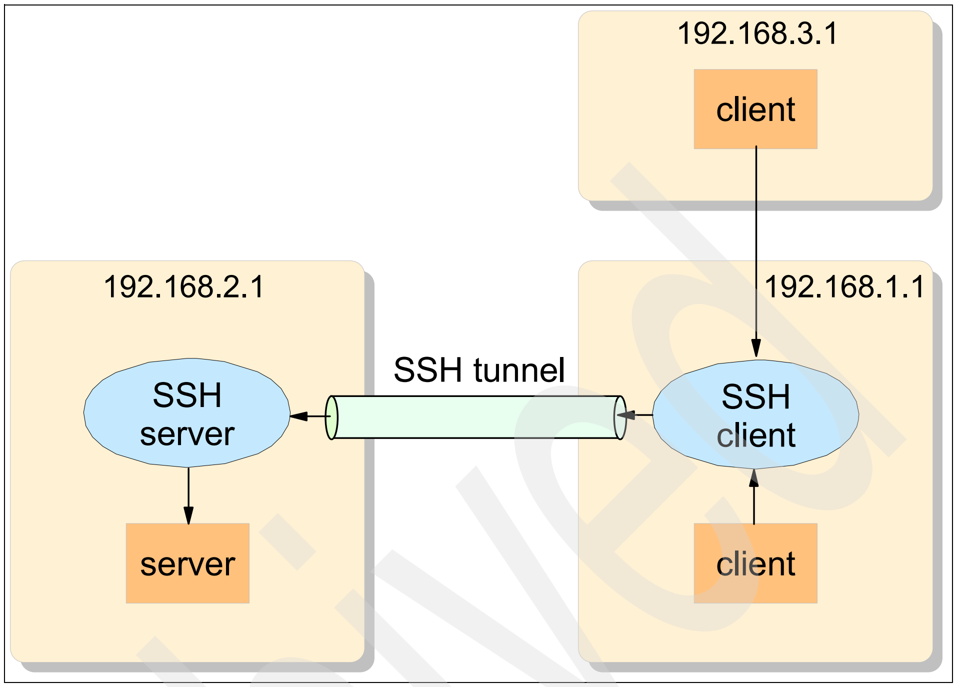


Figure 4-3 Local forwarding acting as a gateway

In this case, the forwarded SSH tunnel acts as a *gateway* for the client running on the 192.168.3.1 host. The default behavior to deny gateway access to SSH forwarded ports can be changed by:

- Adding the `GatewayPorts no` line to the /etc/ssh/ssh_config client configuration file.
- Using the **`-g`** option to the **`ssh`** local forward command, for example:

```
ssh -g -L 4321:localhost:7 192.168.2.1
```

Important: GatewayPorts should be considered a security exposure. Network traffic between the remote client and forwarded port (hosts 192.168.3.1 and 192.168.1.1 in our example) is unencrypted and vulnerable to snooping. In addition, no access control is possible on the forwarded port (allowing an untrusted host access to the server by masquerading as the forwarded client).

4.8.2 Remote port forwarding

In a manner similar to local port forwarding, remote port forwarding opens an SSH tunnel from a local server to a remote client. Figure 4-4 on page 70 illustrates remote port forwarding.

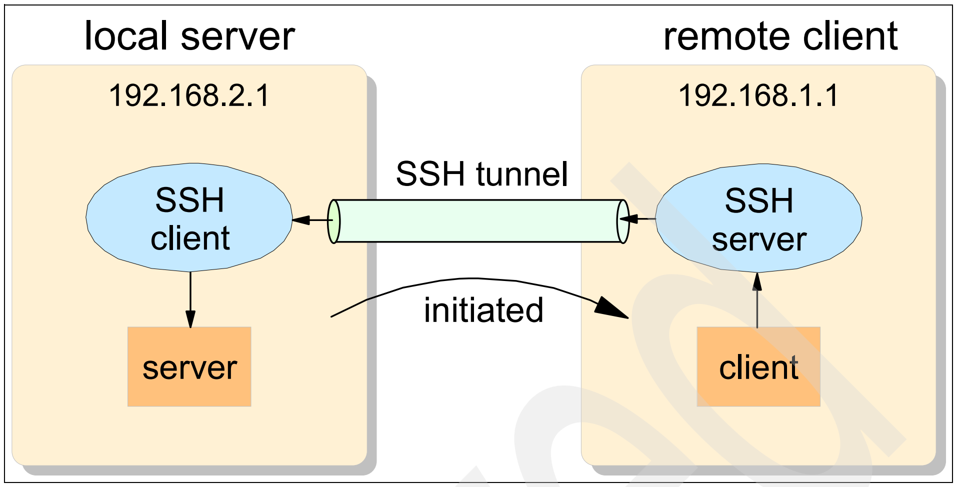


Figure 4-4 Remote port forwarding to a local echo server

In this instance, we initiate the SSH tunnel from the local server (192.168.2.1) to the remote client (192.168.1.1). From the 192.168.2.1 host, issue the command:

```
ssh -R 4321:localhost:7 192.168.1.1
```

The 4321 port now specifies the port to forward on the remote client (192.168.1.1). Clients connecting to port 4321 on the remote client are forwarded to port 7 on the local server over the SSH tunnel.

4.8.3 When to use local or remote forwarding

The distinction between local and remote port forwarding can be characterized as:

- With local port forwarding, the application client connects to an SSH client on the local host. Requests are forwarded to an SSH server connecting to an application server on the other end of the SSH tunnel.
- With remote port forwarding, the application client connects to an SSH server on the local host. Requests are forwarded to an SSH client connecting to an application server on the other end of the SSH tunnel.

In general:

- Use local forwarding when the application client resides on the local host.
- Use remote forwarding when the application client resides on the remote host.

4.8.4 Implications of and options for port forwarding

SSH port forwarding has several implications and effects a Linux administrator should be aware of.

Choosing a forwarded port number

When choosing a port on the local host to forward, remember that superuser authority is required to bind to port numbers less than 1024. Select a port number that is not used by an existing server (and will not be used in the future).

Bypassing a firewall

Port forwarding is a mechanism that allows firewall rules to be bypassed. Consider an example where firewall rules allow SSH to pass (using the default port 22) and deny access to another protocol (IMAP on port 143, for example). With port forwarding, it is possible for a non-privileged user to establish a forwarded connection to an IMAP server through the firewall (and thereby possibly violate a site security policy). There can be circumstances where using port forwarding to bypass a firewall is a legitimate practice. Some control can be imposed on port forwarding by adding `AllowTCPForwarding no` to the /etc/ssh/sshd_config file.

Note: Disallowing TCP forwarding is not sufficient to prevent port forwarding (knowledgeable users can still install their own forwarders). To prevent forwarding, SSH users should be denied shell access (see 4.5.2, “SSH access control” on page 63).

Constructing an SSH tunnel in the background

When creating the SSH tunnel, the **`ssh`** command normally runs a shell on the remote machine in the foreground. It is possible to run the shell in the background (and therefore reuse the terminal session). To create a background tunnel, use the **`-f`** and **`-N`** options. For example:

```
ssh -f -N -L 4321:localhost:7 192.168.2.1
```

Terminating a forwarded port

The SSH tunnel is terminated by logging out from the remote shell. If the tunnel has an active forwarded connection, and you log out from the remote shell, SSH sends itself into the background (maintaining the forwarded connection). In this case, the logout from the remote shell occurs when the forwarded connection completes.

Note: Use the **`kill`** command on the tunnel SSH process to terminate a background SSH tunnel.

One shot forwarding

In "Terminating a forwarded port" on page 71, we use the **-f** option to place the SSH tunnel in the background. Normally, the **ssh** command expects a parameter that specifies a remote command to execute when operating in the background. The **-N** option specifies that no command is to be executed on the remote machine.

> **Tip:** If we omit the **-N** option when creating a background tunnel, we get the following error message:
>
> ```
> # ssh -f -L 4321:localhost:7 192.168.2.1
> 11378: Cannot fork into background without a command to execute.
> ```

We can create a one shot forwarded tunnel by specifying a simple shell command to execute on the remote machine. In Example 4-14, we create a tunnel to an echo server and connect to the server in one command sequence.

Example 4-14 Creating a one shot forwarded tunnel

```
# ssh -f -L 4321:localhost:7 192.168.2.1 sleep 1; telnet localhost 4321
root@9.12.6.153's password:
Trying ::1...
Connected to localhost.
Escape character is '^]'.
this is a one shot tunnel
this is a one shot tunnel
^]
telnet> quit
Connection closed.
```

The **sleep 1** remote command opens the tunnel in the background. We connect immediately to the echo server over the tunnel. Although the remote command completes before the Telnet session terminates, the tunnel remains open as described in "Terminating a forwarded port" on page 71.

4.9 X forwarding

X forwarding is a special type of port forwarding provided by SSH specifically to secure X Windows (the Linux graphical display system). X Windows is a client/server architecture using the X protocol for network communication:

- The X server runs on the machine displaying the graphics.
- The X client is the application creating the graphics.

X protocol is a problem from a security perspective. Network traffic between the server and client is unencrypted and can be monitored by network attackers. X forwarding can help to secure X Windows sessions. With X forwarding, all X protocol network transmissions occur over a secure SSH tunnel.

To enable X forwarding, add `X11Forwarding yes` to the SSH server configuration file (/etc/ssh/sshd_config). With X forwarding enabled, it is important to ensure that `X11UseLocalHost yes` is specified in the /etc/ssh/sshd_config file. This option causes the forwarded port to be bound to the loopback interface (avoiding the exposure discussed in “GatewayPorts” on page 68).

Tip: When X forwarding is enabled and the `X11UseLocalHost yes` option is specified, the DISPLAY environment variable is set to “localhost:10.0” on login. If the `X11UseLocalHost no` option is specified, the DISPLAY variable uses the remote host name on login.

4.9.1 Security considerations with X forwarding

X forwarding should be enabled only after carefully considering the security implications. Although the `X11UseLocalHost` option binds the forwarded port to the loopback address, users can change the DISPLAY environment variable (bypassing the SSH tunnel, and instead exchanging unencrypted data between client and server). If X forwarding is enabled, it is important to have SSH set the DISPLAY variable and ensure that users do not modify this environment variable.

4.10 Securing VNC using port forwarding

Virtual Networking Computing (VNC) is a cross-platform, remote control software package. VNC allows access to an X Windows session from Microsoft® Windows® *without* running an X server on the Windows machine. VNC is distributed under the GNU General Public License (GPL) and is available from RealVNC (the official home of VNC):

`http://www.realvnc.com/`

VNC is a client/server application:

- The VNC server runs on the remote machine.
- A VNC client connects to the server to access the remote machine.

A password is required to connect to a VNC server, and the password is encrypted when passed across the network. However, the VNC protocol is cannot be considered secure; cleartext network traffic is transmitted. Using SSH port forwarding, it is possible to add security to a VNC connection (data is passed over the encrypted SSH tunnel). We outline a procedure to secure a VNC viewer running on Windows when connecting to a Linux guest running on zSeries.

4.10.1 Installing the VNC server

VNC is available as part of the SLES 8 distribution on Linux for zSeries. Both the VNC server and client are part of the VNC RPM package. Use the **`vncserver :0`** command to start the server. The VNC server listens on TCP port 5900 by default (you can verify this using the **`netstat --numeric --listening --tcp`** command). The first time a VNC server is started, you are prompted to provide a password (used when a VNC client attempts to connect to the server).

> **Tip:** The **`:0`** parameter indicates that VNC is to use display 0. Each VNC display uses the next available port after 5900. For example, display 1 listens on port 5901.

4.10.2 Installing the VNC client on Windows

Download and install VNC from RealVNC. To use VNC over a forwarded port, ensure the Windows System Registry contains:

```
HKEY_LOCAL_MACHINE\Software\ORL\WinVNC3\AllowLoopback=1
```

> **Tip:** Use the **`regedit`** command to edit the System Registry.

4.10.3 Installing an SSH server on Windows

Local port forwarding requires SSH on the Windows machine. OpenSSH is provided as part of Cygwin (a Linux-like environment for Windows). Cygwin is available at:

http://www.cygwin.com/

Cygwin provides an install and update utility (setup.exe) to retrieve packages from the Internet. When installing Cygwin, select the OpenSSH package (available in the Net category).

After installing Cygwin, complete the Cygwin configuration. Select **My Computer** → **Properties** → **Advanced** → **Environment Variables**, and then:

- Add the variable “CYGWIN=ntsec tty”.
- Add “C:\cygwin\bin” to the PATH environment variable.

4.10.4 Configuring the Windows SSH server

Configure OpenSSH from a Cygwin console window using the **`ssh-host-config`** command, as shown in Example 4-15.

Example 4-15 Configuring OpenSSH under Cygwin

```
$ ssh-host-config
Generating /etc/ssh_host_key
Generating /etc/ssh_host_rsa_key
Generating /etc/ssh_host_dsa_key
Generating /etc/ssh_config file
Privilege separation is set to yes by default since OpenSSH 3.3.
However, this requires a non-privileged account called 'sshd'.
For more info on privilege separation read
/usr/share/doc/openssh/README.privsep
.

Should privilege separation be used? (yes/no) yes
Generating /etc/sshd_config file

Host configuration finished. Have fun!
```

4.10.5 Creating a local forwarded tunnel from Windows to Linux

Use the **`ssh -L 5900:127.0.0.1:5900 root@192.168.4.1`** command to forward port 5900 on the Windows machine (where the VNC viewer runs) to Linux host 192.168.4.1 (where the VNC server runs).

Note: We found that the loopback interface address (127.0.0.1) *must* be used. If localhost is specified, connecting the VNC viewer to the loopback interface fails with the message:

```
channel 2: open failed: connect failed: Connection refused
```

4.10.6 Connecting to the VNC server over the SSH tunnel

To complete the procedure, start the VNC viewer on the Windows machine (pointing to the server at 127.0.0.1:0). At this point, the connection is forwarded from the Windows machine to the Linux VNC server. All network traffic is transparently encrypted by the SSH tunnel.

5

Implementing virtual private networks using FreeS/WAN

In this chapter, we discuss FreeS/WAN, an open source implementation of virtual private network (VPN) for Linux. Topics include:

- An overview of FreeS/WAN
- Starting FreeS/WAN
- Configuring FreeS/WAN

5.1 An overview of FreeS/WAN

SUSE LINUX Enterprise Server Version 8 (SLES8) for zSeries includes FreeS/WAN, an open source virtual private network (VPN) implementation based on Internet Protocol Security (IPSEC). Details and documentation can be found on the FreeS/WAN home page at:

```
http://www.freeswan.org
```

Note: FreeS/WAN is not included in Red Hat distributions.

IPSEC is a mechanism to authenticate and encrypt IP traffic at the IP level. Operating at this level, IPSEC enables transparent authentication and encryption for IP-based protocol (for example, FTP and Telnet). FreeS/WAN uses standard open protocols for authentication and encryption:

- Internet Key Exchange (IKE) protocol
  IKE provides key exchange and negotiates connection parameters.
- Encapsulating Security Payload (ESP) protocol
  ESP is used to encrypt and authenticate packet payload.

To illustrate VPN configuration using FreeS/WAN, we create the network depicted in Figure 5-1.

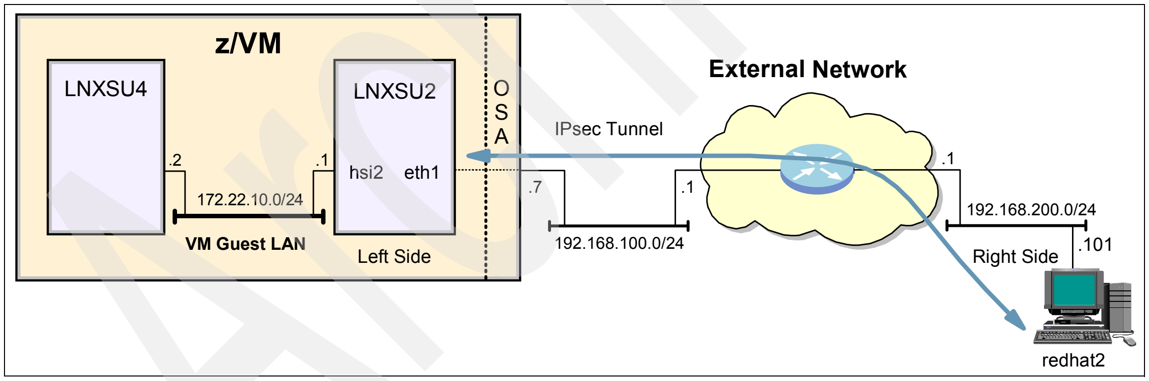


Figure 5-1 VPN implementation using FreeS/WAN on Linux for zSeries

We construct an encrypted IPSEC tunnel between Linux guest LNXSU2 and the redhat2 Linux server. The tunnel traverses an untrusted, external network connected to an OSA-Express adapter. We note:

- The tunnel connects hosts 192.168.100.7 and 192.168.200.101.
  The IPSEC tunnel endpoints are:
  - The LNXSU2 Linux guest at IP address 192.168.100.7 that forms the *left side* of the IPSEC tunnel.
  - The redhat2 Linux host at IP address 192.168.200.101 that forms the *right side* of the IPSEC tunnel.
- The external network simulates a public network (such as the Internet). Normally, traffic on this network is unencrypted and visible to any host connected to the network. However, tunnel traffic between the two endpoints is encrypted.
- The LNXSU2 Linux guest can route packets from the LNXSU4 guest. The LNXSU4 Linux guest forwards traffic destined for the external network to the LNXSU2 Linux guest. IP forwarding is enabled on the LNXSU2 guest, and the guest acts as a packet filtering firewall using the **iptables** command.

5.1.1 Opportunistic encryption

Using FreeS/WAN, it is possible to implement a nonstandard VPN mechanism referred to as *opportunistic encryption*. Typically, two VPN endpoints are configured in advance to exchange public keys for authentication and encryption. With opportunistic encryption, it is possible for VPN endpoints to authenticate using information obtained from the Domain Name System (DNS). This mechanism can reduce administrative overhead in configuring IPSEC.

Note: Although opportunistic encryption has been proposed as an extension to the IPSEC protocol, it is a feature unique to FreeS/WAN. In this redbook, we only discuss VPN scenarios where FreeS/WAN is configured for specific VPN endpoints.

5.2 Starting FreeS/WAN

FreeS/WAN is started from the /etc/rc.d/ipsec init script. To enable FreeS/WAN at startup, issue the command:

```
# chkconfig -s ipsec on
```

Check that the FreeS/WAN components are running using the **ipsec verify** command, as shown in Example 5-1 on page 80.

Example 5-1   Verify IPSEC operation

```
# /etc/rc.d/ipsec status
IPsec running
pluto pid 20576
# ipsec verify
Checking your system to see if IPsec got installed and started correctly
Version check and ipsec on-path                               [OK]
Checking for KLIPS support in kernel                          [OK]
Checking for RSA private key (/etc/ipsec.secrets)             [OK]
Checking that pluto is running                                [OK]
DNS checks.
Looking for TXT in forward map: redhat2                       [MISSING]
Does the machine have at least one non-private address        [OK]
```

Note: The missing TXT record in the forward map for host redhat2 indicates that FreeS/WAN is not configured for opportunistic encryption.

5.3 Configuring FreeS/WAN

FreeS/WAN is configured in the /etc/ipsec.conf file. The /etc/ipsec.conf configuration for the for the topology in Figure 5-1 on page 78 is shown in Example 5-2 on page 81.

Example 5-2 The /etc/ipsec.conf file for the LNXSU2 Linux guest

```
# /etc/ipsec.conf - FreeS/WAN IPsec configuration file
config setup
interfaces="ipsec0=eth1"                                        1
# Debug-logging controls:  "none" for (almost) none, "all" for lots.
klipsdebug=none
plutodebug=none
# Use auto= parameters in conn descriptions to control startup actions.
plutoload=%search
plutostart=%search
# Close down old connection when new one using same ID shows up.
uniqueids=yes
# sample VPN connection
conn sample
# Left security gateway, subnet behind it, next hop toward right.
left=192.168.100.7                                              2
leftsubnet=192.168.100.7/32                                     3
leftnexthop=192.168.100.1                                       4
# Right security gateway, subnet behind it, next hop toward left.
right=192.168.200.101                                           5
rightsubnet=192.168.200.101/32                                  6
rightnexthop=192.168.200.1                                      7
auto=start
authby=rsasig
keyingtries=3
ikelifetime=1h
keylife=1h
rekey=no
auth=esp
pfs=yes
leftrsasigkey=0sAQNq8mSt0na....                                8
rightrsasigkey=0sAQNgQLr15r....                                9
```

Notable configuration parameters include:

1. The `interfaces="ipsec0=eth1"` parameter indicates that IPSEC is logically bound to the eth1 interface using the ipsec0 interface name.
2. The `left` parameter specifies the IP address for the “left” side of the IPSEC tunnel (in this case, the LNXSU2 Linux guest).
3. The `leftsubnet` parameter specifies the left subnet to encrypt. The 32-bit netmask indicates the left end of the tunnel goes to the left host only.
4. The `leftnexthop` parameter specifies the router that forwards packets from the left side to the right side.
5. The `right` parameter specifies the IP address for the “right” side of the IPSEC tunnel (in this case, the redhat2 Linux host).

6. The `rightsubnet` parameter specifies the right subnet to encrypt. Again, the 32-bit netmask indicates host-to-host encryption.
7. The `rightnexthop` parameter specifies the router that forwards packets from the right side to the left side.
8. The `leftrsasigkey` parameter specifies the public key for the left side of the IPSEC tunnel.
9. The `rightrsasigkey` parameter specifies the public key for the right side of the IPSEC tunnel.

Note: The choice of labelling “left” and “right” sides of the tunnel is arbitrary.

The /etc/ipsec.conf file for the redhat2 host is shown in Example 5-3.

Example 5-3 Contents of the /etc/ipsec.conf file for the redhat2 host

```
version 2.0     # conforms to second version of ipsec.conf specification
# basic configuration
config setup
# Debug-logging controls:  "none" for (almost) none, "all" for lots.
klipsdebug=none
plutodebug=none
# sample VPN connection
conn vpn
# Left security gateway, subnet behind it, next hop toward right.
left=192.168.100.7
leftsubnet=192.168.100.7/32
leftnexthop=192.168.100.1
right=192.168.200.101
rightsubnet=192.168.200.101/32
rightnexthop=192.168.200.1
auto=start
authby=rsasig
keyingtries=3
ikelifetime=1h
keylife=1h
rekey=no
auth=esp
pfs=yes
leftrsasigkey=0sAQNq8mSt0n....
rightrsasigkey=0sAQNgQLr15....
```

Note that the “left” and “right” sides of the tunnel are the same as defined on the LNXSU2 host.

5.3.1 Displaying public/private keys

The public key can be displayed using the **ipsec** command:

```
# ipsec showhostkey --right
# RSA 2192 bits   redhat2   Tue Jul 15 07:47:25 2003
rightrsasigkey=0sAQNgQLr15rYk.....
```

To recreate the public/private keys, use the command:

```
 # ipsec newhostkey --output /etc/ipsec.secrets
```

5.3.2 Testing the IPSEC tunnel

Having loaded FreeS/WAN successfully after editing the /etc/ipsec.conf files at both sides, we first checked to see whether IPSEC was operational or not by issuing the highlighted command in Example 5-4.

Example 5-4 Output of the ifconfig command before data transfer through IPSEC

```
# ifconfig
eth1      Link encap:Ethernet  HWaddr 00:06:29:6C:A5:BC
          inet addr:192.168.100.7  Mask:255.255.0.0                        1
          inet6 addr: fe80::6:2900:106c:a5bc/10 Scope:Link
          UP RUNNING MULTICAST  MTU:1492  Metric:1
          RX packets:51109 errors:0 dropped:0 overruns:0 frame:0
          TX packets:63697 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:11509609 (10.9 Mb)  TX bytes:23863443 (22.7 Mb)
          Interrupt:3
ipsec0    Link encap:IPIP Tunnel  HWaddr
          inet addr:192.168.100.7  Mask:255.255.0.0                        2
          UP RUNNING NOARP  MTU:16260  Metric:1                            3
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0               4
          TX packets:5 errors:0 dropped:0 overruns:0 carrier:0             5
          collisions:0 txqueuelen:10
          RX bytes:0 (0.0 b)  TX bytes:1386 (1.3 Kb)
```

The significance of the highlighted numbers in Example 5-4 are as follows:

1. This is the IP address of the Ethernet interface.
2. This is the IP address of the ipsec0 interface that is (logically) bounded to the Ethernet interface.
3. This indicates that it is UP, which is good news.
4. Nothing has been received on the IPSEC tunnel from the other side, which on surface is in contradiction with point three above. However, this is good as far as IKE phase 1 is concerned; see the next item.

5. This means that five packets were sent to the other side to establish IKE phase 1.

To establish the IPSEC tunnel, one side needs to send some data to the other side. For example, we issued three pings from “left” to “right,” and then we checked the interfaces on the “left” again for any changes, as shown in Example 5-5.

Example 5-5 Output of the ifconfig command after data transfer through IPSEC

```
# ping 192.168.200.101
PING 192.168.200.101 (192.168.200.101) from 192.168.100.7 : 56(84) bytes of
data.
64 bytes from 192.168.200.101: icmp_seq=1 ttl=64 time=1.10 ms
64 bytes from 192.168.200.101: icmp_seq=2 ttl=64 time=1.04 ms
64 bytes from 192.168.200.101: icmp_seq=3 ttl=64 time=0.888 ms

--- 192.168.200.101 ping statistics ---
3 packets transmitted, 3 received, 0% loss, time 2014ms
rtt min/avg/max/mdev = 0.888/1.010/1.100/0.096 ms
# ifconfig
eth1      Link encap:Ethernet  HWaddr 00:06:29:6C:A5:BC
          inet addr:192.168.100.7  Mask:255.255.0.0
          inet6 addr: fe80::6:2900:106c:a5bc/10 Scope:Link
          UP RUNNING MULTICAST  MTU:1492  Metric:1
          RX packets:51194 errors:0 dropped:0 overruns:0 frame:0
          TX packets:63818 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:11518628 (10.9 Mb)  TX bytes:23876001 (22.7 Mb)
          Interrupt:3
ipsec0    Link encap:IPIP Tunnel  HWaddr
          inet addr:192.168.100.7  Mask:255.255.0.0
          UP RUNNING NOARP  MTU:16260  Metric:1
          RX packets:3 errors:0 dropped:0 overruns:0 frame:0                 1
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0               2
          collisions:0 txqueuelen:10
          RX bytes:252 (252.0 b)  TX bytes:1836 (1.7 Kb)
```

As can be seen in 1 and 2 in Example 5-5, packets are sent and received; therefore, there is data communication through the “ipsec” interface. Normally, this is good enough evidence that encryption is taking place.

StoneGate firewall

In this chapter, we introduce StoneGate, a pre-hardened, commercial firewall and virtual private network (VPN) solution available on Linux for zSeries. We also discuss the underlying security principles of firewalls in general, including what firewalls are, which different types of firewalls are available, how they are used, what they are capable of, and their potential weaknesses.

We discuss:

- The role of firewalls
- Firewall technologies
- StoneGate firewall components
- StoneGate on Linux for zSeries
- StoneGate installation

6.1 The role of firewalls

Firewalls are a cornerstone in a corporate defense strategy designed to eliminate misuse of confidential data and resources. A firewall is one (albeit important) element in an overall security policy. Its role is to implement and enforce the network security policy. Firewalls regulate communication between networks with different security levels. Their purpose is to control the traffic passing from one network to another, denying access to undesired or potentially harmful packets and connections.

The principle of access control is ideally expressed as *whatever is not expressly permitted is denied*. By default, no one should be permitted entry to a protected network. This means any traffic allowed into a network must be allowed through a specific rule. Typically, internal network clients are granted more unrestricted access to external networks, but outbound connections also need to be controlled.

Some advantages that firewalls can provide include:

- Firewalls protect a corporate intranet from undesired traffic.
  This including everything from malicious attackers to unsolicited e-mail (spam).
- Firewalls secure sensitive corporate resources within a LAN.
  This insulates departments such as Research and Development or Human Resources from other company departments, limiting the access within any one area to authorized users only.
- Firewalls concentrate network security policies at a single point.
  Network security can focus on the firewall choke point. When policies or administrators change, configurations need only be changed in one place. However, the firewall should not become a *single point of failure*.

Firewalls can also be used for other purposes, such as monitoring network traffic and the use of network resources, authenticating users, and implementing virtual private networks (VPN).

6.2 Firewall technologies

The firewall market offers a wide range of both software-based and hardware-based firewall solutions. Firewalls are usually categorized by the way they handle network traffic. In this section, we give an overview of existing firewall technologies. We also introduce a new type of firewall: StoneGate.

Traditionally, firewalls can be divided into three main types:

- Packet filtering firewalls
- Application-level proxy firewalls
- Stateful inspection firewalls

Each type has its benefits and disadvantages. We briefly compare each technology, pointing out their strong points and drawbacks.

6.2.1 Packet filtering firewalls

Packet filtering firewalls allow or deny access based on header information in the packets they receive. Packet filtering firewalls examine each packet and can be implemented by most common routing devices.

Packet filters typically contain *access control lists* (ACLs) to monitor the following header data:

- Source IP address
- Destination IP address
- Source port
- Destination port
- ICMP message type
- Protocol
- Packet size
- Various header flags

Combining these parameters, more complex policies can be enforced by the packet filter. Packet filters tend to have high performance, because inspection involves very simple parameters at the network layer of the TCP/IP stack. They typically only inspect up to the network layer, as show in Figure 6-1 on page 88.

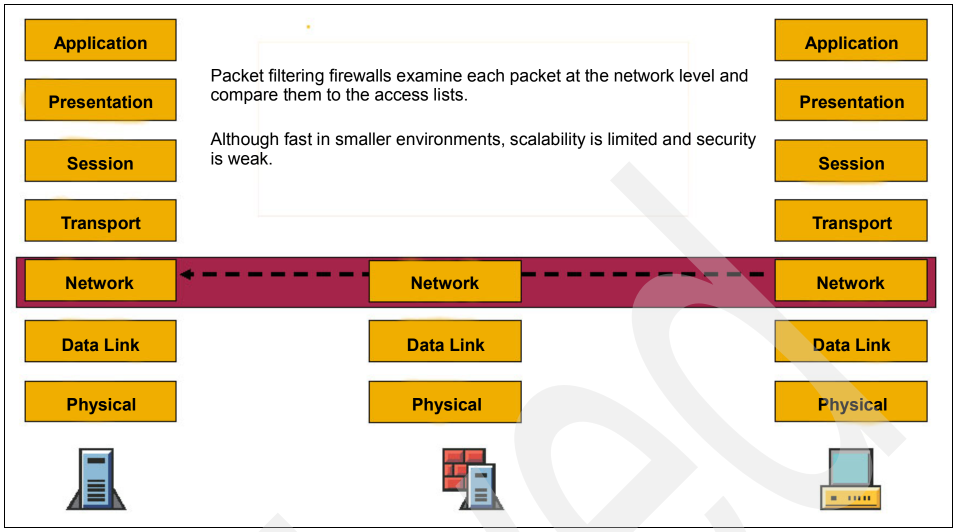

Application
Presentation
Session
Transport
Network
Data Link
Physical
Packet filtering firewalls examine each packet at the network level and compare them to the access lists.
Although fast in smaller environments, scalability is limited and security is weak.
Network
Data Link
Physical
Application
Presentation
Session
Transport
Network
Data Link
Physical

Figure 6-1 Packet filtering model

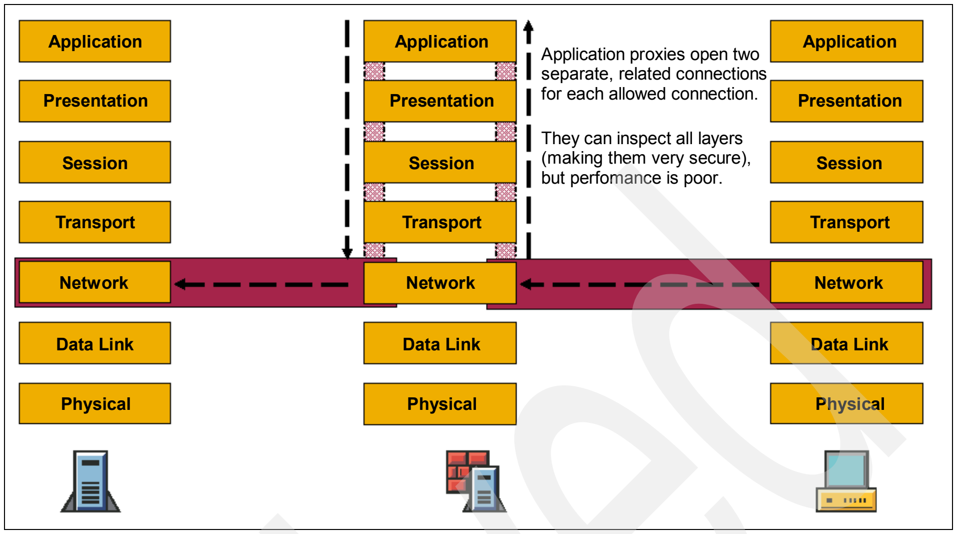
In more complex environments, performance drops significantly as each packet of every connection is checked against the access control rules.

Packet filtering is neither very flexible nor generally very secure because the network layer lacks the context of each packet. This type of filtering can be easily fooled with techniques such as fragmented packets, or bogus or invalid IP address information. Packet filters cannot protect against malicious contents in higher levels of the protocol stack or examine the actual data portion. They are often used in combination with application-level proxies. Packet filtering is commonly used also in routers at the network perimeters.

6.2.2 Proxy firewalls

Proxy firewalls are firewalls running application proxy servers. The firewall server establishes a second, different connection to the destination network on behalf of the host from the source network. Packets are passed if they meet the security policy criteria. Proxy firewalls mediate communications between two different devices located on different networks.

This type of firewall is fully application-aware, and therefore very secure. However, there is a performance trade-off due to the additional overhead required to maintain separate connections and to inspect packets up to the application layer. The proxy firewall is depicted in Figure 6-2 on page 89.



Application
Presentation
Session
Transport
Network
Data Link
Physical
Application proxies open two separate, related connections for each allowed connection.
They can inspect all layers (making them very secure), but perfomance is poor.

Figure 6-2 Proxy firewall model

Proxy firewalls can cache information such as HTML pages in order to gain some performance, but the following drawbacks remain:

- Application layer checking and duplicate connections can affect performance. Traffic inspection takes place at the highest layer of the TCP/IP stack. The necessity of inspecting every layer before returning back down to the physical layer can cause bottlenecks in busy networks.
- The number of services used by a proxy firewall is an issue. Because every service needs its own proxy, the proxy list must be continually updated as new applications and services are added.

6.2.3 Stateful inspection firewalls

Stateful inspection technology was developed to overcome the limitations of packet filtering firewalls. Stateful inspection firewalls use additional criteria (such as historical data about the connection) in determining whether to allow or deny access. Established connections are tracked in *dynamic state tables* to ensure they comply with security policies.

By applying connection status and context information to established connections and packets, access denial can occur before further inspection at a higher level in the protocol stack. This can increase performance. Because stateful inspection understands connection context (returning packets can be correlated to an established connection), previously determined secure connections can be allowed without further examination. This is especially important for services such as Telnet and FTP.

Stateful inspection operates beneath the network layer (in practice, inspection occurs between the data link and network layers). Stateful inspection firewalls have a limited capability to inspect data at the application layer. State tables are maintained to every connection protocol, whether it is valid or not. The stateful inspection model is illustrated in Figure 6-3.

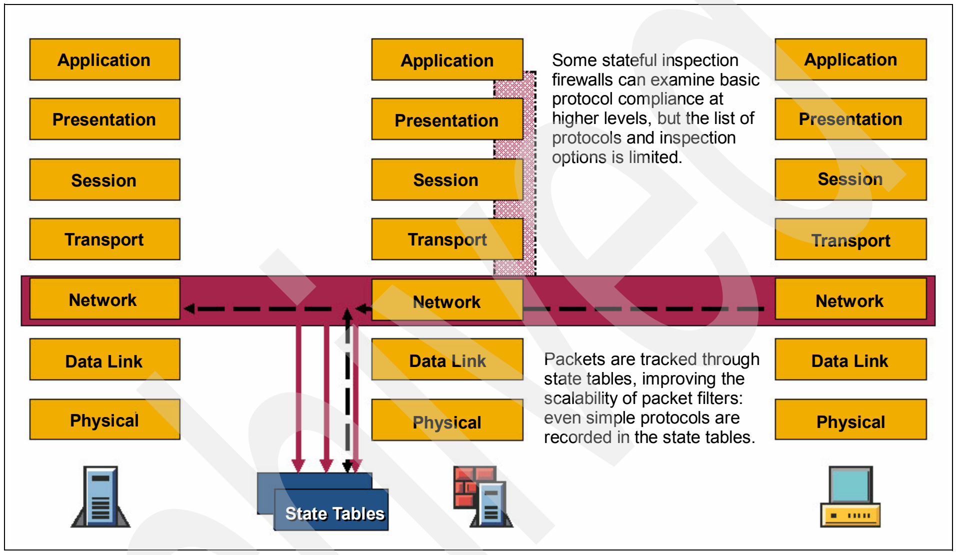

Application
Presentation
Session
Transport
Network
Data Link
Physical
Application
Presentation
Session
Transport
Network
Data Link
Physical
Some stateful inspection firewalls can examine basic protocol compliance at higher levels, but the list of protocols and inspection options is limited.
Application
Presentation
Session
Transport
Network
Data Link
Physical
Packets are tracked through state tables, improving the scalability of packet filters: even simple protocols are recorded in the state tables.
State Tables

Figure 6-3 Stateful inspection model

6.2.4 StoneGate and multi-layer inspection

With StoneGate, the Stonesoft Corporation introduces a new firewall technology called *multi-layer inspection* depicted in Figure 6-4 on page 91. Like stateful inspection, StoneGate uses state tables to track connections and judge whether a packet is a part of an established connection. However, it also features application-layer inspection by implementing specific *protocol agents*, when necessary, for enhanced security.

StoneGate can inspect data all the way up to the application layer to decide whether a packet should be granted access or not. Moreover, StoneGate can also act as a packet filter for types of connections that do not require the security considerations of stateful inspection.

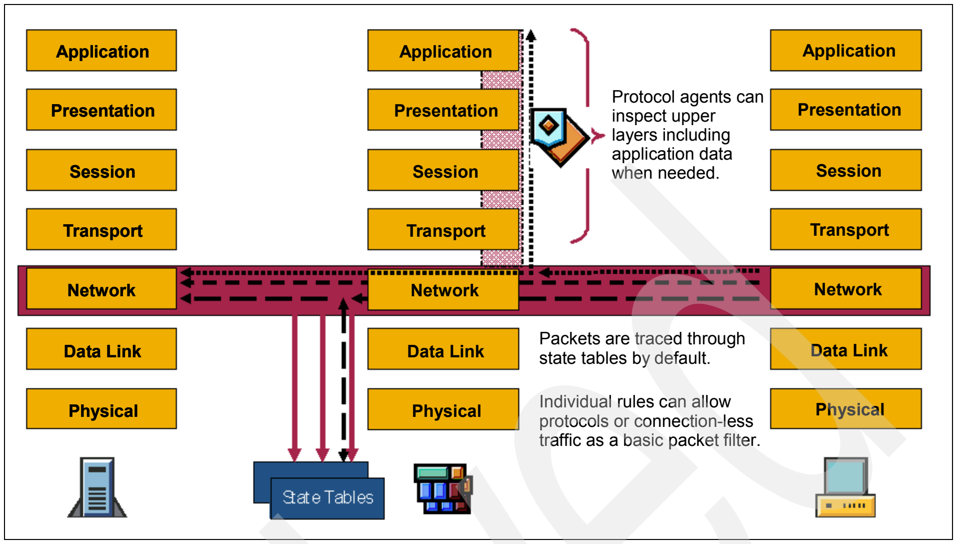

Application
Presentation
Session
Transport
Network
Data Link
Physical
Application
Presentation
Session
Transport
Network
Data Link
Physical
Protocol agents can inspect upper layers including application data when needed.
Application
Presentation
Session
Transport
Network
Data Link
Physical
Packets are traced through state tables by default.
Individual rules can allow protocols or connection-less traffic as a basic packet filter.
State Tables

Figure 6-4   StoneGate multi-layer inspection model

The state of connections provides valuable information for assessing incoming packets. Any packet must be either accepted directly by the rule base or be a part of a previously accepted connection or of a related connection. Whenever packets arrive, the firewall checks them against active connections before proceeding through the rules of the security policy. If a connection has a registered state, all the packets following the opening packet can pass the firewall securely without having to traverse the rulebase.

By default, most rules in StoneGate security policies implement stateful inspection methods. The administrator can configure rules with simple packet filtering for certain types of traffic. For example, SNMP traps from server systems and network devices can pass through the firewall to a management network on the basis of simple packet filtering in case there is no need to enforce a more strict inspection. This flexibility enhances the firewall performance.

In addition, application-level security can be applied to specific rules in the security policy when needed. Multi-layer inspection is capable of providing this type of security without the performance degradation of conventional proxy firewalls. StoneGate can implement application-level inspection without the need to handle two separate connections. This is achieved with the components called protocol agents that can be assigned to certain types of traffic.

Protocol agents are also used to:

- Handle complex connections (for example, connections to an FTP server)
- Redirect traffic to content inspection servers

- Enforce protocol standards
- Modify data payload if necessary

The FTP protocol agent can inspect the control connection and allow only packets containing valid FTP commands. It can be configured to redirect traffic to a content inspection server (CIS) for content screening, or to modify IP addresses in the payload if network address translation is required.

Note: For more details about protocol agents, consult the *StoneGate Administrator's Reference*, available to registered customers at the Stonesoft Web site:

http://www.stonesoft.com

Multi-layer inspection combines application-layer inspection, stateful inspection, and packet filtering technologies for added security without affecting system performance.

6.2.5 Firewall functions

A firewall can have several different functions on a network. Although the main function is to control network access, firewalls can be configured for other basic network security tasks, as well as for complex monitoring and filtering functions.

Access control

The primary task of any firewall is to allow network access to only authorized connections. Access control is enforced in access rules that are combined into rule bases. The rule bases reflect the corporate network security policy.

Monitoring and logging

Firewalls can measure and monitor traffic load and attributes. A very important firewall feature is the ability to log monitored traffic. Properly recorded log data can be used to detect intruders and establish evidence to use against attackers. That kind of forensic evidence might prove to be invaluable in case malicious hacking leads to lawsuits.

More commonly, logging is used to track the use of network resources, building a case for required services or hardware. Logging also helps administrators detect and troubleshoot network mis-configurations or failures.

Network address translation

Network address translation (NAT) is a feature that enables the firewall to modify the IP headers of packets it forwards. It was originally created to alleviate the problem of the rapidly diminishing IP address space. By changing the network address of the originating (internal) network, your network gains an added side-benefit; the private IP addresses of hosts and the structure of an internal network can be concealed by a firewall. In fact, NAT enables you to hide your entire network behind even a single public IP address.

As handy as NAT can be in terms of scarce public IP addresses and security, it is important to understand that NAT is *not* primarily a security feature. It is simply a method of modifying packets that lends itself to security applications.

Authentication

Firewalls are often used to authenticate users accessing network resources from other locations. The various authentication methods ensure that the users trying to connect really are who they claim to be.

These methods can involve a third-party authentication service based on standard protocols, such as RADIUS or TACACS+, but it can also be based on the originating IP address or other parameters, such as user names and passwords.

Virtual private networks

Virtual private networks (VPNs) conceal and encrypt traffic between endpoints to establish a virtual, secure tunnel through an insecure, typically public, network. Firewalls are used at the tunnel endpoints as *security gateways* to encrypt and decrypt data passing between them, creating a *site-to-site VPN*. VPNs can also be established between a client machine, such as a remote laptop, and the firewall. Figure 6-5 illustrates a simple, site-to-site VPN.

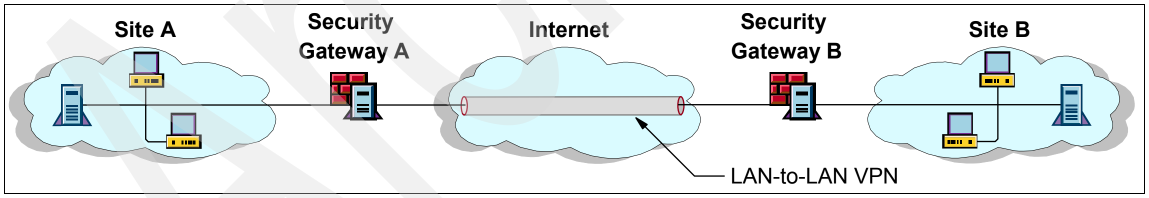


Figure 6-5 Simple VPN example

When a packet leaves Site A, destined for Site B, and reaches gateway A, it is encrypted by the firewall. After it reaches the Site B gateway, the packet is decrypted and re-shaped into cleartext form. When concealing the traffic this way, it is possible for two different sites that might be on other sides of the world from each other to share resources and communicate with each other safely over the Internet.

Content screening

Although not usually used as the primary tool for virus detection, firewalls can be used in combination with content inspection servers (CISs). The data portion of incoming packets can be examined for potentially malicious content. Viruses or hazardous content can be discarded before packets enter the internal network.

For example, incoming SMTP e-mail traffic could be forwarded from the firewall to the CIS for virus and content checking. After any suspicious content is removed, scrubbed packets are returned to the firewall for routing to their final destination.

Content inspection can also flow in the opposite direction. Incoming HTTP traffic can be sent from the firewall to the CIS. The source Uniform Resource Locator (URL) can be examined to deny access to inappropriate sites. Figure 6-6 illustrates firewall interaction with a CIS.

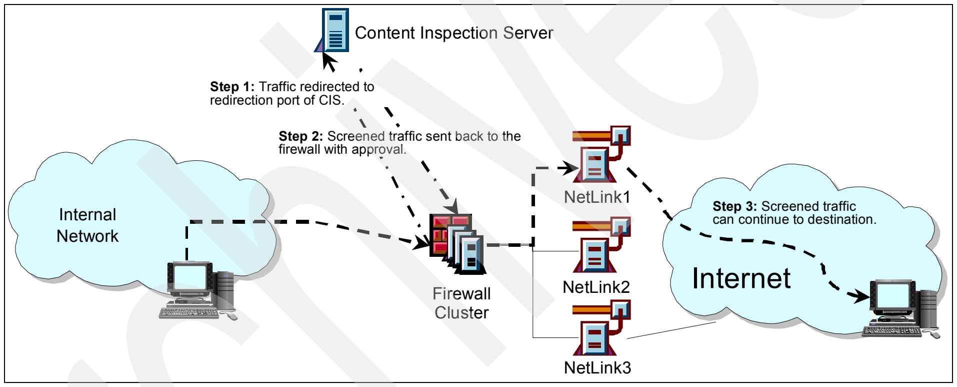

Content Inspection Server
Step 1: Traffic redirected to redirection port of CIS.
Step 2: Screened traffic sent back to the firewall with approval.
Internal Network
Firewall Cluster
NetLink1
NetLink2
NetLink3
Step 3: Screened traffic can continue to destination.
Internet

Figure 6-6 Content screening with CIS

6.2.6 Requirements for modern firewalls

As the volume and the importance of Internet traffic grows, it becomes increasingly essential for firewalls to meet specific requirements.

High availability

Advanced clustering technologies prevent firewalls from being either bottlenecks or single points of failure. Firewalls are clustered when multiple firewall nodes function as a single, virtual entity and enforce identical security policies. Each node contributes to the total throughput, providing a fault tolerant, reliable firewall. High availability means that traffic from a failed node can be transparently switched to other nodes. Node maintenance can occur without disturbing the firewall. Availability of network services is crucial for both user satisfaction and for network security. Clustering can be implemented either with specific additional hardware, with add-on software, or with a built-in clustering capability. The firewall instance depicted in Figure 6-7 acts as a single point of failure.

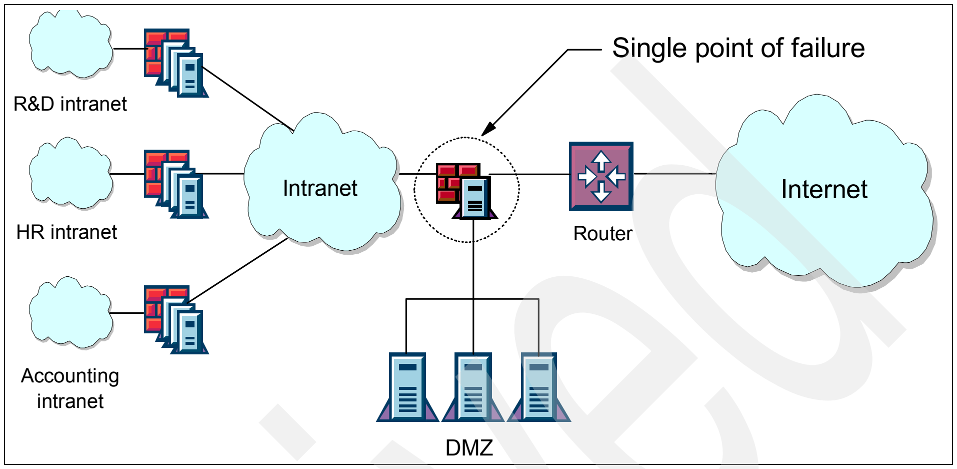


Figure 6-7 Standard network firewalls as single point of failure

By introducing multiple firewall instances, as shown in Figure 6-8, the single point of failure is eliminated.

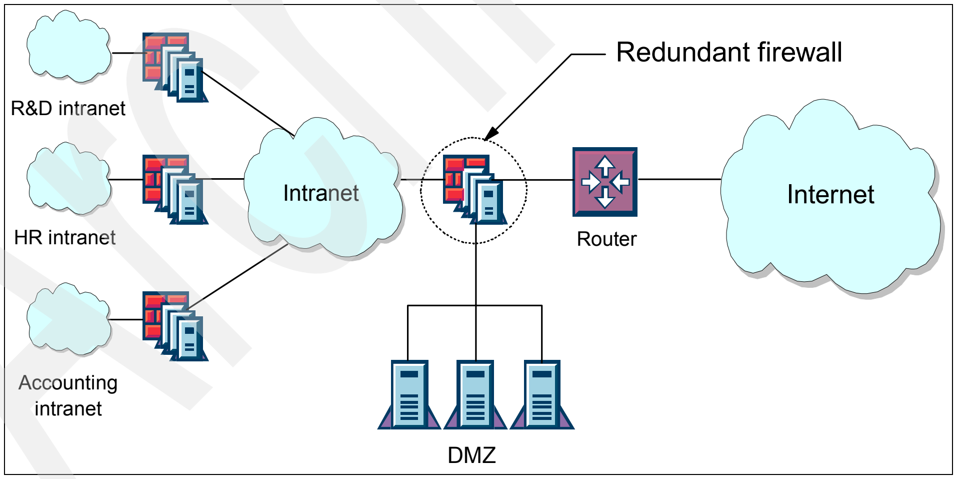


Figure 6-8 Eliminating firewalls as single point of failure through clustering

Scalability

As traffic volume increases, congestion can disrupts network traffic. By definition, the firewall is a choke point through which all traffic should pass. It is crucial to ensure that the firewall does not become the limiting factor for network performance. Clustering or adding hot-standby firewalls nodes adds flexibly as traffic volumes grow. Maintenance can be performed on a live production system with no down time.

Throughput

As gigabit networks become more common, firewalls are required to cope with increasing traffic. Clustering firewall nodes contributes to better throughput.

Management

Geographically widespread, multinational, and smaller networks all benefit from centrally managed firewall security policies. Corporate security policies often coexist with site-specific rules to provide the required degree of granularity in the implementation of network security policy. The implementation of corporate-wide, complicated security policies requires a great deal of flexibility from the firewall management system.

Centralized management minimizes the possibility of human error. To avoid unintentional confusion or harm, access to firewall configuration and rule bases should be carefully planned according to the level of authority and expertise of the administrators. Remote installation and configuration is an important feature for distributed networks.

6.2.7 Firewall weaknesses

Knowing the weaknesses of firewalls is essential for a balanced approach to creating an effective security policy. You need to have a good picture of the whole security framework in order to determine other security measures to implement.

Lack of administration

Complex network environments increased the demand for administrative skills. Firewalls cannot provide effective security without careful attention and maintenance. A firewall must be thoughtfully installed and configured. Security policies need periodic evaluation and regular updates. Too often, firewall administration is neglected until an attack or security incident.

Internal attacks

Firewalls, content inspection servers, and other devices examining packets at the network perimeter are ineffective against *internal* attacks. By some estimates, around 60% of all network attacks are launched from within a corporate network. These attacks are much more difficult to defend against and require different approaches than firewalls or other perimeter defenses. Implementation of the corporate security policy should address these issues (for example, using employee security and host-based virus protection). Properly deployed Intrusion Detection Systems (IDS) can offer a defense against malicious traffic that cannot be blocked with firewalls.

Note: Placing the firewall in the mainframe eliminates the chance of intrusion from inside the corporate intranet. Firewalls external to zSeries can be compromised from the intranet.

6.3 StoneGate firewall components

In this section, we describe the StoneGate architecture and look at various deployment schemes. The design strategy for StoneGate is a distributed architecture illustrated in Figure 6-9.

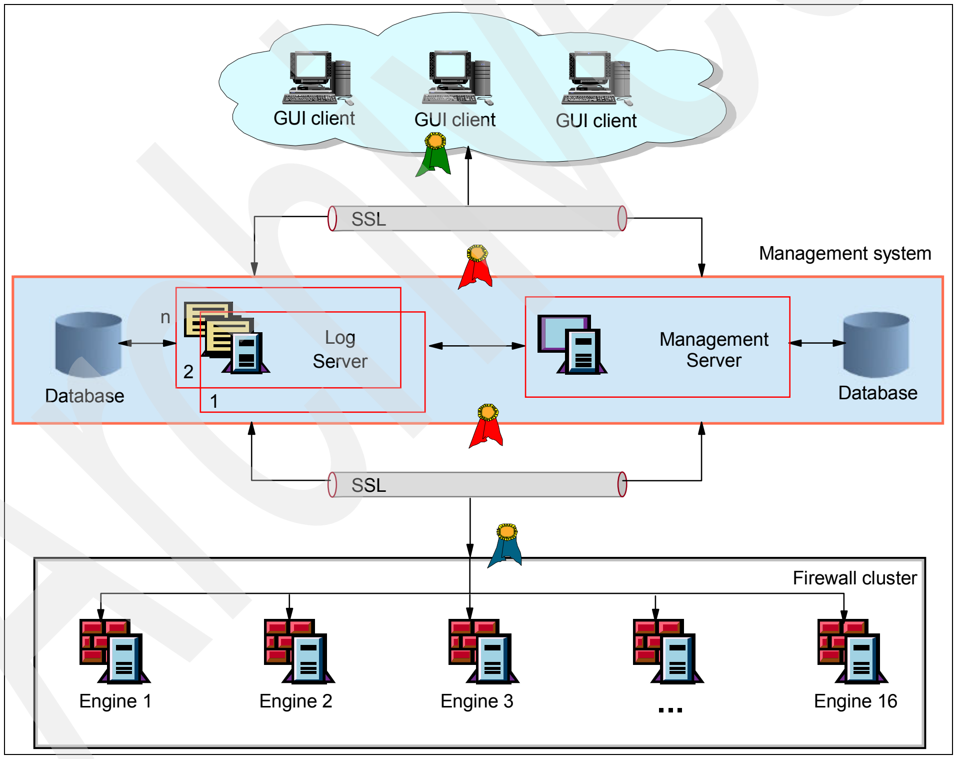


Figure 6-9 StoneGate architecture overview

StoneGate consists of three main components:

- The graphical user interface (GUI)

- The management system
- The firewall engines

6.3.1 StoneGate GUI

The GUI (Administration Client) is the interface used to configure and administer all aspects of StoneGate. It is a Java™-based program that allows administrators to:

- Configure firewall nodes.
- Describe protected networks and systems by defining hosts, servers, services, VPNs, firewalls and firewall clusters, routers, and other network devices.
- Design the policies to protect those systems and networks.
- Administer users and authentication services.
- Manage log data.
- Monitor firewalls and clusters.

The GUI communicates with the various components of the management system, presenting performance data, log data, configuration information, and policies in a user-friendly format. The GUI can be deployed anywhere in the network, and multiple instances be run at once. For example, an administrator can use the GUI to modify the properties of a local network in a branch office, while the senior administrator in the corporate headquarters is updating global policies or objects. Conflicts between simultaneous users are resolved through locking mechanisms in the database engine by the Management Server.

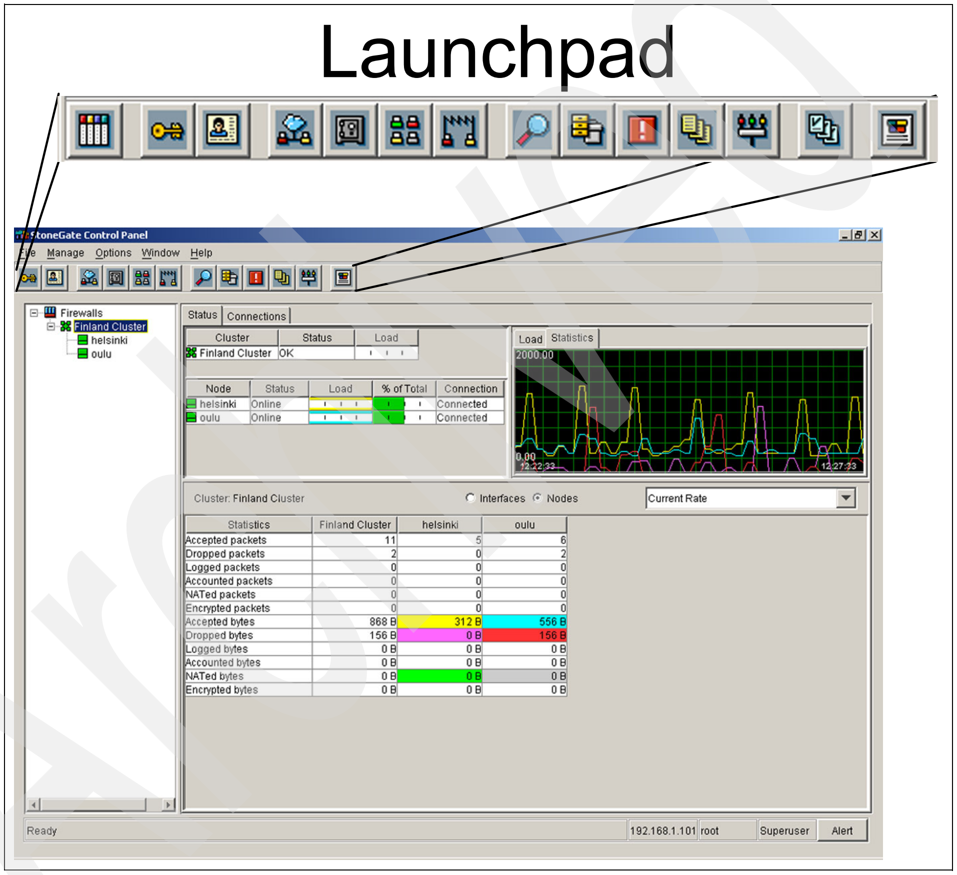
A single GUI can be used to control all the firewalls and clusters under the same management system. For example, an updated security policy can be installed simultaneously on all firewall systems in the corporate network.

Note: The GUI provides support for *managed service providers* (MSP) that offer network services to several customers. The GUI filtering capabilities each single customer to be independently managed.

The main view of the GUI (called the *Control Panel*) contains the *Launchpad* toolbar from which the various manager applications can be started. Each manager is an application component used to perform a particular set of related tasks. For example, the Security Policy Manager is used to create, modify, and delete security policies or rule bases; the Network Element Manager enables the creation, modification, and deletion of various network elements, such as hosts, routers, network links, and firewall clusters. The Control Panel and Launchpad are shown in Figure 6-10 on page 99.

In addition to the Launchpad, the StoneGate Control Panel presents the administrator with statistical information and graphical representations on the current state of the firewall clusters and engines. This information is processed by the monitoring system of the Management Server, as described in 6.3.2, “Management system” on page 100.

Note: StoneGate also uses a number of command line tools documented in the *StoneGate Administrator’s Reference*.



Cluster	Status	Load
Finland Cluster	OK	

Node	Status	Load	% of Total	Connection
helsinki	Online			Connected
oulu	Online			Connected

Cluster: Finland Cluster ( ) Interfaces (●) Nodes Current Rate

Statistics	Finland Cluster	helsinki	oulu
Accepted packets	11	5	6
Dropped packets	2	0	2
Logged packets	0	0	0
Accounted packets	0	0	0
NATed packets	0	0	0
Encrypted packets	0	0	0
Accepted bytes	868 B	312 B	556 B
Dropped bytes	156 B	0 B	156 B
Logged bytes	0 B	0 B	0 B
Accounted bytes	0 B	0 B	0 B
NATed bytes	0 B	0 B	0 B
Encrypted bytes	0 B	0 B	0 B

Figure 6-10 StoneGate Control Panel and Lauchpad

.

Note: For security reasons, the GUI does not communicate directly with the firewall engines. All changes to the firewall nodes are handled through the management system components.

6.3.2 Management system

The management system is composed of the machines that run the management components. As illustrated in Figure 6-11, the management system consists of four components:

- ► Management Server
- ► Log Server
- ► User directory service
- ► Element databases

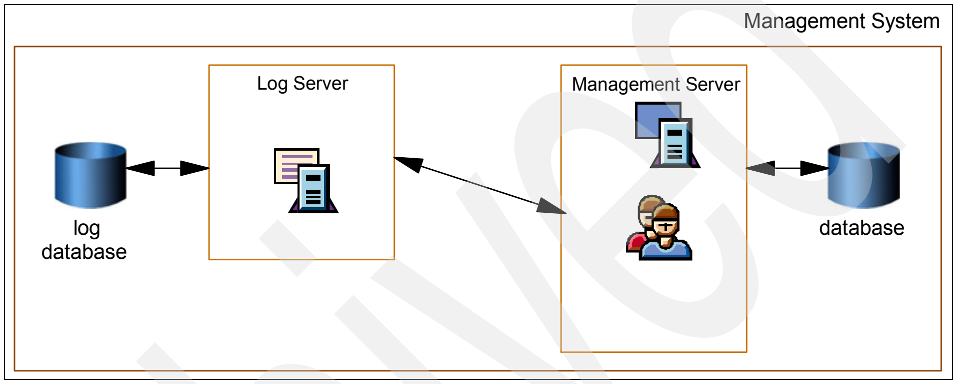


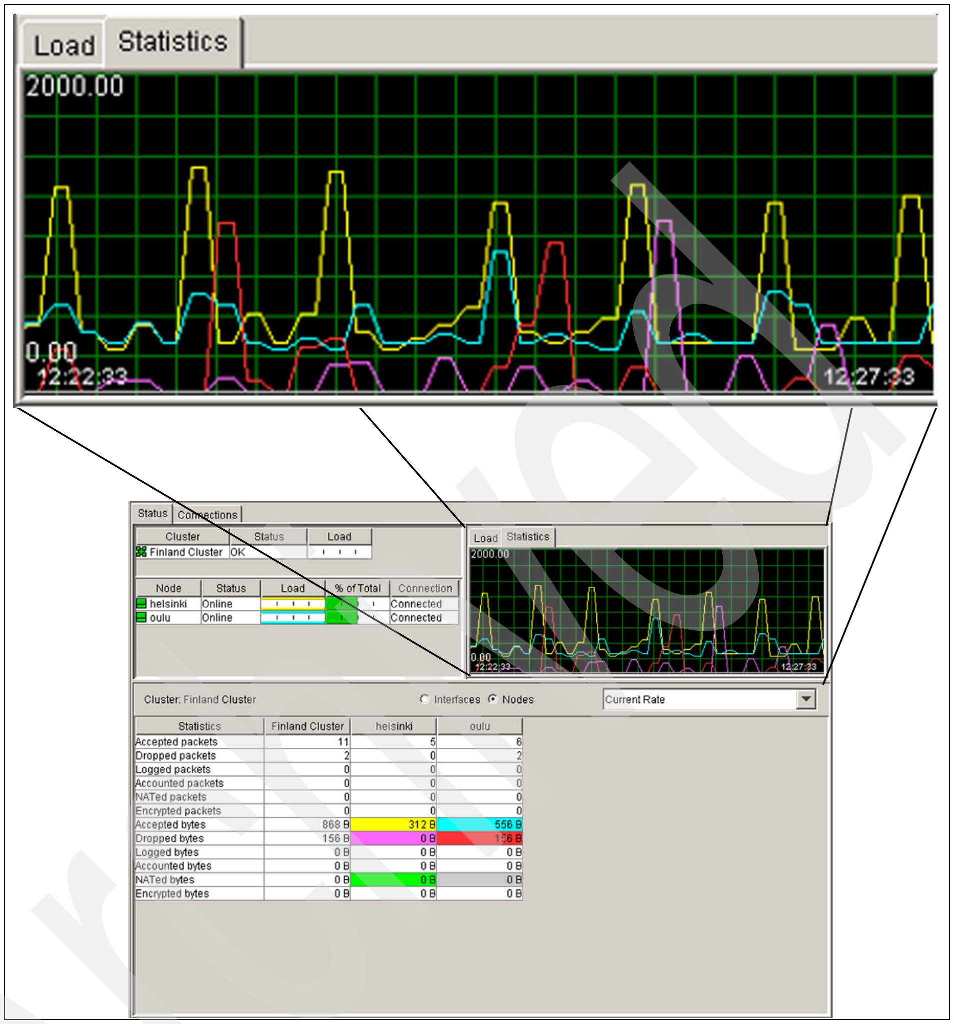
Figure 6-11 Components of the management system

The Management Server and Log Server interpret commands and data from the GUI and communicate that information to the firewall engines. Data from the firewall engines is filtered through the management system and sent to the GUI for display.

Management Server

The Management Server is the core server in the management system and the central point of administration. A single Management Server can be used to manage an entire enterprise network of StoneGate clusters and single firewalls. It is in charge of maintaining policies and network elements and communicating the administrator's instructions to the firewall engines.

The Management Server also features a monitoring system. The monitoring system receives performance information and current node conditions from the firewall nodes. It processes the raw performance data and presents it to the GUI for display in a clear, comprehensive format to the administrator. Figure 6-12 on page 101 shows performance information displayed in the GUI.



Cluster	Status	Load
Finland Cluster	OK	

Node	Status	Load	% of Total	Connection
helsinki	Online			Connected
oulu	Online			Connected

Cluster: Finland Cluster ( ) Interfaces (● Nodes) Current Rate

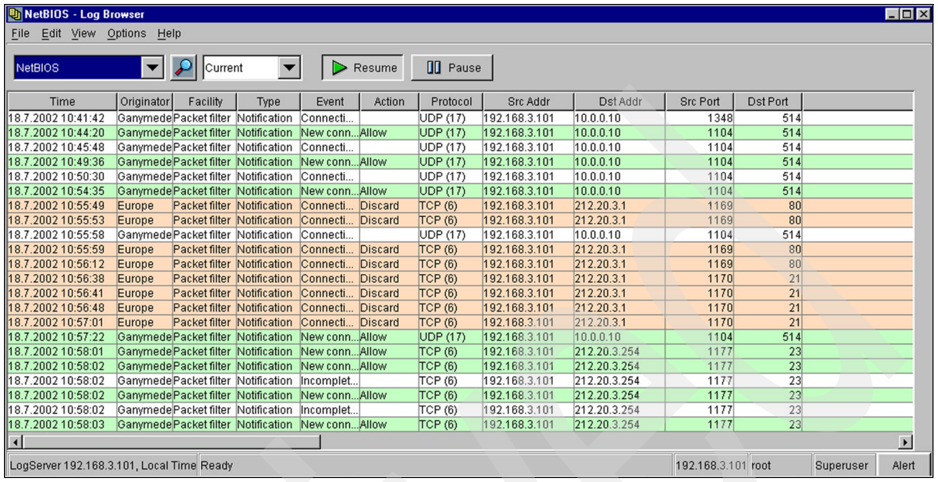
Statistics	Finland Cluster	helsinki	oulu
Accepted packets	11	5	6
Dropped packets	2	0	2
Logged packets	0	0	0
Accounted packets	0	0	0
NATed packets	0	0	0
Encrypted packets	0	0	0
Accepted bytes	868 B	312 B	556 B
Dropped bytes	156 B	0 B	156 B
Logged bytes	0 B	0 B	0 B
Accounted bytes	0 B	0 B	0 B
NATed bytes	0 B	0 B	0 B
Encrypted bytes	0 B	0 B	0 B

Figure 6-12 Statistical data from the Management Server

The monitoring system also provides several control commands for the firewall nodes. Through the GUI, the administrator can command a node or set of nodes to go online, offline, or reboot. Control requests are passed to the Management Server, which communicates the requests to the firewall engines.

Log Server

The Log Servers manage log data generated by each firewall node. Event information is stored in a database that can be queried, sorted, and exported. What data is viewed and how it is presented can be controlled through user-defined filters. Filters can process log data to be exported or deleted. An example of log data as displayed by the Log Browser is shown in Figure 6-13.



Time	Originator	Facility	Type	Event	Action	Protocol	Src Addr	Dst Addr	Src Port	Dst Port
18.7.2002 10:41:42	Ganymede	Packet filter	Notification	Connecti...		UDP (17)	192.168.3.101	10.0.0.10	1348	514
18.7.2002 10:44:20	Ganymede	Packet filter	Notification	New conn...	Allow	UDP (17)	192.168.3.101	10.0.0.10	1104	514
18.7.2002 10:45:48	Ganymede	Packet filter	Notification	Connecti...		UDP (17)	192.168.3.101	10.0.0.10	1104	514
18.7.2002 10:49:36	Ganymede	Packet filter	Notification	New conn...	Allow	UDP (17)	192.168.3.101	10.0.0.10	1104	514
18.7.2002 10:50:30	Ganymede	Packet filter	Notification	Connecti...		UDP (17)	192.168.3.101	10.0.0.10	1104	514
18.7.2002 10:54:35	Ganymede	Packet filter	Notification	New conn...	Allow	UDP (17)	192.168.3.101	10.0.0.10	1104	514
18.7.2002 10:55:49	Europe	Packet filter	Notification	Connecti...	Discard	TCP (6)	192.168.3.101	212.20.3.1	1169	80
18.7.2002 10:55:53	Europe	Packet filter	Notification	Connecti...	Discard	TCP (6)	192.168.3.101	212.20.3.1	1169	80
18.7.2002 10:55:58	Ganymede	Packet filter	Notification	Connecti...		UDP (17)	192.168.3.101	10.0.0.10	1104	514
18.7.2002 10:55:59	Europe	Packet filter	Notification	Connecti...	Discard	TCP (6)	192.168.3.101	212.20.3.1	1169	80
18.7.2002 10:56:12	Europe	Packet filter	Notification	Connecti...	Discard	TCP (6)	192.168.3.101	212.20.3.1	1169	80
18.7.2002 10:56:38	Europe	Packet filter	Notification	Connecti...	Discard	TCP (6)	192.168.3.101	212.20.3.1	1170	21
18.7.2002 10:56:41	Europe	Packet filter	Notification	Connecti...	Discard	TCP (6)	192.168.3.101	212.20.3.1	1170	21
18.7.2002 10:56:48	Europe	Packet filter	Notification	Connecti...	Discard	TCP (6)	192.168.3.101	212.20.3.1	1170	21
18.7.2002 10:57:01	Europe	Packet filter	Notification	Connecti...	Discard	TCP (6)	192.168.3.101	212.20.3.1	1170	21
18.7.2002 10:57:22	Ganymede	Packet filter	Notification	New conn...	Allow	UDP (17)	192.168.3.101	10.0.0.10	1104	514
18.7.2002 10:58:01	Ganymede	Packet filter	Notification	New conn...	Allow	TCP (6)	192.168.3.101	212.20.3.254	1177	23
18.7.2002 10:58:02	Ganymede	Packet filter	Notification	New conn...	Allow	TCP (6)	192.168.3.101	212.20.3.254	1177	23
18.7.2002 10:58:02	Ganymede	Packet filter	Notification	Incomplet...		TCP (6)	192.168.3.101	212.20.3.254	1177	23
18.7.2002 10:58:02	Ganymede	Packet filter	Notification	New conn...	Allow	TCP (6)	192.168.3.101	212.20.3.254	1177	23
18.7.2002 10:58:02	Ganymede	Packet filter	Notification	Incomplet...		TCP (6)	192.168.3.101	212.20.3.254	1177	23
18.7.2002 10:58:03	Ganymede	Packet filter	Notification	New conn...	Allow	TCP (6)	192.168.3.101	212.20.3.254	1177	23

LogServer 192.168.3.101, Local Time Ready 192.168.3.101 root Superuser Alert

Figure 6-13 StoneGate Log Browser

Multiple Log Servers can be deployed in a StoneGate environment. To improve performance and availability, each firewall or firewall cluster can be assigned a different Log Server (a particularly useful option for geographically distributed systems). Individual nodes within a single cluster or hot-standby configuration must log to the same Log Server. In the event that a node cannot communicate with its Log Server, the node logs locally until communication is restored.

Database engines

The Management Server stores properties of network elements, rules, aliases, templates, VPNs, users, services, and other information in an internal database engine. This embedded database engine provides StoneGate with a fast and robust SQL relational database. This self-maintaining database is installed automatically as a component of the Management Server.

The Log Server also uses an embedded database engine. This database allows the Log Server to store, query, and export log data very quickly, using the same kind of a fast, robust SQL relational database.

User directory

User information is stored in the Management Server's internal database. Although stored in the embedded SQL database, the user directory can be accessed through an LDAP API. This interface enables existing LDAP directory services to used for access to the user directory.

Management system backups

Proper care of system backups is an essential security precaution that should not be overlooked. In the event of a catastrophic failure of the Management Server or Log Servers, the firewall engines are designed to continue operation. In the unlikely event the management system needs to be restored (or if you want to transfer the Management Server to a new computer), a backup of the management system (Log Servers and Management Server) can be created. The backup consists of data from the Management Server database (network elements, firewall configurations, security policies, and user accounts) stored in a single ZIP file. The backup file can be encrypted and protected with a password. The Log Server and its database with all of the log entries can be backed up in a similar manner.

Firewall engines

The firewall engines at the core of StoneGate run on an integrated, hardened, Linux-based system installed with the engine. Each node can have multiple network interfaces. The engines inspect and filter packets addressed to or sent from the network they protect.

StoneGate firewalls can be configured in *standby mode*; one node is online, the other waits to takeover from the first. Should the online node fail or be taken offline, the standby node automatically assumes the role of the online node.

Supporting systems (such as the configurable test subsystem) also run on the firewall nodes. The kernel, protocol agents, firewall, test subsystem, and logging services each function independently. This ensures that the engine is robust; an error or corruption in one service will not affect other services. The system can continue until repairs or regular maintenance corrects them. The engine on each node communicates with the management system sending performance statistics, state information, and log data and receiving policy updates and configuration changes.

Engine are configured through configuration files loaded onto the nodes by the management system. These files are encrypted on disk for each node. Unauthorized access to the firewall itself will not reveal further information about the network. Configuration files are generated by the management system based on input from the administrator through the GUI. No manual re-configuration is required, and mode software nodes can be upgraded remotely from the GUI. There is no need to physically access the firewalls in order to upgrade perform an upgrade.

Each firewall node runs a test subsystem. Administrators can define or enable additional tests and test conditions on the firewall nodes. For example, if a network interface fails, a node can be switched offline, traffic redistributed to remaining online system, and an alert sent to an administrator.

6.3.3 Communications between the components

Table 6-1 summarizes the communications between different StoneGate components and shows traffic volumes typically generated. Traffic volumes are categorized as:

- High
  High volume regular traffic. A high throughput link is desirable.
- Medium
  A low bandwidth link, such as a modem, might not suffice.
- Low
  One-time contact only.

Table 6-1 Communication between StoneGate components

	GUI client	**Log Server**	**Management Server**	**Engine**
GUi client	NA	Medium when the client requests log information.	Medium: Periodically and on user action.	Never
Log Server	Never	NA	Medium at startup, and then after that periodically and low. A broken link does not disturb the log server's operation.	Never
Management Server	Never	Never	NA	Medium: when monitoring and uploading a configuration.
Engine	Never	High when sending firewall logs.	Low: In initial configuration only.	Medium to High: Heartbeat and state sync traffic if HA mode is used.

The GUI client never communicates with the firewall engines directly. Instead, communication is though the Management Server. The highest volume of traffic is generated by the logs sent from the engines to the Log Server. To reduce log traffic, log data can be filtered. Each engine defined in a cluster communicates with all other engine in that cluster, sharing information about current connections, load, and state information through Ethernet (IEEE 802) multicast.

6.3.4 Network address translation between components

StoneGate components can be deployed in a distributed architecture; components located on different can be managed from a single location. Network address translation (NAT) is required for component communication in this configuration. StoneGate supports NAT for management communications on the basis of contact IP addresses that can be defined in the GUI for each component.

6.3.5 Secured communication

The firewall engines, Management Server, and Log Servers authenticate using public key cryptography and digital certificates based on Secure Sockets Layer (SSL) protocol. This ensures that only authorized nodes and an authorized management system can take part in the cluster's operation and configuration. Keys and certificates are automatically generated during installation. An internal StoneGate certificate authority (CA) is created automatically when installing the Management Server.

6.3.6 Certificate backups

In addition to performing backups of the Management and Log Server databases, you should create at least one backup of the certificates used by the management system and firewall engines. Store the certificates in a secure location. You can use the option of encrypting them with a password as well. Backed-up certificates can be easily restored when needed, for example, when you are reinstalling your Management Server on a new computer and you want to use existing certificates instead of generating new ones.

6.3.7 Distributed management

With its distributed architecture, a single management system can monitor, control, and manage multiple StoneGate firewall clusters throughout the network. A distributed system is illustrated in Figure 6-14 on page 106.

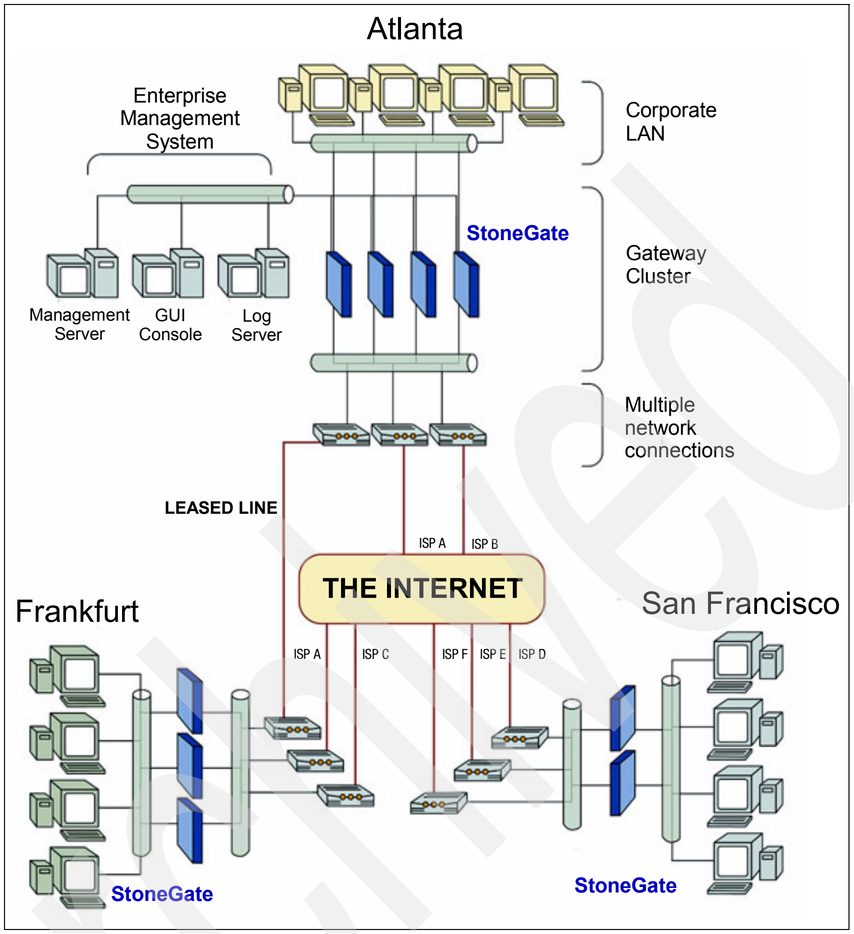


Figure 6-14 Example of a distributed enterprise network

The main administrator can deploy a management system on a dedicated, trusted LAN segment at the corporate headquarters. The Log Server and Management Server can be installed on a single server, or multiple machines. Additional GUI clients can be run from any trusted system. With this configuration, an administrator can define corporate-wide security policies as well as policies for remote office networks. StoneGate clusters in different offices can all be monitored, controlled, and managed from the same graphical interface interacting with a single, distributed management system.

Local administrators can be defined with different administrator rights to delegate management tasks. They can be given certain permissions relating, for example, to the maintenance and monitoring of the system or rights to configure specific types of elements.

When used in an MSP environment, administrators can centrally manage firewalls for several different customers with a single StoneGate management system. Each element (network element, VPN, custom service, license, rule base, or LDAP domain) in a given customer network can be categorized in the GUI to distinguish it from other customer elements.

6.3.8 Implementation strategies

One of the most useful StoneGate features is the ability to distribute components on different machines. Each machine can run on a different platform, for example, running the Log Server on Linux while running the Management Server on Solaris and the user interface on a Microsoft Windows platform.

Single management system

One possible implementation, depicted in Figure 6-15 on page 108, runs the Management Server, Log Server, and GUI on the same machine. This strategy has multiple cost saving benefits:

- Minimized hardware expense because of multiple servers on the same machine.
- Simplified installation and reduced administrative costs.

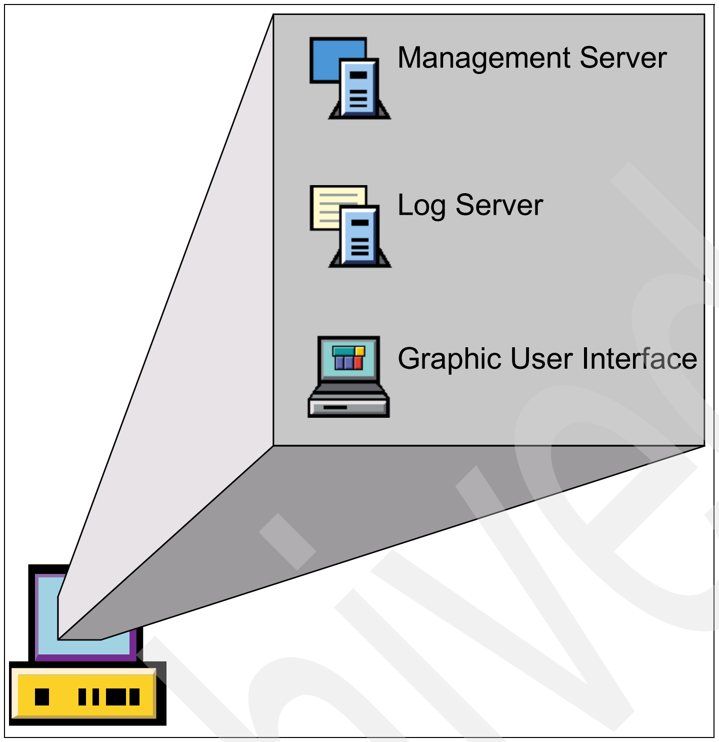


Figure 6-15 Single management system

Distributed management system

Using the distributed management system, depicted in Figure 6-16 on page 109, the Management Server, Log Server, and GUI are installed on different machines. This strategy can improve performance; each machine performs only a portion of the work of the entire system. In addition, this offers more flexibility; any StoneGate component can run on any platform.

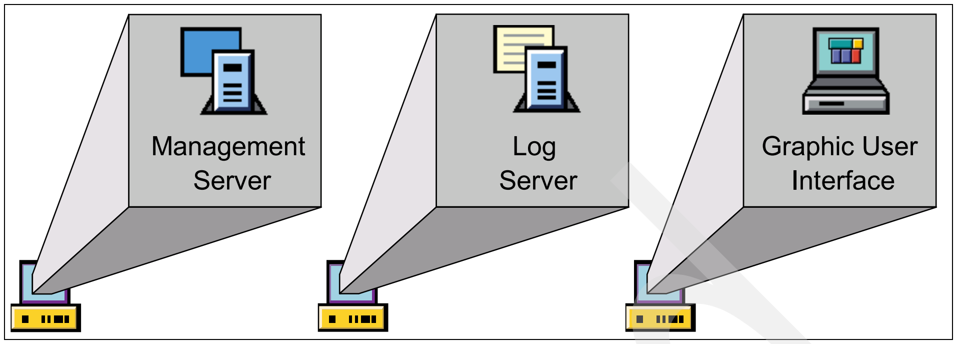


Figure 6-16 Distributed management system

Virtual LANs

A virtual local area network (VLAN) is a logical grouping of hosts and network devices appearing as a single local area network (LAN) segment, regardless of physical topology. Workstations and servers in the same VLAN operate as though connected to the same physical network, forming a single OSI Layer 2 broadcast domain. VLAN segregation is implemented by VLAN tagging as defined by the IEEE 802.1q standard. Network ports on a switch can be configured to make a distinction between different VLANs by adding a two-byte VLAN tag on each Ethernet frame that passes between ports. Control over network traffic is enhanced by containing broadcast and multicast traffic within the VLAN. Unnecessary traffic to other LAN segments is reduced. With the separation of the logical and physical LAN structure, the topology of a VLAN can follow, for example, a company's organizational structure (connecting department users to the same VLAN, even if located in different physical LANs). This also reduces required cabling; LAN segments are separated at the switch.

StoneGate supports VLAN tagging. It can distinguish traffic belonging to different VLANs. StoneGate can be configured with VLANs in the GUI. This can reduce the number of required physical interfaces and simplify deployment on geographically distributed firewall clusters. An example StoneGate VLAN deployment is shown in Figure 6-17 on page 110.

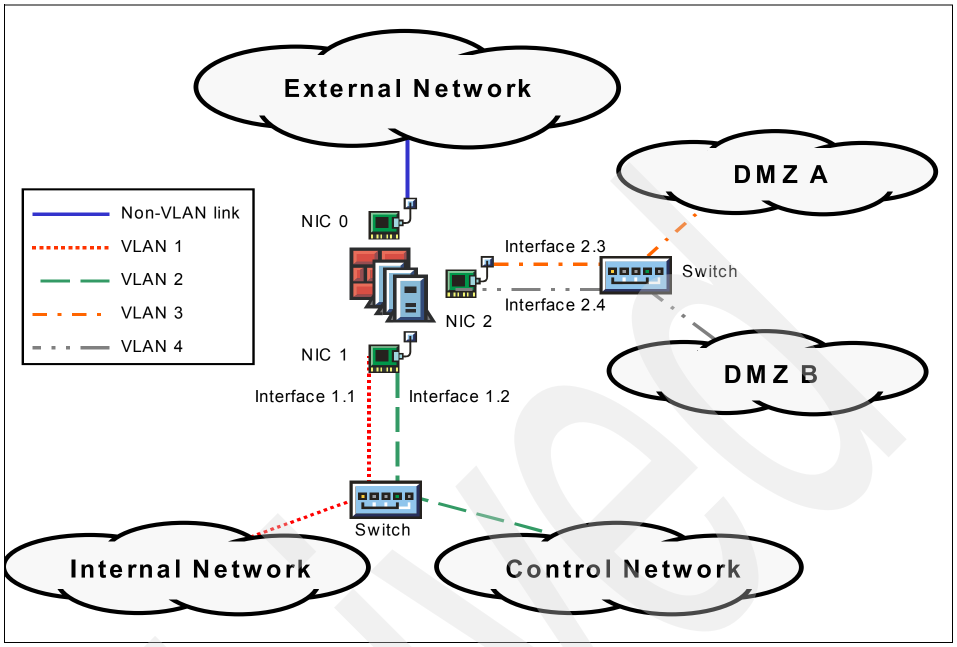


Figure 6-17 StoneGate with VLAN tagging

Each firewall nodes has three network interface cards (NIC), but is connected to five isolated networks:

- NIC 1 is configured as two logical VLAN interfaces:
  - Interface 1.1 for VLAN 0
  - Interface 1.2 for VLAN 1

  The VLANs isolate the internal network from the control network.
- NIC 2 is configured as interface 2.3 and 2.4 for VLANs 2 and 3 to isolate the two DMZ networks (DMZ A and DMZ B).
- NIC 3 is connected to the Internet.

For a secure implementation, the access to the switches is restricted. Traffic belonging to different VLANs is securely isolated. Figure 6-18 on page 111 depicts a geographically distributed VLAN cluster.

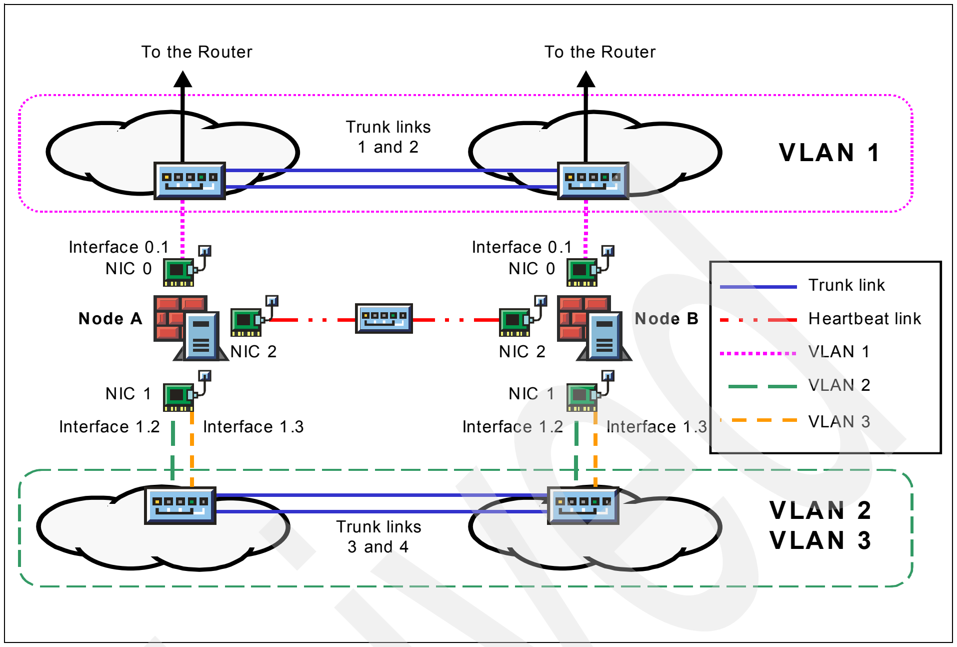


Figure 6-18 Geographically distributed cluster

For example, Node A could be located in one building, while Node B is located in another. A heartbeat link is needed between the nodes. Both nodes have an interface to VLAN 1 and another interface to VLAN 2 and VLAN 3.

6.4 StoneGate on Linux for zSeries

StoneGate is the only commercially available and certified firewall and VPN product for IBM @server systems. StoneGate for zSeries removes the need for external firewall servers between front-end and back-end applications, saving cost in investment, labor, and maintenance. StoneGate enables multiple simultaneous firewalls in a virtual network environment and is able to use the high degree of scalability offered by zSeries. Centralized management system simplifies deployment and maintenance of multiple firewall instances. StoneGate on zSeries features VPNs that enhance security for sensitive network traffic (such as financial transactions within the mainframe). Figure 6-19 on page 112 depicts a zSeries StoneGate deployment.

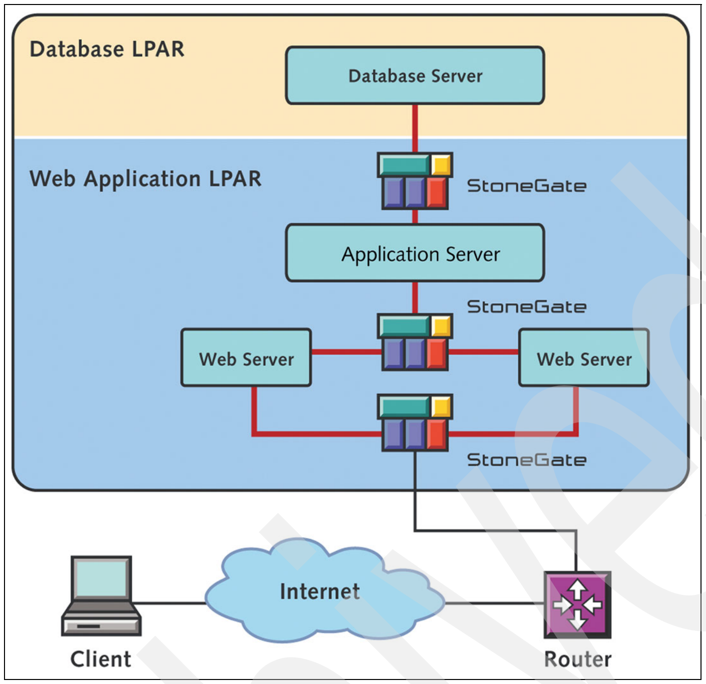


Figure 6-19 StoneGate and zSeries mainframe

6.4.1 High availability technologies

In the past, firewalls could be made highly available through add-on software or redundant hardware solutions. Often, this resulted in moving a single point of failure to another network component (typically the network link). Other network services (such as Web, FTP, or content inspection servers) would require yet more hardware and software to eliminate single points of failure in the enterprise.

StoneGate high availability firewall and VPN includes several clustering technologies to reduce the complexity needed for high availability. StoneGate's innovative clustering and load-balancing features provide several benefits over traditional solutions.

Built-in high availability

To achieve firewall high availability, additional hardware switches, software clustering products, or special load balancing devices are often used. With StoneGate, high availability is built into the firewall engines. Individual connections can be transparently moved to a hot-standby node if an active node becomes overloaded or experiences a failure.

Note: Connection switching also happens during remote upgrades.

Using virtual IP addresses combined with MAC addresses, firewall engines are seen by the rest of the network as a single entity (reducing the need for configuration changes elsewhere in the network). The Cluster Virtual Interface *(*CVI) has IP and MAC addresses shared by all the nodes in a cluster. CVIs are used for any load-balanced traffic between nodes. Traffic arriving at a CVI reaches all the nodes simultaneously. However, only a single node processes a given connection. Figure 6-20 illustrates a high availability set of firewall nodes.

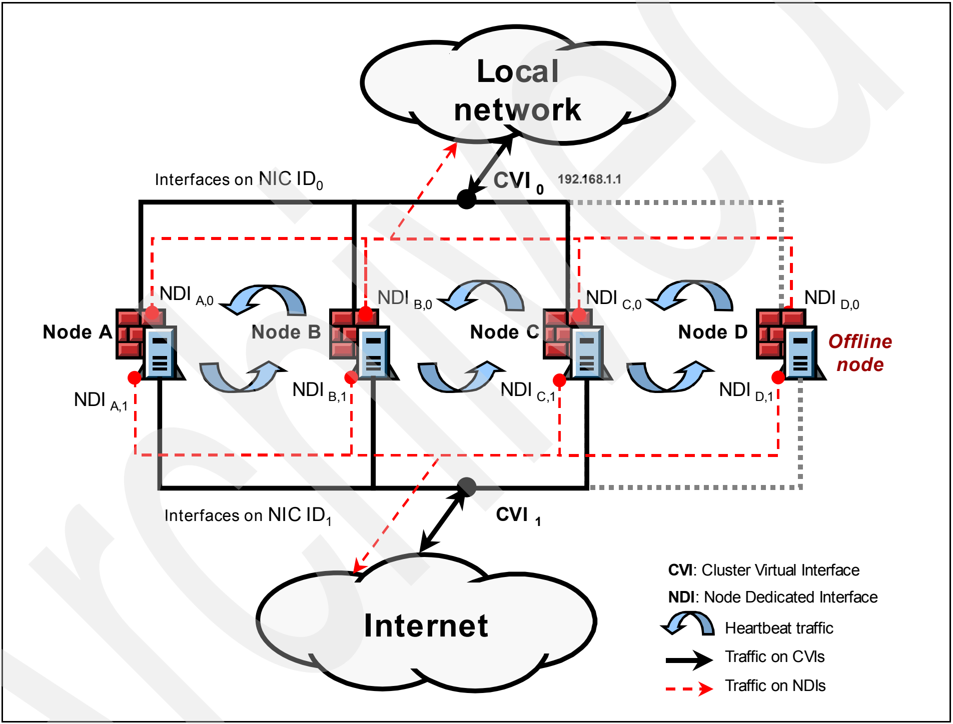


Figure 6-20 Cluster Virtual Interfaces

Each firewall node is assigned two NICs:

- Each node's NIC ID0 interface addresses the CVI0 virtual interface. This virtual interface IP address shared is shared by all nodes; the Node Dedicated Interface (NDI) IP address is unique for each node.
- Each node's NIC ID1 interface addresses the CVI1 virtual interface. This virtual interface provides connectivity to the Internet.

The number of NICs is limited by the network interface card hardware, but any number and combination of CVIs and NDIs can be configured on a single NIC. VLANs can alleviate any possible shortage of available physical interfaces. Each engine communicates with other engines in the host standby configuration with a special heartbeat protocol (preferably run on a dedicated network to ensure inter-node communications can take place even in the event of a denial-of-service attack or periods of peak network traffic).

High availability provides for scalability and maintenance during business hours. Individual nodes can be taken offline. Connections previously handled by the engine are transparently transferred to the standby node.

High availability connections

StoneGate provides high availability options for related network components. Internet service providers (ISPs) and virtual private networks (VPNs) have often been a single point of failure for enterprise communications. High availability was attainable, but at a cost of complexity. StoneGate eliminates these issues by introducing innovative technologies.

As the role of Internet-driven business grows, the reliability of connections and constant availability of services is an absolute necessity for corporations. Even with clustering products, corporate network are subject to outages should network links fail. Elimination of this single point of failure often requires a battery of redundant external routers and switches using complex routing protocols, such as Border Gateway Protocol (BGP) and Hot Standby Routing Protocol (HSRP), as well as peering arrangements through the ISPs providing Internet connections. In addition to the expense of redundant hardware, network administrators face additional complexity to achieve high availability.

Multilink technology

StoneGate enables network administrators to use several ISPs with multilink technology This removes problems associated with high availability routing protocols. High availability network links reduce the need for costly leased lines. Any IP-based link with a dedicated IP address range can be used as part of a multilink configuration:

- Standard Internet connections provided by ISPs
- ISDN connections
- Leased lines

With multilink technology, redundant routes to the Internet are represented as a special network element (a NetLink) in the StoneGate GUI. Network traffic is load balanced between NetLinks, as depicted in Figure 6-21 on page 115.

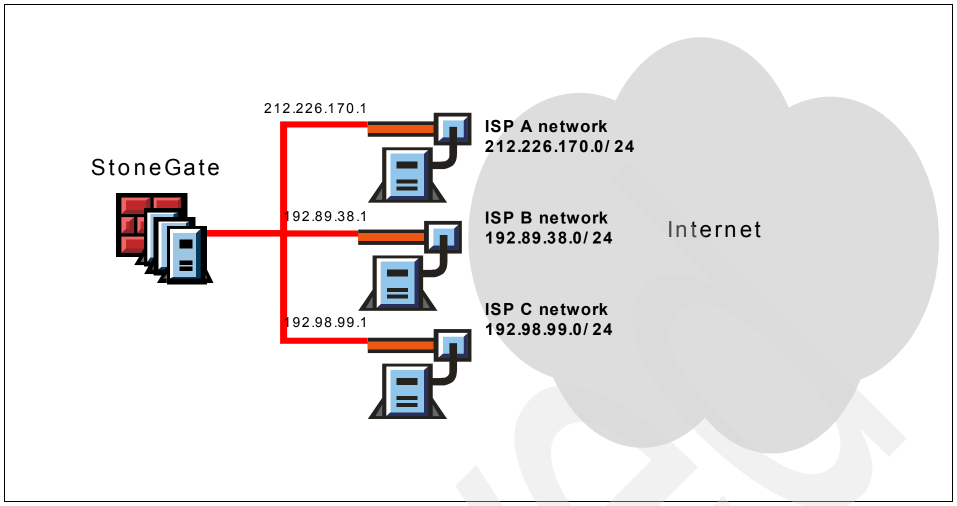


Figure 6-21 StoneGate multilink technology for high availability

NetLinks enable easily configured load-balanced connections in the event of a router or network failure.

Note: The number network links is not limited by or dependent on the number of the physical firewall interfaces. A single NIC can accommodate several CVIs, each defined in a different IP address space. In this way, a single physical interface can connect several ISPs if required. When firewall nodes are clustered, a node failure does not mean that all connections to the ISP are lost. Instead, connections are immediately switched over to a functioning node.

Load-balanced routing

Traffic can be load balanced across all available links, improving overall network throughput. For example, Stonesoft's patented load-balanced routing technology enables routing through the fastest available link. The dynamic routing engine (which performs load-balanced routing) sends an initial packet out through each network link, measuring the response times. Based on the measurement, connections are routed automatically through the link with the best performance.

Increasing the bandwidth

As bandwidth requirements increase, the traditional solution has been to pay more to the existing provider; changing providers would result in the expensive change of public IP addresses. With StoneGate, you can increase the bandwidth by simply adding new ISPs to your network. StoneGate can incorporate multiple ISPs into a single, virtual Internet connection.

6.4.2 Benefits of multilink technology

The advantages of StoneGate's multilink technology include:

- More secure networks
  There are fewer possibilities for traffic to bypass firewalls through other network devices.
- Less complicated networks
  The firewall takes care of the load balancing (eliminating the need for special load re-directors, switches, or other devices that might introduce single points of failure). The time and cost required for network maintenance and configuration is reduced.

Multilink technology simplifies the use of multiple ISPs and increases the availability and bandwidth at use for critical applications. Transactions with load-balanced routing always select the optimal route to a destination.

6.4.3 Applying multilink technology

Multilink technology can be applied in three unique scenarios:

- Outbound traffic management
- Multilink VPNs
- Inbound traffic management

Each scenario handles high-availability and load balancing differently.

Outbound traffic management

Outbound traffic management in StoneGate provides both high availability and load balancing between multiple NetLinks. Load balancing between ISPs increases overall network throughput. For each connection request, StoneGate firewall can select the fastest route for the connection. Figure 6-22 on page 117 illustrates outbound traffic management decisions, using the round trip time of the first SYN packet as the basis for the load-balanced routing decision.

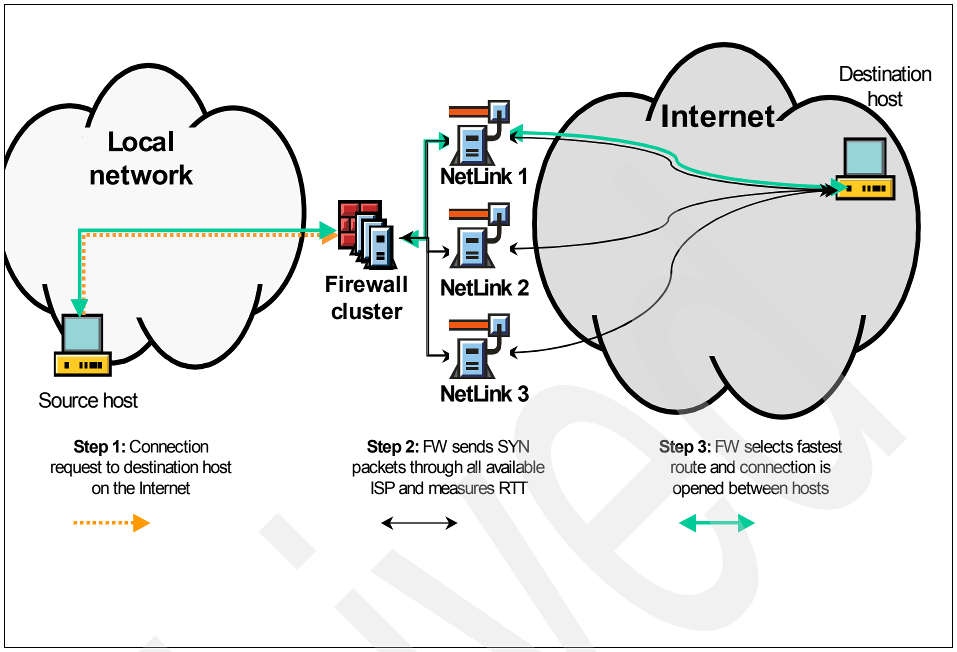

Local network
Source host
Firewall cluster
NetLink 1
NetLink 2
NetLink 3
Internet
Destination host
Step 1: Connection request to destination host on the Internet
Step 2: FW sends SYN packets through all available ISP and measures RTT
Step 3: FW selects fastest route and connection is opened between hosts

Figure 6-22 Outbound load balancing

Multilink VPNs

VPNs are an increasingly critical component of enterprise networks. Trusted communication with remote offices or users through the insecure public Internet is possible using encrypted tunnels to internal networks. Historically, VPN tunnels were subject to failure in the event of router or link failures. Often, only one VPN tunnel could be established with a remote site, placing bandwidth restrictions on an already limited connection.

Using StoneGate, clustered, load-balanced IPsec VPN tunnels are now possible. A logical tunnel consisting of one or more physical VPN tunnels through different ISPs is possible. Figure 6-23 on page 118 demonstrates a VPN consisting of two physical links at one end and three at the other. In this example configuration, StoneGate can use six different *subtunnels*, forming one virtual VPN tunnel to load balance VPN traffic between sites.

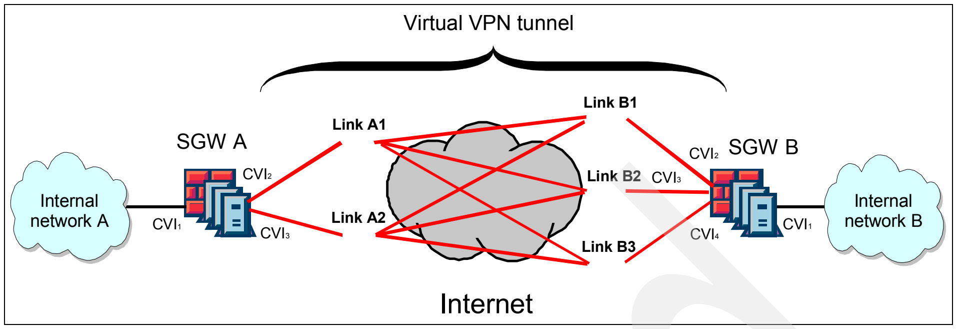


Figure 6-23 Clustered, highly available VPN tunnels

If one subtunnel collapses, connections using that tunnel are automatically shifted to another subtunnel established with the remote site. Should all ISP links be down, StoneGate can bring standby links (such as leased lines or dialup connections) into use. This introduces true fault tolerance and high availability to VPN traffic.

Inbound traffic management

StoneGate can load balance connections to different services in an internal network (Web servers, for example). With existing technologies, it is often necessary to deploy additional load-balancing switches, clustering software, or special load-balancing hardware to ensure high availability and improved performance. As shown in Figure 6-24 on page 119, network administrators are faced with additional complexity to achieve network high availability.

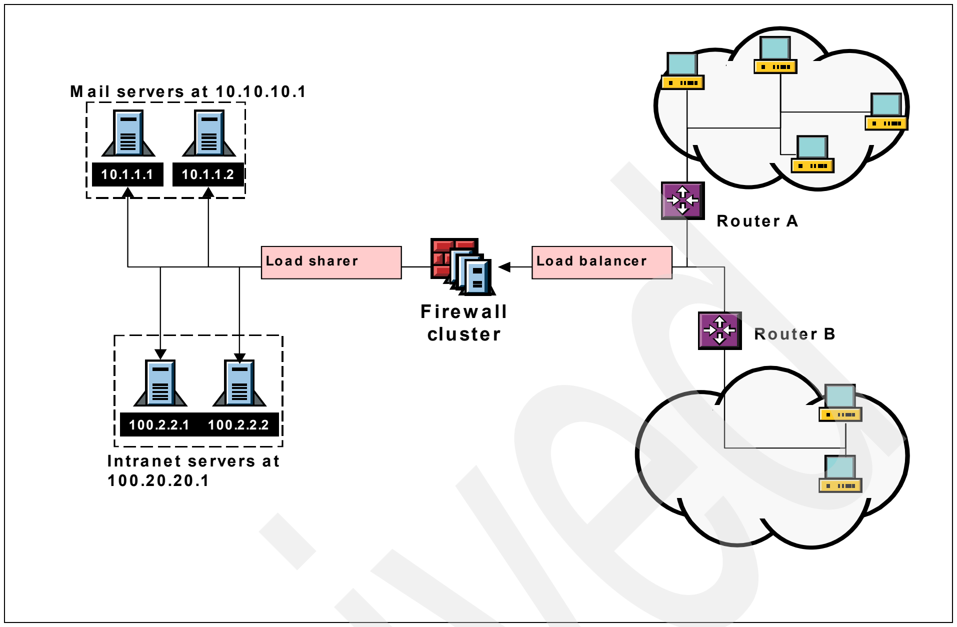


Figure 6-24 Traditional inbound load balancing

With StoneGate, administrators can taking advantage of multilink technology to load balance network services. For example, a Web server pool can be set up on a DMZ network, as illustrated in Figure 6-25 on page 120. StoneGate can share connections to the pool using load balancing between ISPs. The selection of the ISP can be based on dynamic DNS updates, or on multiple DNS entries. StoneGate constantly monitors server availability to ensure that connections from a particular source address are handled by one server only.

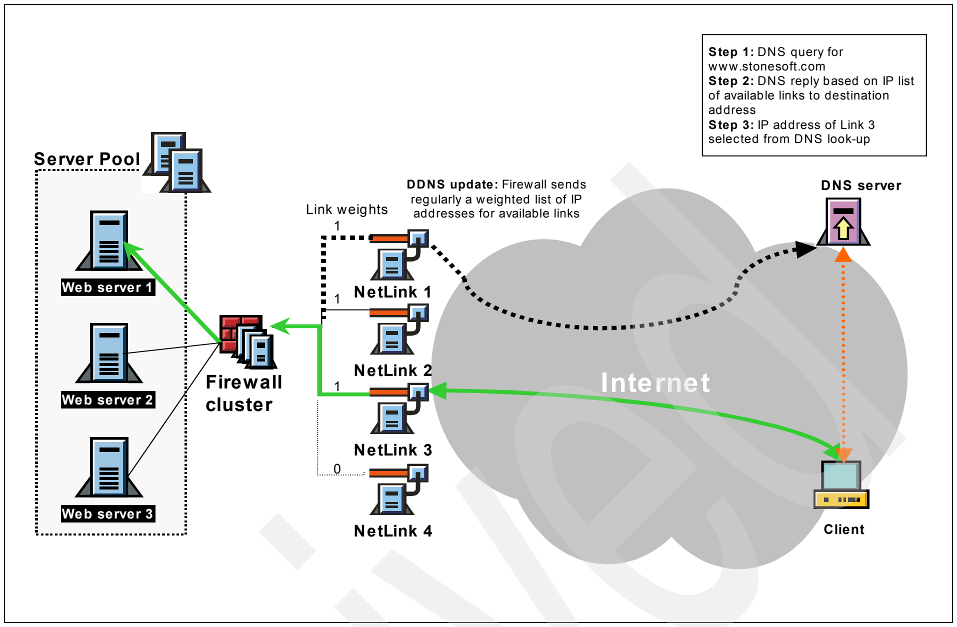

Step 1: DNS query for www.stonesoft.com
Step 2: DNS reply based on IP list of available links to destination address
Step 3: IP address of Link 3 selected from DNS look-up
Server Pool
Web server 1
Web server 2
Web server 3
Firewall cluster
Link weights
1
1
1
0
DDNS update: Firewall sends regularly a weighted list of IP addresses for available links
NetLink 1
NetLink 2
NetLink 3
NetLink 4
Internet
DNS server
Client

Figure 6-25 StoneGate inbound traffic management with dynamic DNS

StoneGate administration

StoneGate also offers flexible access control for configuring and maintaining firewall clusters, networks, users, and other related components. Access to the GUI is provided at three different user levels:

- Superuser
  The Superuser is the highest level of access, much like the Administrator account in Microsoft Windows NT® or root in UNIX. This level enables the creation of other administrator accounts, as well as the ability to create, modify, or delete network objects, rule bases, templates, users, and so on. The Superuser can also grant additional access privileges to Editors or Operators.
- Editor
  This level is the intermediate access level. Editors can modify some objects, users, and rule bases. By default, they cannot modify templates, create or modify editors, or delete objects. Additional privileges can be given to them by Superusers.

- Operator
  These are the least-privileged accounts, enabling the monitoring of systems and the viewing of rule bases and configuration information. However, Operators cannot modify elements by default. They can be granted additional privileges by a Superuser, or by an Editor who has been given that privilege by the Superuser.

These levels enable the delegation of administrator tasks in enterprise networks. Additional privileges can be granted to lower levels. The network administrator can create multiple accounts of each type, with unique user IDs and passwords.

6.5 StoneGate installation

In this section, we describe the installation of StoneGate for zSeries firewall and VPN on a z/VM guest.

6.5.1 The z/VM guest definition

The z/VM guest used to host StoneGate should be defined with class G privileges. StoneGate requires two type 3390-3 DASD devices with at least 1 GB per minidisk device. A sample user directory entry for the StoneGate virtual machine is shown in Example 6-1.

Example 6-1 Example z/VM user directory entry for StoneGate

```
USER LNXSG2 LNXSG2 128M 1G G
  INCLUDE IBMDFLT
  IPL 190 PARM AUTOCR
  MACHINE XA
  DEDICATE 7104 7140
  DEDICATE 7105 7141
  DEDICATE 7106 7142
  LINK TCPIP 0592 0592 RR
  MDISK 0191 3390 3238 100 440U1R
  MDISK 0171 3390 001 3338 LX15C9
  MDISK 0172 3390 001 3338 LX15CB
```

We use the 171 and 172 minidisks for StoneGate. The 100 cylinder (30 MB) 191 minidisk is used for the StoneGate for zSeries firewall and VPN installation files. An OSA-Express adapter is defined to the guest using virtual addresses 7104-7106 (real addresses 7140-7142).

6.5.2 Ensuring file integrity

Before installing StoneGate, ensure installation file integrity using MD5 or SHA-1 checksums. The checksums can be found on the StoneGate installation CD-ROM and from the release information section of the Stonesoft Web site at:

http://www.stonesoft.com/support/StoneGate

To automatically check MD5 sums, issue the **md5sum** command in the directory containing the StoneGate installation files, as shown in Example 6-2.

Note: By default, Windows does not have MD5 or SHA-1 checksum programs; however, several third-party programs are available.

Example 6-2 Checking file integrity using the md5sum command

```
$ md5sum -c SG2.1.0-MD5.txt
boot_catalog: OK
install.bin: OK
sg_engine_2.2.3.1017_s390.list: OK
sg_engine_2.2.3.1017_s390.zip:
```

The **-c** option specifies that MD5 sums are to be checked against values provided in the SG2.1.0-MD5.txt file.

Use the **sha1sum -c** ***sumfile*** command to compare SHA-1 checksums

Important: Do not use files that have invalid checksums. Contact Stonesoft technical support to resolve the issue.

6.5.3 Downloading the installation files to z/VM

Begin installation by transferring the StoneGate installation files to the A-disk of the StoneGate z/VM guest. In Example 6-3 on page 123, we show the command sequence to download the installation files from an anonymous FTP server.

Important: Make sure that network connections from the guests protected by StoneGate are routed through the firewall engine to ensure that the firewall cannot be bypassed.

Example 6-3 Downloading the StoneGate installation files

```
link tcpmaint 592 592 rr                          1
Ready; T=0.01/0.01 19:42:00
acc 592 b
DMSACP723I B (592) R/O
Ready; T=0.01/0.01 14:09:47
ftp 9.12.6.170                                    2
VM TCP/IP FTP Level 440
Connecting to 9.12.6.170, port 21
220 Serv-U FTP Server v4.1 for WinSock ready...
USER (identify yourself to the host):
anonymous
>>>USER anonymous
331 User name okay, please send complete E-mail address as password.
Password:

>>>PASS ********
230 User logged in, proceed.
Command:
locsite fix 80                                    3
>>>TYPE a
200 Type set to A.
Command:
ascii                                             4
Command:
get SGINST.EXEC (REPLACE                          5
Command:
get SGINST.PARMFILE (REPLACE                      6
Command:
binary                                            7
Command:
get SGINST.KERNEL (REPLACE                        8
Command:
get SGINST.INITRD (REPLACE                        9
Command:
quit
```

The highlighted steps are:

1. Link to the TCPMAINT 592 minidisk to gain access to the FTP commands. We access the minidisk as our B-disk.
2. Access the FTP server containing the StoneGate installation files using the **ftp** command. When prompted, provide the appropriate user ID and password.
3. Files are stored locally in fixed block format with a record length of 80.
4. Issue the **ascii** subcommand to transfer text files in ASCII mode.

5. The SGINST.EXEC file initiates the installation from the VM reader.
6. The SGINST.PARMFILE file contains the installation boot parameters.
7. Issue the **binary** subcommand to transfer text files in binary mode.
8. The SGINST.KERNEL file contains the installation kernel.
9. The SGINST.INITRD file contains the installation initial ramdisk.

6.5.4 Installing the firewall engine

Run the SGINST EXEC script to start the installation. This script punches the StoneGate installation files to the VM reader, and then IPLs from the reader. The output is shown in Example 6-4.

Example 6-4 Executing the SGINST EXEC script

```
sginst
0000003 FILES PURGED
RDR FILE 0015 SENT FROM LNXFW4   PUN WAS 0015 RECS 034K CPY  001 A NOHOLD NOKEEP
RDR FILE 0016 SENT FROM LNXFW4   PUN WAS 0016 RECS 0001 CPY  001 A NOHOLD NOKEEP
RDR FILE 0017 SENT FROM LNXFW4   PUN WAS 0017 RECS 249K CPY  001 A NOHOLD NOKEEP
0000003 FILES CHANGED
```

After the system is up, the StoneGate end-user license agreement (shown in Example 6-5 on page 125) is displayed. Scroll down to read through the license agreement. To accept the license agreement, type `YES`. To cancel the installation, type `NO`.

Note: Arguments must be in uppercase, because the installation wizard is case sensitive.

Example 6-5 The StoneGate license agreement

```
PLEASE READ CAREFULLY THE TERMS OF THIS END USER LICENSE AGREEMENT
PRIOR TO FIRST USE OF THE STONEGATE PRODUCT. BY INSTALLING OR USING
THE PRODUCT YOU AGREE TO BE BOUND BY THE TERMS OF THIS AGREEMENT. IF
YOU DO NOT AGREE TO ALL OF THE TERMS YOU MAY NOT USE THE PRODUCT. IN
THIS CASE YOU ARE ENTITLED TO, PRIOR TO THE FIRST USE OF THE PRODUCT,
RETURN THE PRODUCT TO THE PLACE OF PURCHASE FOR A FULL REFUND.

1. DEFINITIONS
"Channel Partner" means distributors and resellers authorized by
Stonesoft or its distributors to sell the Stonegate Product.

"License File" means the file created by Stonesoft according to the
Proof of License based on the purchase order made by Licensee or
--More--


License agreement must be accepted before continuing.
Type YES to continue and NO to cancel.

Do you want to continue? YES
```

Defining DASD for the StoneGate guest

After accepting the license agreement, you are prompted to define two DASD devices for use by StoneGate. Enter the DASD device numbers when prompted in the screen shown in Example 6-6 on page 126. We use the 170 and 171 minidisks defined to the LNXSG2 user.

Note: To manually modify the DASD configuration, refer to the “Installing the Engine in Expert Model” section of the *StoneGate zSeries Installation Guide*.

Example 6-6   Define DASD devices for the StoneGate guest

```
StoneGate Engine Installation System

First you have to define two DASDs to be used with StoneGate.

List of detected subchannels, device number is the first column:

Device sch.  Dev Type/Model CU  in use  PIM PAM POM CHPIDs
----------------------------------------------------------------------
0009   0000           3215/00  yes     80  80  FF  00000000 00000000
000C   0001           2540/00          80  80  FF  00000000 00000000
000D   0002           2540/00          80  80  FF  00000000 00000000
000E   0003           1403/00          80  80  FF  00000000 00000000
0190   0004  3390/0A   3990/E2          80  80  FF  FDFFFFFF FFFFFFFF
019D   0005  3390/0A   3990/E2          80  80  FF  FDFFFFFF FFFFFFFF
019E   0006  3390/0A   3990/E2          80  80  FF  FDFFFFFF FFFFFFFF
0191   0007  3390/0A   3990/E2          80  80  FF  FDFFFFFF FFFFFFFF
0170   0008  3390/0A   3990/E2          80  80  FF  FDFFFFFF FFFFFFFF
0171   0009  3390/0A   3990/E2          80  80  FF  FDFFFFFF FFFFFFFF
0592   000A  3390/0A   3990/E2          80  80  FF  FDFFFFFF FFFFFFFF
Enter the device number of first DASD: 170
Enter the device number of second DASD:171
0170(ECKD) at ( 94:  0) is dasda      : active at blocksize: 4096, 601020 blocks, 2347 MB
0171(ECKD) at ( 94:  4) is dasdb      : active at blocksize: 4096, 601020 blocks, 2347 MB

Check that the DASDs are correct.
Type YES to continue and NO to cancel.

Do you want to continue? YES
```

Accept the DASD assignments by typing `YES` in the screen.

Note: To redefine the DASD assignment, type `NO`.

Selecting the installation type

If no existing StoneGate installation is detected, the installation screen shown in Example 6-7 on page 127 opens. Select type 1 for a normal full installation.

Example 6-7 Selecting the StoneGate installation type

```
Existing StoneGate installation has not been detected.

1. Full install
2. Full install in expert mode
Enter your choice: 1
Hard disk will be partitioned automatically and all existing data will be lost.
Type YES to continue and NO to cancel.

Do you want to continue? YES
```

If an existing StoneGate installation is detected, the screen shown in Example 6-8 opens. Select option 3 for a full reinstallation (this does not preserve the existing configuration).

Example 6-8 StoneGate reinstallation screen

```
Do you want to continue? YES

An existing StoneGate Engine installation has been detected.

1. Upgrade existing installation
2. Re-install using configuration from existing installation
3. Full re-install (old configuration is not preserved)
4. Full re-install in expert mode

Enter your choice: 3
```

Note: Details about upgrading existing StoneGate installations can be fund in the *StoneGate Administrator's Guide* and the *StoneGate Installation Guide*.

Completing the StoneGate installation

The installation proceeds with the automatic partitioning of the DASD devices and installing the StoneGate firewall, as shown in Example 6-9 on page 128.

Example 6-9 Automatic DASD partitioning in a full StoneGate installation

```
Hard disk will be partitioned automatically and all existing data will be lost.
Type YES to continue and NO to cancel.

Do you want to continue? YES
DASD 0170 is correctly formatted with blocksize 1024. Formatting skipped...
DASD 0171 is correctly formatted with blocksize 4096. Formatting skipped...

Partitioning hard disk...
Creating filesystems...
Extracting StoneGate image...
```

IPL firewall engine

The StoneGate firewall engine is now installed and ready for configuration. IPL the StoneGate engine using the **`#CP IPL 171`** command (where 171 is the second DASD device defined to the StoneGate firewall engine), as shown in Example 6-10 on page 128.

Example 6-10 IPL StoneGate firewall engine

```
Installing boot loader...
Executing zIPL (disc0/part1)...
The StoneGate can be started by IPLing the DASD number 0171
Installation finished! Please IPL the DASD to continue.

#cp ipl 171
cp ipl 171
Linux version 2.4.21+smp (root@bluedev) (gcc version 2.95.4 20011002 (Debian pre
release)) #1 SMP Mon Nov 3 11:45:45 UTC 2003
We are running under VM (31 bit mode)
This machine has no PFIX support
This machine has an IEEE fpu
On node 0 totalpages: 65536
zone(0): 65536 pages.
zone(1): 0 pages.
zone(2): 0 pages.
Kernel command line: root=/dev/discs/disc0/part1 devfs=mount ro noinitrd dasd=0171,0172
vmpoff="LOGOFF"

Highest subchannel number detected (hex) : 0016
Calibrating delay loop... 946.99 BogoMIPS
Memory: 255844k/262144k available (1834k kernel code, 0k reserved, 658k data, 56k init)
```

6.5.5 Configuring the StoneGate firewall engine

After IPLing the firewall engine, the Configuration Wizard (shown in Example 6-11 on page 129) opens. During initial configuration, the operating system settings, network interfaces, and Management Server point of connection are defined.

Example 6-11 Configure operating system settings screen

```
Welcome to the StoneGate Engine Configuration Wizard.

This wizard will configure the StoneGate engine and
contact the management server. After this you can perform
all tasks using the administration client.

No channel devices detected

Step 1 of 3: Configure OS settings
--------------------------------------------------------------------------------
Current OS settings:
 - Host name not set
 - Sshd enabled not set
 - Root password is not set
 - Timezone not set
Are you happy with these settings (Y/N)? N
Host name:LNXFW1
Enable SSH daemon (Y/N)? Y
=== Change root password ===

Enter new root password:

Re-Enter it:
Select timezone (empty or prefix = list zones):
Selected: EST5EDT
Current OS settings:
 - Host name: 'LNXFW1'
 - Sshd enabled: 1
 - Root password is set
 - Timezone: EST5EDT
Are you happy with these settings (Y/N)? Y
```

Use the initial screen to:

- Set the StoneGate host name.
- Enable the SSH daemon.
- Create a root password.
- Set the correct time zone.

Configuring network interfaces

Next, configure network interfaces using the screen shown in Example 6-12 on page 130.

Example 6-12 Configure network interfaces screen

```
Step 2 of 3: Configure network interfaces
--------------------------------------------------------------------------------
Current network interfaces:
Id Iface   MAC             Type      Status
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
R)reprobe channels, P)rint channels,
A)dd driver, C)lear, M)gmt interface, N)IC id mapping, S)niff, D)one. A
Detected following channels:
1  lcs+ctc (0x05) devno: 0x0900 in use/reg: no/no
2  lcs+ctc (0x05) devno: 0x0901 in use/reg: no/no
3  qeth (0x10) devno: 0x7104 in use/reg: no/no
4  qeth (0x10) devno: 0x7105 in use/reg: no/no
5  qeth (0x10) devno: 0x7106 in use/reg: no/no
Select the device layer:
  iucv
  ctc
  qeth
  lcs
Your selection: qeth
```

To add a network interface, select option `A`. We configure the virtual HiperSockets Guest LAN interface by selecting the `qeth` device driver. The QETH driver supports HiperSockets virtual devices (Guest LANs), OSA Express Fast Ethernet (OSAX-FE), and Gigabit Ethernet (OSAX-GE) adapters in QDIO mode.

Note: The QETH driver as well as the QDIO driver are included in the 2.4.21+smp kernel.

Configure the interface as shown in Example 6-13 on page 131.

Example 6-13 Configure the QETH device

```
QETH for OSA-Express and Hipersockets channel device selected

Paramter syntax:
  <devname>,<read_devno>,<write_devno>[,<data_devno>][,<memusage>][,<port_no>][,<chksum_recv>]
Please refer to IBM document 'Linux for zSeries and S/390: Device Drivers and
Installation Commands' for more comprehensive syntax.
HiperSockets example: qeth1, 0x7c00, 0x7c01, 0x7c02, 4096, -1
OSA-Express example:
qeth2,0x500,0x501,0x502;add_parms,0x10,0x500,0x502,portname:1789

Parameters:
qeth0,0x7104,0x7105,0x7106;add_parms,0x10,0x7104,0x7106,portname:OSA7140,primary_router
You have given the following configuration:
-Module: qeth
-Parameters:qeth0,0x7104,0x7105,0x7106;add_parms,0x10,0x7104,0x7106,portname:OSA7140,primary_ro
ut

Is it correct (Y/N)? Y
```

Automatic device number assignment should not be used when configuring network device drivers. Instead, manually assign network device numbers.

Note: We illustrate the configuration of the QETH network device. The screens to configure iucv, ctc, and lcs devices prompt for the specific parameters required to configure those devices. Details about configuring Linux device drivers can be found in *Linux for zSeries and S/390 Device Drivers and Installation Commands*, LNUX-1303, available at:

http://www10.software.ibm.com/developerworks/opensource/linux390/docu/lx24jun03dd01.pdf

If the firewall needs to route to other networks not directly attached to a particular OSA card, the primary_router or the secondary_router parameters, or both, might be required.

Assigning network interfaces

Next, we assign the network interface card (NIC) IDs to each interface in the screen shown in Example 6-14 on page 132 by selecting option N.

Example 6-14 Defining NIC ID mapping screen

```
Current network interfaces:
Id Iface   MAC               Type      Status
-------------------------------------------------------------------------------
0 iucv0   - No HW address - link/slip unknown (mgmt)
1 ctc0    - No HW address - link/slip unknown
2 hsi0    - No HW address - link/ether unknown
3 eth1      00:06:29:6c:5d:e2 link/ether unknown
-------------------------------------------------------------------------------
R)reprobe channels, P)rint channels,
A)dd driver, C)lear, M)gmt interface, N)IC id mapping, S)niff, D)one: N
New NIC id for iucv0 (now it's 0)? 1
```

For each interface, define the NIC ID number. Select option D (Done) when all interfaces are configured.

Defining the management interface

Next, we define the management network interface by selecting option M, as shown in Example 6-15.

Example 6-15 Define management interface screen

```
Current network interfaces:
Id Iface   MAC               Type      Status
-------------------------------------------------------------------------------
0 iucv0   - No HW address - link/slip unknown (mgmt)
1 ctc0    - No HW address - link/slip unknown
2 hsi0    - No HW address - link/ether unknown
3 eth1      00:06:29:6c:5d:e2 link/ether unknown
-------------------------------------------------------------------------------
R)reprobe channels, P)rint channels,
A)dd driver, C)lear, M)gmt interface, N)IC id mapping, S)niff, D)one:
Management interface NIC id: 2
```

Select option D to complete the management interface configuration.

Contacting the Management Server

Next, the screen shown in Example 6-16 on page 133 opens. This is used to prepare for management contact.

Example 6-16 Prepare for management contact screen

```
Step 3 of 3: Prepare for management contact
--------------------------------------------------------------------------------
Current management contact settings:
Switch to initial configuration: No
 - Node IP address not set
 - Peer IP address not set
Perform initial contact: No
 - Management IP address not set
 - Management One-time password not set
 - Management fingerprint not set
Are you happy with these settings (Y/N)? N
```

Select `N` to change the default settings. If the defined management interface is a QDIO-type network, the screen shown in Example 6-17 opens. If the management interface is a point-to-point connection, the screen shown in Example 6-18 on page 134 opens.

Note: QDIO-type networks include OSA, OSA-Express, HiperSockets, and VM Guest LANs. Point-to-point networks include CTC and IUCV connections.

Example 6-17 Configure QDIO management interface screen

```
Enter data for switching to the initial configuration and/or
contacting the management server. Fields marked with * must be filled.
Firewall node:
  Node IP address:*
  Netmask:*
  Gateway to management:
  Use VLAN (Y/N)?
```

To configure a QDIO management interface, provide the following information:

- Node IP address
  This required parameter is the IP address of the firewall node used for management connections. This address should be the same as specified in the firewall element on the Management Server.
- Netmask
  Enter the required network mask for the node IP address.

- Gateway to management
  This parameter is the IP address of the default gateway to the Management Server. If the firewall engine and Management Server are on the same network, this can be left blank.
- Use VLAN
  Enter `Y` if VLAN tagging is used on the firewall engine management interface, and then enter the VLAN ID of the management interface. If no VLAN tagging is used, enter `N`.

> **Note:** Do not enter `Y` if VLAN tagging is used on the firewall, but not between the Management Server and this firewall engine instance. This parameter is used only to configure the VLAN on the management interface.

The screen to configure a point-to-point management interface is shown in Example 6-18.

Example 6-18 Configure point-to-point management interface screen

```
Enter data for switching to the initial configuration and/or
contacting the management server. Fields marked with * must be filled.
Firewall node:
  Node IP address:*
  Peer IP address*:
```

The required configuration parameters are:

- Node IP address
  Provide the IP address of the firewall node to be used for management connections. This should be the same address specified in the firewall element on the Management Server.
- Peer IP address
  Provide the peer IP address on the Management Server.

When the configuration of the management interface is complete, StoneGate attempts to establish an initial contact to the Management Server. At this point, a trust relationship is established between the engine and the Management Server. The screen to configure the initial contact is shown in Example 6-19 on page 135.

Example 6-19 Configure initial contact screen

```
Perform new initial contact (Y/N)? Y
Enter data required for contacting the management server.
Fields marked with * must be filled, others are optional.
  Management IP address:* 9.12.6.170
  One-time password:* xbDf5GhYtf
  Key fingerprint: 0D:32:1D:B2:03:54:DC:9E:A0:B7:18:7D:C5:0B:FD:2F
Are you happy with these settings (Y/N)? Y
```

Note: The required parameters for the initial contact can be retrieved in the management GUI client's Control Panel: Right-click the firewall element and select **Save Initial Configuration**.

The parameters to supply include:

- Management IP address
  Provide the IP address of the Management Server.
- One-time password
  Enter a 10 alphanumeric character, case-sensitive, one-time password. The password is engine specific and is only used for the initial contact to the Management Server.
- Key fingerprint
  Enter the 16 hexadecimal character Management Server key fingerprint. Values should be separated by colons.

Accept the configuration by entering `Y` when prompted. To complete the configuration, save the configuration when prompted by the screen shown in Example 6-20 on page 136.

Example 6-20 Contact Management Server

```
Your choices:
----------------------------------
R) Reconfigure
P) Print current settings
S) Save and exit
C) Cancel
----------------------------------
Your selection: S
SSH IPSEC version 4.1.1 build 53
Contacting management system...
Private key does not exist. Creating.
Generating RSA private key, 2048 bit long modulus
..............+++
........+++
unable to write 'random state'
e is 65537 (0x10001)
Contact succeeded.
Press Enter to continue



StoneGate s390 LNXFW4 console

LNXFW4 login:
```

The firewall engine then attempts to contact the Management Server. If initial management contact fails, the management interface can be reconfigured using the **`sg-reconfigure`** command.

Note: If you receive a "`connection refused`" error message when contacting the Management Server, ensure that the one-time password is correct and the Management Server IP address is reachable from the node.

After a successful Management Server contact, the firewall engine installation is complete and ready for the security policy upload from the management system. You can verify this from the management GUI client's Control Panel, where the firewall's status has changed from Unknown to Not Configured.

For more information, consult the *StoneGate Administrator's Guide*.

7

Using z/OS features in a Linux environment

In this chapter, we discuss of a number of z/OS features that can be used to enhance security in a Linux on zSeries environment. In addition, Linux for z/Series can provide a number of security functions for greater flexibility in a z/OS environment. We consider:

- z/OS HiperSockets Accelerator
- Authentication using IBM Tivoli Access Manager
- IBM Tivoli Access Manager WebSEAL
- Securing z/OS Web resources from Linux

7.1 z/OS HiperSockets Accelerator

z/OS Version 1Release 2 Communications Server introduced performance improvements for IP routing between HiperSockets (also known as *Internal Queued Direct Input/Output* or iQDIO) and Queued Direct I/O (QDIO) adapters referred to as HiperSockets Accelerator. Network traffic is concentrated on a single OSA-Express QDIO connection then accelerated over a HiperSockets link (bypassing the z/OS TCP/IP stack).

HiperSockets Accelerator can simplify the management of OSA-Express interfaces; network traffic to and from Linux guests is routed over internal path HiperSockets networks. A number of security measures usually required for physical networks can be avoided, resulting in a simpler, less labour-intensive configuration. Using HiperSockets Accelerator, a large number of Linux images in single CEC can access a OSA-Express adapter using only the number channel addresses required by a single z/OS image.

Note: Highly optimized routes from z/OS to the Linux images allow data to be routed through the HiperSockets devices with minimal processing by the TCP/IP stack, and without exhausting the channel addresses available to the OSA-Express. HiperSockets Accelerator requires minimal explicit configuration and runs in a single z/OS TCP/IP. Network traffic is routed by the device driver instead of transversing the entire z/OS TCP/IP stack.

HiperSockets Accelerator can decrease the need for many separate OSA-Express adapters and I/O cage slots, while increasing server scalability at lower cost when consolidating Linux, z/VM, and z/OS servers on zSeries.

HiperSockets Accelerators support Virtual IP Address (VIPA) takeover and takeback sysplex environment for high availability. Separate TCP/IP stacks with HiperSockets Accelerator can run in the same sysplex for redundancy. Each backup stack must have an equivalent connection through both HiperSockets and OSA-Express. VIPA takeover and takeback are supported only between z/OS systems and within the same sysplex. Figure 7-1 on page 139 illustrates an example HiperSockets Accelerator configuration for Linux guests.

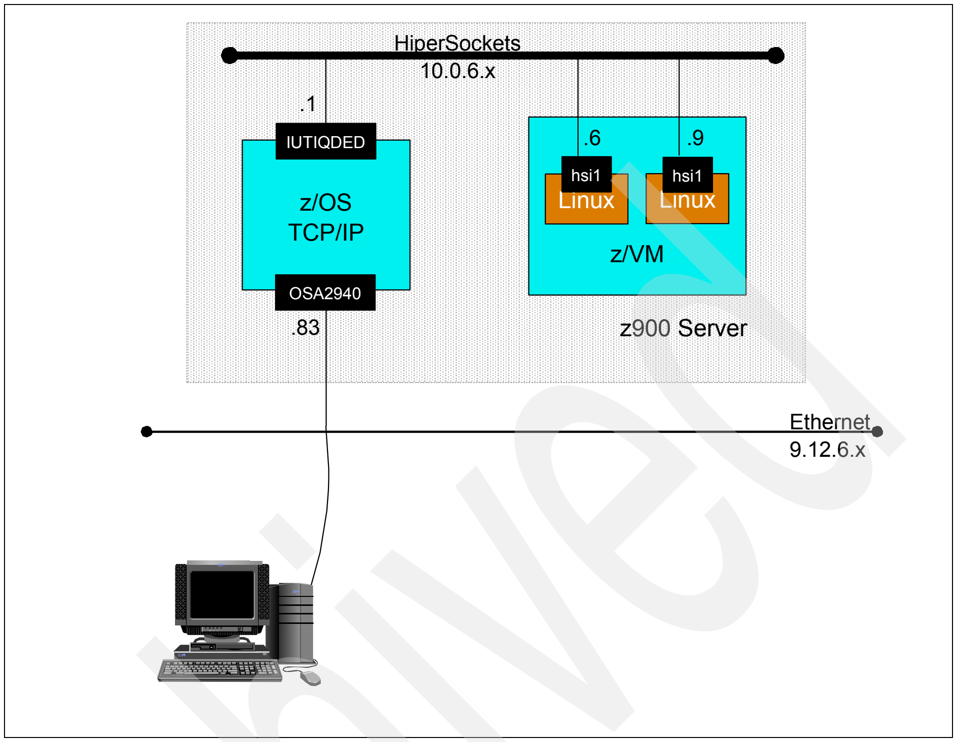


Figure 7-1 Using HiperSockets Accelerator for Linux guests

Note: Detailed discussion and setup information about HiperSockets Accelerator can be found in *Linux on IBM @server zSeries and S/390: Large Scale Linux Deployment*, SG24-6824, available at:

http://www.ibm.com/redbooks/abstracts/sg246824.html

7.2 IBM Tivoli Access Manager for e-business

IBM Tivoli Access Manager forms the basis for the IBM Tivoli Access Manager product suite. Delivered as an integrated solution, these products provide access control management solutions to centralize network and application security policies for e-business applications.

Note: IBM Tivoli Access Manager is the new name of the previously released Tivoli SecureWay® Policy Director. For users familiar with Tivoli SecureWay Policy Director, the *management server* is now referred to as the *Policy Director*. Details about Tivoli Access Manager for e-business can be found in *IBM Tivoli Access Manager for e-business*, REDP-3677, available at:

http://www.ibm.com/redbooks/abstracts/redp3677.html

7.3 Authentication using IBM Tivoli Access Manager

In this section, we discuss using Tivoli Access Manager for Linux user authentication against users defined to RACF running in a z/OS LPAR. The scenario is illustrated in Figure 7-2.

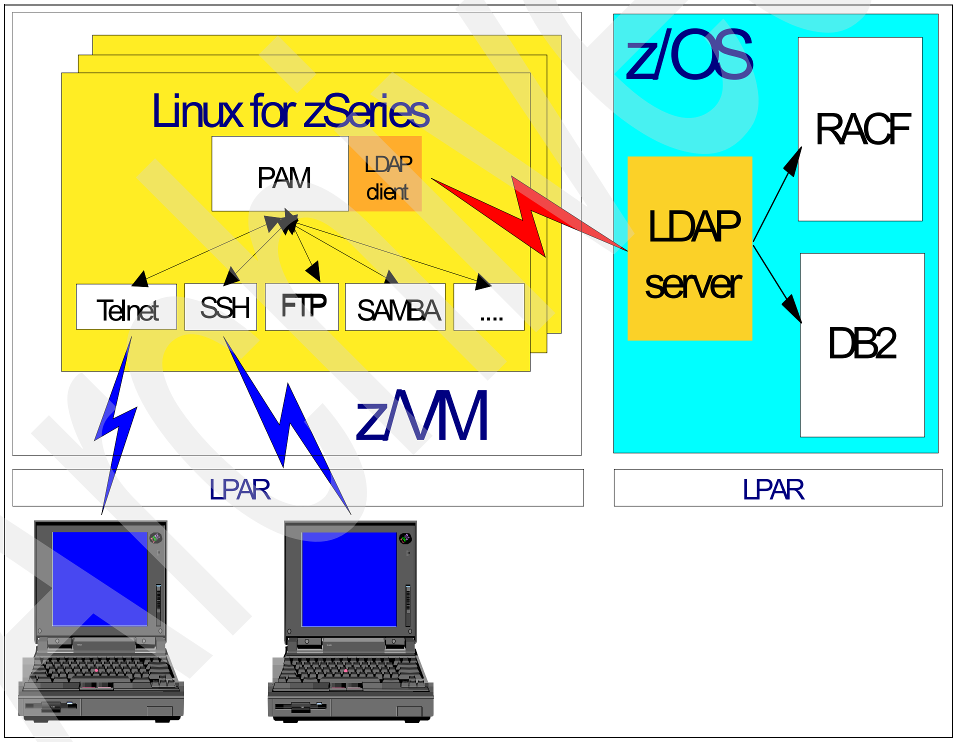


Figure 7-2 User authentication using Tivoli Access Manager

In this scenario, network clients (such as Telnet and SSH) are authenticated against users defined to an LDAP directory running on z/OS. Linux users are defined in the LDAP directory (stored in DB2®) on z/OS. However, no user passwords are stored in the LDAP directory. Instead, password authentication is performed against user passwords stored in RACF running on z/OS. Using the capabilities provided by PAM (discussed in 3.2, “Pluggable Authentication Modules” on page 31), we outline the procedure to enable Linux users against LDAP on z/OS.

Note: This procedure is similar to the steps outlined in *Securing Linux for zSeries with a Central z/OS LDAP Server (RACF)*, REDP-0221. Using Tivoli Access Manager enables users to change their passwords from Linux (using the `passwd` command).

7.3.1 Configuring LDAP on z/OS

We begin by configuring LDAP on z/OS using the `ldapcnf` utility:

1. Create a new LDAP configuration file.
   Copy the /usr/lpp/ldap/etc/ldap.profile file to a working directory such as /etc/ldap.
2. Customize the ldap.profile file for your system.
   Use the comments in the configuration file to determine appropriate values.

   Note: Some attributes in the ldap.profile file are required, but have no default value. Be sure to examine the entire file and assign values to all required variables. In “LDAP configuration parameters” on page 142, we describe several important parameters.

3. Run the `ldapcnf` command from z/OS UNIX.
   This generates a set of jobs in the MVS™ data set defined in the ldap.profile file.
4. Add the LDAP server started task to the system proclib.
   Copy the LDAP server started task procedure from the output data set into the system proclib. The default name for the LDAP started task is LDAPSRV.
5. Copy the PROGxxx file to the system parmlib.
6. Start RACF and its associated tasks.
   Run each job in the following sequence:
   a. RACF.
   b. APF.
   c. DBCLI (ensure that DB2 is started before submitting this job).
   d. PGRMCTRL (if required).

   Remember to check for a successful return code in each case.
7. Start the DBSPUFI task.
   Use DB2 SPUFI to submit the job.

8. Start the LDAP server.
   From SDSF, start the server using the command **`/s LDAPSRV`**.

The LDAP server is started when you receive the "`slapd is ready for requests`" message.

LDAP configuration parameters

Some important TDBM-specific LDAP configuration parameters include:

Database tdbm GLDRDMB
This parameter specifies that LDAP is to connect to DB2.

AdminDN "cn=LDAP Administrator"
This parameter specifies the LDAP administrator. To perform administrative tasks, clients and applications use this user name (and corresponding password) when binding to the LDAP server.

Suffix "o=itso"
Suffixes are required in the LDAP configuration file. At least one suffix is found at the end of each directory entry. Suffix names are determined by standards determined by the LDAP administrator and are used by applications accessing the directory. In our example, we choose the suffix name *o=itso*. The schema is loaded using the form "*cn=schema,o=itso*".

DSNAOINI
This parameter points to the MVS data set.

Servername
This parameter is defined using the name of the DB2 server.

Dbuserid
This parameter specifies the user ID used to connect to DB2. This must have both SAF authority and DB2 admin authority in order to connect to both RACF and to DB2.

Databasename LDAPDB
This parameter specifies the DB2 LDAP database name.

Some important SDBM-specific LDAP configuration parameters include:

Database sdbm GLDSDBM
This parameter specifies that LDAP is to connect to RACF.

AdminDN="racfid=admin,profiletype=user,sysplex=SYSPLX"
The parameter specifies that LDAP is to look up RACF users when binding. The profiletype attribute is required; the sysplex attribute can be assigned any value (we use the z/OS LPAR name). No password is required.

Suffix=sysplex=SYSPLX
SDBM back ends require the sysplex attribute be defined as the suffix.

Note: Ensure that a RACF subsystem ID is defined before attempting to bind to RACF.h

7.3.2 Modifying the z/OS LDAP schema

After starting the LDAP server, we next modify the LDAP schema to support Linux user authentication. We use our specific suffix and add some test user data the steps outlined here:

1. Customize the user schema using your specific suffix.
   Copy the following files from the /usr/lpp/ldap/etc directory to the /etc/ldap directory:
   - schema.user.ldif
   - schema.IBM.ldif

   Edit both files, changing the line "cn=schema,<suffix>" to specify the suffix defined in your ldap.profile configuration file.
2. Implement the modified LDAP schema.
   Use the **ldapmodify** command:

```
ldapmodify -h 9.12.6.16 -p 3389 -D "cn=LDAP Administrator" -w secret -f \
/etc/ldap/schema.user.ldif
ldapmodify -h 9.12.6.16 -p 3389 -D "cn=LDAP Administrator" -w secret -f \
/etc/ldap/schema.IBM.ldif
```

   Note: In this example, 9.12.6.16 is the IP address of the LDAP server. LDAP listens on port number 3389.

3. Add a suffix entry and test user to the LDAP directory.
   We create a suffix.ldif file, as shown in Example 7-1, to define the LDAP entries. Add these entries using the command:

```
ldapadd -h 9.12.6.16 -p 3389 -D "cn=LDAP Administrator" -w secret -f \
suffix.ldif
```

Example 7-1 The suffix.ldif file

```
dn: o=itso
objectclass: organization
objectclass:top
o: itso
dn: cn=test1,o=itso
objectclass: top
objectclass: ePerson
cn: test1
sn: user
```

To ensure that the LDAP is correctly configured, execute the command:

```
ldapsearch -h 9.12.6.16 -p 3389 -V 3 -s base -b " " "objectclass=*"
```

The expected response appears similar to the response shown in Example 7-2.

Example 7-2 Response

```
supportedcontrol=2.16.840.1.113730.3.4.2
supportedcontrol=1.3.18.0.2.10.2
supportedcontrol=1.3.18.0.2.10.10
supportedcontrol=1.3.18.0.2.10.11
supportedextension=1.3.6.1.4.1.1466.20037
namingcontexts=o=itso
namingcontexts=secAuthority=Default
namingcontexts=o=itsoracf
subschemasubentry=CN=SCHEMA,o=itso
supportedsaslmechanisms=EXTERNAL
supportedsaslmechanisms=CRAM-MD5
supportedsaslmechanisms=DIGEST-MD5
supportedldapversion=2
supportedldapversion=3
ibmdirectoryversion=z/OS V1R4
ibm-sasldigestrealmname=wtsc59oe
```

7.3.3 Enabling z/OS LDAP native authentication

To enable LDAP to bind against RACF, native authentication must be configured:

1. Customize the LDAP native authentication schema.
   Copy the /usr/lpp/ldap/etc/NativeAuthentication.ldif file to the /etc/ldap directory. Edit the file, changing the line "cn=schema,<suffix>" using the suffix defined in your ldap.profile configuration file.
2. Implement the native authentication schema.
   Use the command:

   ```
   ldapmodify -h 9.12.6.16 -p 3389 -D "cn=LDAP Administrator" -w secret -f \
   /etc/ldap/NativeAuthenication.ldif
   ```

3. Enable native authentication in LDAP.
   Add the following to the TDBM section of the LDAP configuration file (ldap.profile):

   ```
   useNativeAuth SELECTED
   "nativeAuthSubtree" o=itso
   nativeUpdateAllowed YES
   ```

4. Modify existing users to use native authentication.
   Add the ibm-nativeAuthentication object class and ibm-nativeId attribute to users defined in the schema. We create the nativeupdate.ldif file, as shown in Example 7-3, and modify the directory using:

```
ldapmodify -h 9.12.6.16 -p 3389 -D “cn=LDAP Administrator” -w secret -f
/etc/ldap/nativeupdate.ldif
```

Example 7-3 The nativeupdate.ldif file

```
dn: cn=test1, o=itso
changetype: modify
add: x
ibm-nativeId: test1
objectclass: ibm-nativeAuthentication
```

7.3.4 Installing Tivoli Access Manager Policy Director on Linux

The Policy Director requires installation of both the IBM Global Security Toolkit and LDAP client, as well as the Tivoli Access Manager packages for Linux for zSeries. We install all packages using the standard **rpm** command, as shown in Example 7-4.

Example 7-4 Installing Tivoli Access Manager and its prerequisites

```
# rpm -ivh gsk5bas-5.0-5.46.s390.rpm \        1
           ldap-clientd-4.1-1.s390.rpm \      2
           PDMgr-4.1.0-2.s390.rpm \           3
           PDRTE-4.1.0-2.s390.rpm             4
```

This installs the:

- Global Security Toolkit
- LDAP client
- Tivoli Access Manager Policy Director
- Policy Director runtime

Note: These packages are distributed as part of Tivoli Access Manager.

7.3.5 Configuring Tivoli Access Manager for Linux

After a successful installation, we use the **pdconfig** command to configure the Tivoli Access Manager runtime and Policy Director.

Configuring the Tivoli Access Manager runtime

Configure the Tivoli Access Manager runtime to point to the z/OS LDAP server. When prompted, use the parameters specified in Example 7-1.

Table 7-1 Tivoli Access Manager runtime configuration parameters

Parameter	Description
Registry Selection	The type of remote registry; choose **LDAP registry**.
LDAP server hostname	The TCP/IP address of the LDAP server. Use the IP address or name of the LDAP server running on z/OS.
LDAP server port	The port number of z/OS LDAP server.

Configuring the Tivoli Access Manager Policy Director

Next, configure the Policy Director using the parameters shown in Example 7-2.

Table 7-2 Tivoli Access Manager Policy Director configuration parameters

Parameter	Description
LDAP administrative user DN	The distinguished name (DN) of the LDAP server. In our example, we use **LDAP Administrator**.
LDAP administrative user password	The password LDAP administrator. The example uses **secret**.
Enable SSL communications	To enable secure SSL communications, specify **yes**. In the example, we use HiperSockets for secure TCP/IP networking and specify **no**.[a]

a. For guidelines about using SSL, refer to *Securing Linux for zSeries with a Central z/OS LDAP Server (RACF)*, REDP-0221, available at:
http://www.ibm.com/redbooks/abstracts/redp0221.html

The configuration is stored in the /opt/PolicyDirector/etc directory in three files:

- pd.conf
- ivmgrd.conf
- ldap.conf.

7.3.6 Enabling Linux LDAP user authentication

To enable LDAP user authentication in Linux, we use the capability provided by Name Service Switch (NSS). We configure PAM to use LDAP authentication and then configure NSS to point to the z/OS LDAP server.

Configuring PAM for LDAP authentication

Modify the /etc/pam.d configuration file for each service that uses LDAP authentication. Example 7-5 illustrates the configuration file used for Telnet login.

Example 7-5 The /etc/pam.d/login file for LDAP authentication

```
#%PAM-1.0
auth     required      pam_nologin.so
auth     sufficient    pam_unix_auth.so                                 1
auth     required      pam_ldap.so        try_first_pass                2
account  sufficient    pam_localuser.so                                 3
account  required      pam_ldap.so                                      4
password required      pam_pwcheck.so     nullok
password required      pam_ldap.so        use_first_pass use_authok     5
password required      pam_unix.so        nullok use_first_pass use_authtok
session  required      pam_unix_session.so
session  required      pam_limits.so
session  required      pam_env.so
session  optional      pam_mail.so
```

These changes include:

1. The pam_unix_auth.so module implements auth type user authentication and is capable of supporting NSS. The sufficient control flag ensures success if a locally defined user (in the /etc/passwd file) is authenticated.
2. If auth type authentication against a local does not succeed, the required pam_ldap.so module attempts to authenticate using LDAP.
3. The pam_localuser.so module performs account authentication for local users. The sufficient control flag ensures success in the event a local user is authenticated.
4. In the event account type authentication does not succeed against a local user, the pam_ldap.so module authenticates against LDAP.
5. The pam_ldap.so module is capable of authentication password update requests against an LDAP directory.

Tip: Although it is possible to maintain all user information in the remote LDAP directory, it is a good idea to define at least one user with superuser authority locally. This allows host access in the event LDAP is unavailable.

Linux users usually have their home directory (typically the /home/*userid* directory) created when the user ID is defined. When using LDAP authentication, users are remotely defined (and have no home directory created on the local host). Two possibilities exist to avoid manual creation of a home directory for each LDAP defined user:

- The user home directory can be located on a network file server (for example, an NFS-mounted file system).
- The home directory can be automatically created when a user first logs in:
  - For login services except SSH, add the pam_mkhomedir.so module to the PAM configuration file for the service (/etc/pam.d/login, for example):

    ```
    session required pam_mkhomedir.so skel=/etc/skel/ umask=0077
    ```

    If a user authenticates and no home directory exists, the home directory is created in /home. The umask=0077 parameter causes the directory permission to be set to 700. The home directory is constructed from the skeletal files found in the /etc/skel directory.
  - Beginning with OpenSSH Version 3.3, automatic creation of a user home directory using pam_mkhomedir.so is no longer supported due to a security modification in SSH. You can use the make_home_dir replacement for pam_mkhomedir.so. The make_home_dir package is available at:

    http://www.trustsec.de/soft/oss

Configuring NSS

To complete the process, we configure NSS to use LDAP for user and group authentication. We modify the /etc/nsswitch NSS configuration file, as shown in Example 7-6.

Example 7-6 The /etc/nsswitch file for LDAP authentication

```
...
passwd: ldap compat
group: ldap compat
shadow: ldap compat
```

Next, we create the /etc/nss_ldap.conf file, as shown in Example 7-7.

Example 7-7 The /etc/nss_ldap.conf file for LDAP authentication

```
# Your LDAP server. Must be resolvable without using LDAP.
host 9.12.6.16:3389
# The distinguished name of the search base
base o=itso
uri ldaps://9.12.6.16:3389
ldap_version 3
```

The `base` parameter uses the suffix defined to the z/OS LDAP server. The `uri` parameter points to the LDAP server running on z/OS.

7.4 IBM Tivoli Access Manager WebSEAL

IBM Tivoli Access Manager WebSEAL is a security manager for Web-based resources. WebSEAL is a high performance, multithreaded Web server that applies fine-grained security policy to the protected Web object space. WebSEAL can provide single sign-on solutions and incorporate back-end Web application server resources into its security policy.

In addition to authenticating Linux users, we can use WebSEAL in conjunction with LDAP running on z/OS to protect Web-based resources. This scenario is depicted in Figure 7-3.

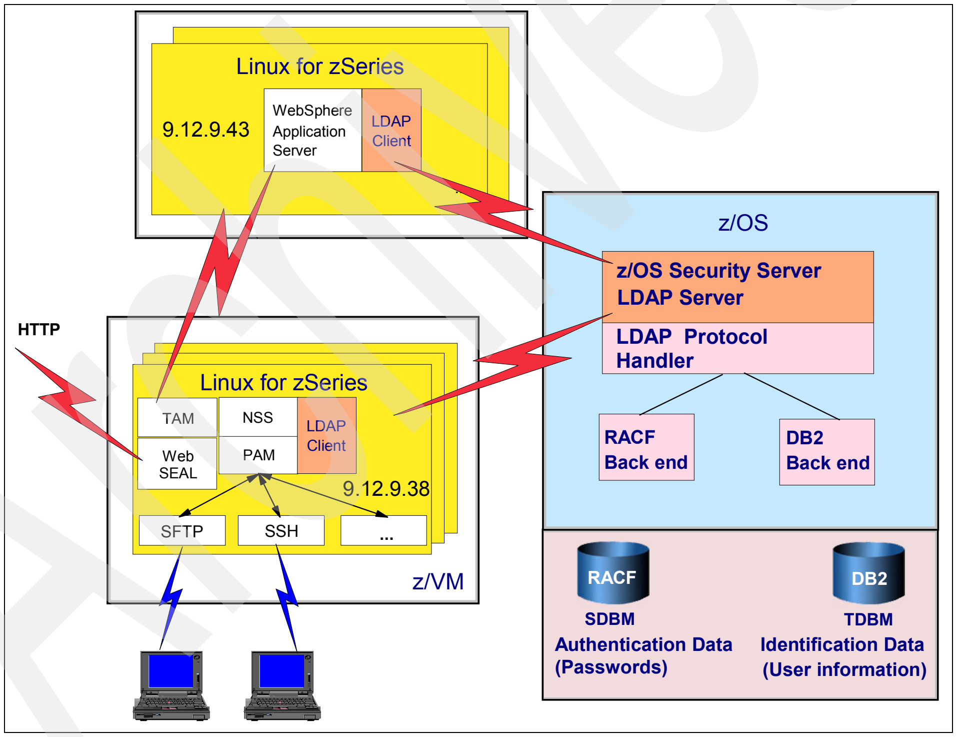


Figure 7-3 Authenticating users against a z/OS RACF database

In this scenario, we use a single Tivoli Access Manager instance. For scalability and high availability, several Tivoli Access Manager/Linux instances are required.

A request spraying mechanism distributes the workload across all instances (for example, a round-robin DNS or Linux clustering using LVS).

7.4.1 Configuring WebSEAL

To configure WebSEAL:

1. First install the PDWeb-PD-4.1.0-2.s390.rpm package from Tivoli Access Manager using the command:

   ```
   rpm -ivh PDWeb-PD-4.1.0-2.s390.rpm
   ```

2. Configure WebSEAL using the **pdconfig** command. The Access Manager for e-business setup menu is shown in Example 7-8.

Example 7-8 Access Manager setup menu

```
Access Manager for e-business Setup Menu
1. Configure Package
2. Unconfigure Package
3. Display Configuration Status
x. Exit
Please select the menu item [x]: 1
```

3. Select option 1 to navigate to the Access Manager configuration menu shown in Example 7-9.

Example 7-9 Access Manager configuration menu

```
Access Manager for e-business Configuration Menu
1. Access Manager WebSEAL Configuration
x. Return to Access Manager for e-business Setup Menu
Please select the menu item [x]: 1
```

4. Select option 1 to navigate to the Access Manager WebSEAL setup menu shown in Example 7-10.

Example 7-10 Access Manager WebSEAL setup menu

```
Access Manager WebSEAL for e-business Setup Menu
1. Configure
2. Display Configuration Status
x. Exit
Please select the menu item [x]: 1
```

5. Select option 1. You are prompted for a Tivoli Access Manager Administrator (sec_master) password. We use the password *secret*.

Note: If you repeatedly enter an incorrect password, you might get an error:

```
Error: This account has been temporarily locked out due to too many
failed login attempts.
```

Obtain the correct password, wait five minutes for the lock to clear, and then restart the **pdconfig** command.

6. You are prompted to enable SSL communication between the WebSEAL server and the LDAP server. Type y (yes) or n (no) to enable SSL communication between the Access Manager server and the LDAP server:

   ```
   Do you want to enable SSL communication between the Access Manager
   server and the LDAP server (y/n) [Yes]?
   ```

Note: In this redbook we do not use SSL, because we are connecting to z/OS using an internal HiperSockets network. For additional security, you can use SSL to encrypt all network traffic (even on an internal network). Refer to *Securing Linux for zSeries with a Central z/OS LDAP Server (RACF)*, REDP-0221 for a full description and guidelines about how to encrypt the LDAP communication between your Linux virtual machines and the z/OS LDAP server.

The WebSEAL configuration is stored in the /opt/pdweb/etc/webseald.conf file. To use Tivoli Access Manager WebSEAL as a security front end requires the creation of a *junction* (a specification used to redirect Web requests to another URL). Junctions are created using the **pdadmin** command.

If your back-end application servers are configured for high availability, the WebSEAL junctions can be configured to point to several hosts. To assure unified service, these hosts should be configured with exactly the same services.

7.4.2 Creating the WebSEAL junctions

Using the **pdadmin** command, we define two WebSEAL junctions:

- One connects to the WebSphere HTTP-catcher (port 9080).
- One connects to the WebSphere administration server (port 9090).

Example 7-11 on page 152 illustrates the process to create the two junctions.

Example 7-11 Configuring Tivoli Access Manager using the pdadmin command

```
# pdadmin
pdadmin> login                                                    1
Enter User ID: sec_master                                         2
Enter Password:
pdadmin> server task webseald-lnxsu4.itso.ibm.com create -t tcp \      3
         -h 9.12.9.43 -p 9080 /lnxwas
pdadmin> server task webseald-lnxsu4.itso.ibm.com create -t tcp \      4
         -h 9.12.9.43 -p 9090 /wasadm
```

In the example, we:

1. Log in to the WebSEAL administration server.
2. Specify administration user ID.
   We use the default `sec_master` user ID.
3. Create a junction to the WebSphere HTTP-catcher.
   The WebSEAL server is `webseald-lnxsu4.itso.ibm.com`, and `9.12.9.43` is the IP address of the WebSphere Application Server. The WebSphere HTTP-catcher listens on port `9080`, and `lnxwas` is the junction name.
4. Create a junction to the WebSphere administration server.
   The WebSphere administration server listens on port `9090`, and the junction is named `wasadm`.

After the junctions are configured, the WSsamples application can be accessed using the URL:

```
http://9.12.9.38/lnxwas/WSsamples
```

Administration functions are accessed using the URL:

```
http://9.12.9.38/wasadm/admin
```

When accessing these URLs, WebSEAL prompts for a user ID and password. The user ID and password is forwarded to the LDAP server on z/OS and verified against the RACF database on z/OS. With this configuration, expired RACF passwords can be changed from Linux.

7.4.3 Configuring the WebSphere Application Server

To further protect the administrator function of WebSphere, we enable global security on the WebSphere Application Server itself. We use an LDAP server and RACF running on z/OS to secure access to the admin console function by the wasadmin user.

We modify the security settings of the WebSphere Application Server. We point to the z/OS LDAP server and enable Global Security for the WebSphere server:

1. From the Administrative Console for WebSphere, in the **User Registries** → **LDAP** → **General Properties** menu, provide:
   - Server User ID parameter (cn=wasadmin,o=itso)
   - Server User Password parameter (secret)
   - Select the **Type** parameter of SecureWay
   - Host parameter (9.12.6.16)
   - Port parameter (3389)
   - Base Distinguished Name parameter (o=itso)
   - Bind Distinguished Name parameter (cn=ldap administrator)
   - Bind Password (secret)

   Note: Our example values are shown in parentheses.

   Keeping the remaining default settings, click the **Apply** button, and click **Save** to save the configuration.
2. In the **Security** → **Global Security** → **General Properties** menu:
   - Choose the **Enabled** selection.
   - From the Active Authentication Mechanism menu, select **LTPA**.
   - From the Active User Registry menu, select **LDAP**.

   Click the **Apply** button, click **Save** to save the configuration, and restart the WebSphere Application Server.

This setup ensures that WebSEAL protects access to WebSphere itself. Access to the administrator functions is secured by a WebSphere security function. Both cases authenticate against LDAP and RACF running on z/OS. Direct access to the WebSphere HTTP-catcher port (9080) and the administrator port (9090) must be blocked. We recommend that you use a firewall between WebSEAL and WebSphere that allows only connections between WebSEAL and the two WebSphere ports.

7.5 Securing z/OS Web resources from Linux

We can use Tivoli Access Manager secure access to a WebSphere Application Server running on z/OS. Figure 7-4 on page 154 illustrates the configuration.

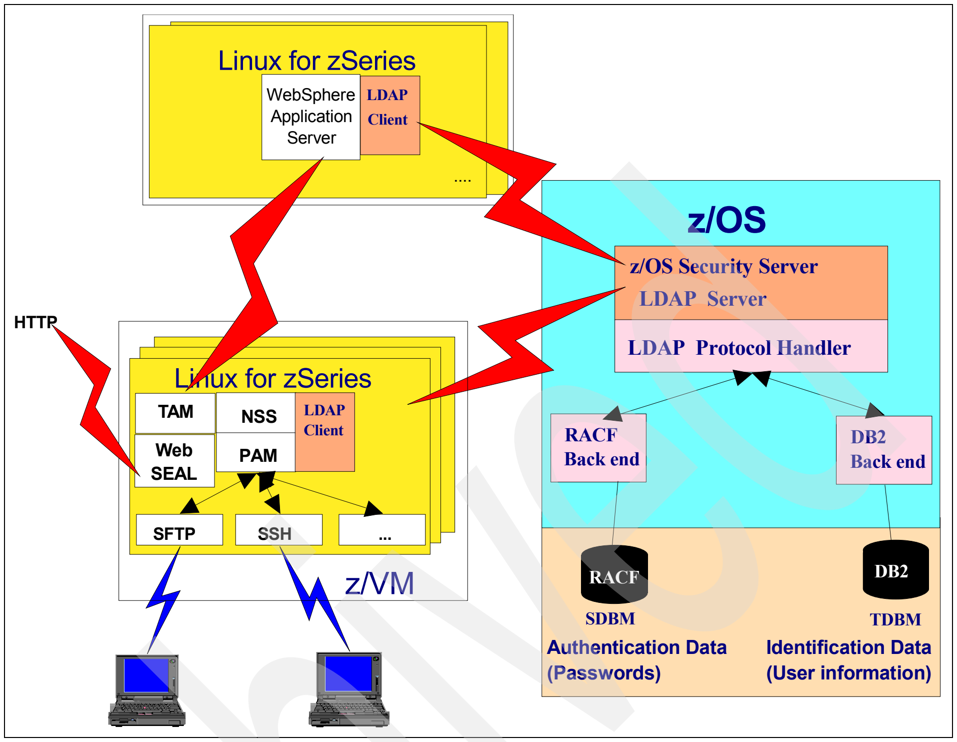


Figure 7-4 Securing WebSphere on z/OS using Tivoli Access Manager

Incoming HTTP requests to the Linux guest are intercepted by Tivoli Access Manager. We define two junctions:

- One connects to the z/OS WebSphere HTTP server (port 80).
- One connects to the z/OS WebSphere Application Server (port 9087).

In Example 7-12, we use the **pdadmin** command to define the junctions.

Example 7-12 Defining junctions to WebSphere on z/OS

```
# pdadmin
pdadmin> login
Enter User ID: sec_master
Enter Password:
pdadmin> server task webseald-lnxsu4.itso.ibm.com create -t tcp -h 9.12.6.16 -p 80 /test390
pdadmin> server task webseald-lnxsu4.itso.ibm.com create -t tcp -h 9.12.6.16 -p 9087 /was390
```

The WebSEAL server is `webseald-lnxsu4.itso.ibm.com`, and `9.12.6.16` is the IP address of the z/OS WebSphere Application Server. The WebSphere HTTP server listens on port 9080, and test390 is the junction name. The WebSphere Application Server listens on port 9087, and was390 is the junction name.

We access the z/OS WebSphere server from WebSEAL running on Linux for zSeries (IP address 9.12.9.38). For example, to execute the IBMEBizWeb/EJBCaller application, we use the URL:

```
http://9.12.9.38/was390/IBMEBizWeb/EJBCaller
```

WebSEAL prompts for a user ID and password that is verified against LDAP and RACF running on z/OS.

Related publications

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

IBM Redbooks

For information about ordering these publications, see "How to get IBM Redbooks" on page 160. Note that some of the documents referenced here may be available in softcopy only.

- *Linux for S/390*, SG24-4987

  `http://www.ibm.com/redbooks/abstracts/sg244987.html`
- *Linux on IBM @server zSeries and S/390: Distributions*, SG24-6264

  `http://www.ibm.com/redbooks/abstracts/sg246264.html`
- *Linux on IBM @server zSeries and S/390: ISP/ASP Solutions*, SG24-6299

  `http://www.ibm.com/redbooks/abstracts/sg246299.html`
- *Linux on IBM @server zSeries and S/390: System Management*, SG24-6820

  `http://www.ibm.com/redbooks/abstracts/sg246820.html`
- *Linux on IBM @server zSeries and S/390: Large Scale Linux Deployment*, SG24-6824

  `http://www.ibm.com/redbooks/abstracts/sg246824.html`
- *zSeries HiperSockets*, SG24-6816

  `http://www.ibm.com/redbooks/abstracts/sg246816.html`
- *Securing Linux for zSeries with a Central z/OS LDAP Server (RACF)*, REDP-0221

  `http://www.ibm.com/redbooks/abstracts/redp0221.html`
- *IBM Tivoli Access Manager for e-business*, REDP-3677

  `http://www.ibm.com/redbooks/abstracts/redp3677.html`
- *Linux on IBM @server zSeries and S/390: TCP/IP Broadcast on z/VM Guest LAN*, REDP-3596

  `http://www.ibm.com/redbooks/abstracts/redp3596.html`

- *Linux on IBM @server zSeries and S/390: VSWITCH and VLAN Features of z/VM 4.4*, REDP-3719

  http://www.ibm.com/redbooks/abstracts/redp3719.html

Other publications

These publications are also relevant as further information sources:

- *z/VM V4R4.0 CP Command and Utility Reference*, SC24-6008
- *z/VM V4R4.0 CP Planning and Administration*, SC24-6043
- *z/VM V4R4.0 Directory Maintenance Facility Function Level 410 Tailoring and Administration Guide*, SC24-6024
- *z/VM V4R4.0 System Operation*, SC24-6000
- *z/VM V4R4.0 Running Guest Operating Systems*, SC24-5997
- *z/VM V4R4.0 Virtual Machine Operation*, SC24-6036
- *z/VM V4R4.0 General Information*, GC24-5991
- *Linux for zSeries and S/390 Device Drivers and Installation Commands*, LNUX-1303
- Zwicky, et al., *Building Internet Firewalls*, O'Reilly & Associates, 2000, ISBN 1565928717
- Barrett, Daniel J. and Richard E. Silverman, *SSH, The Secure Shell: The Definitive Guide*, O'Reilly & Associates, 2001, ISBN 0596000111
- Viega, John, et al., *Network Security with OpenSSL*, O'Reilly & Associates, 2002, ISBN 059600270X
- Osisek, D.L., et al., "ESA/390 interpretive-execution architecture, foundation for VM/ESA," *IBM Systems Journal*, Vol 30, No 1, 1991
- Northcutt, Stephen and Judy Novak, *Network Intrusion Detection*, Que, 2002, ISBN 0735712654
- Barrett, Daniel J., et al., *Linux Security Cookbook*, O'Reilly & Associates, 2003, ISBN 0596003919
- Mann, S., et al., *Linux System Security: The Administrator's Guide to Open Source Security Tools*, Prentice Hall, 2002, ISBN 0130470112
- Skoudis, Ed, *Counter Hack: A Step-by-Step Guide to Computer Attacks and Effective Defense*, Prentice Hall, 2001, ISBN 0130332739
- McCarty, Bill, *Red Hat Linux Firewalls*, John Wiley & Sons, 2002, ISBN 0764524631

Online resources

These Web sites and URLs are also relevant as further information sources:

- IBM developerWorks Linux for zSeries and S/390 home page

  http://www-124.ibm.com/developerworks/oss/linux390/index.shtml
- Altmark, Alan and Cliff Laking, *z/VM Security and Integrity*

  http://www.ibm.com/servers/eserver/zseries/library/techpapers/gm130145.html
- Stonesoft home page

  http://www.stonesoft.com/
- OpenSSH home page

  http://www.openssh.com
- FreeS/WAN home page

  http://www.freeswan.org
- Fraser, B., *RFC2196: Site Security Handbook*

  http://www.ietf.org/rfc/rfc2196.txt?number=2196
- Linux-PAM documentation

  http://www.kernel.org/pub/linux/libs/pam/
- Sudo home page

  http://www.courtesan.com/sudo/
- Linux Security Modules home page

  http://lsm.immunix.org
- Linux Security home page

  http://www.linuxsecurity.com/
- *Linux Administrator's Security Guide* by Kurt Seifried

  http://www.seifried.org/lasg/
- Wright, Chris, et al., "Linux Security Modules: General Security Support for the Linux Kernel" *2002 USENIX Security Symposium*

  http://www.usenix.org/events/sec02/wright.html
- Security-Enhanced Linux home page

  http://www.nsa.gov/selinux/
- VLAN Information

  http://net21.ucdavis.edu/newvlan.htm

- IBM @server zSeries partitioning certification

  http://www.ibm.com/servers/eserver/zseries/security/certification.html
- Certification Report from the German Federal Office for information technology security

  http://www.bsi.bund.de/zertifiz/zert/reporte/0179a.pdf

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

IBM Redbooks

Linux on IBM @server zSeries and S/390: Best Security Practices

(0.2”spine)
0.17”<->0.473”
90<->249 pages

IBM®

Linux on IBM @server zSeries and S/390: Best Security Practices

Redbooks

Security topics for z/VM

Securing Linux guests

Firewalls on Linux for zSeries

This IBM Redbook discusses best security practices for running Linux as a z/VM guest on IBM @server zSeries and S/390 machines. This publication is intended for system administrators and IT architects responsible for deploying secure Linux servers running under z/VM. We consider both z/VM and Linux security topics.

We examine the unique security and integrity features zSeries offers for consolidating a large number Linux servers under z/VM. We discuss virtual machine isolation and command privileges assigned to VM guests. Security configuration options for z/VM Version 4.4 are explained.

In this book, we also discuss Linux security topics. We examine options for hardening a Linux installation. Securing Linux network traffic using Secure Sockets Layer and Secure Shell is considered. We look at implementing a virtual private network using FreeS/WAN. Commercial firewall technology and implementation using the StoneGate firewall for zSeries is discussed. We examine using IBM Tivoli Access Manager in conjunction with an LDAP server running on z/OS to authenticate Linux users against a RACF running on z/OS.

SG24-7023-00 ISBN 0738497665