IBM

AIX and Linux Interoperability

- Effective centralized user management in AIX 5L and Linux environments
- Sharing files and printers between AIX 5L and Linux systems
- Learn interoperable networking solutions

Abhijit Chavan
Dejan Muhamedagic
Jackson Afonso Krainer
Janethe Co
KyeongWon Jeong

Redbooks

ibm.com/redbooks

International Technical Support Organization

AIX and Linux Interoperability

April 2003

SG24-6622-00

Note: Before using this information and the product it supports, read the information in “Notices” on page xi.

First Edition (April 2003)

This edition applies to IBM @server pSeries and RS/6000 Systems for use with the AIX 5L for POWER Version 5.2 Operating System, Program Number 5765-E62, and is based on information available in November 2002.

Contents

Figures

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:
This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

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

Affinity™	eServer™	pSeries™
Approach®	Home Director™	Redbooks™
AIX 5L™	IBM®	Redbooks (logo) ™
AIX®	iSeries™	RS/6000®
CT™	NetView®	SecureWay®
Domino™	Notes®	SP™
DB2®	OS/2®	Tivoli®
@server ™	Perform™	XT™
@server™	PowerPC®	zSeries™

The following terms are trademarks of other companies:

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

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.

C-bus is a trademark of Corollary, 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.

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

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

Preface

This IBM® Redbook discusses interoperability in terms of UNIX to UNIX cross platform data sharing and user/system management. This redbook also demonstrates the similarities and differences between the AIX® and Linux operating systems.

This redbook is intended to help IT specialists who have AIX systems and Linux systems in their environments understand how to integrate and optimize AIX systems in a Linux environment and share AIX resources with Linux systems. We have focused our descriptions on the following areas:

- User management
  - Identification and authentication
  - Centralized user management
  - Single sign-on
- Networking
  - Networking services
  - Sendmail
  - NFS
- Data sharing
  - Samba file and print server
  - File systems and data archiving
  - Printer sharing
- System management
  - Security
  - System Administration Application
  - Linux for IBM @server pSeries™ and RS/6000®

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, Austin Center.

KyeongWon Jeong is a Consulting I/T Specialist at the International Technical Support Organization, Austin Center. He writes extensively on AIX and education materials and teaches IBM classes worldwide on all areas of AIX. Before joining the ITSO four years ago, he worked in IBM Global Learning Services in Korea as a Senior Education Specialist and was a class manager of all AIX classes for customers and interns. He has many years of teaching and development experience. He is an IBM Certified Advanced Technical Expert - RS/6000 AIX.

Abhijit Chavan is a Systems Engineer at GDI-Infotech, Mumbai, India. He has two years of experience in UNIX networking and systems programming. His areas of expertise include Linux and UNIX system administration, network management and monitoring, security, and Linux on LPAR. He holds a Bachelor's degree in Computer Engineering from the University of Mumbai.

Dejan Muhamedagic is currently with IBM Austria. He has ten years of experience with UNIX and TCP/IP networks. In the last couple of years, his main interest has been security. His latest project was LDAP authentication in AIX and Linux environments.

Jackson Afonso Krainer is a support analyst at HSBC Bank in Brazil. He holds an engineer's degree in Industrial Electrical Engineering from Federal Center of Technological Education (CEFET-PR), and a post graduation degree in Telecommunications from Pontifical Catholic University of Paraná at Curitiba, Brazil. He has five years of experience in Internet technologies. His areas of expertise includes UNIX servers security, communications security, LDAP, and cryptography.

Janethe Co is a System Services Representative at IBM Philippines. She holds a Bachelor of Science Degree in Electronics and Communications Engineer from De La Salle University, Manila. She has three years of experience in supporting pSeries and RS/6000 servers. Her areas of expertise include AIX, Linux, Security, and Firewall. She is a co-author of a previous redbook, *Linux Applications on pSeries*, SG24-6033.

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

Keigo Matsubara
International Technical Support Organization, Austin Center

Julianne Frances Haugh, Drew Walters, Yantian (Tom) Lu, Ut Le, Ufuk Celikkan, Mark McConaughy, Frank Irwin
IBM Austin

Ling Tai and Larry Knain
IBM Raleigh

Jan-Rainer Lahmann
IBM Germany

Bjorn Roden
Pulsen Systems, Sweden

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. JN9B Building 003 Internal Zip 2834
  11400 Burnet Road
  Austin, Texas 78758-3493

1

Identification and authentication

Identification and authentication are real world processes, but it is not always easy to recognize them. For example, it is enough for our friends and family just to see us or hear our voice over the phone to establish unmistakably who we are. There seems to be no authentication involved and yet they do subconsciously verify our identity. Another example is proving our identity to authorities by showing an ID with a photo. A clerk then tries to verify that the ID is ours by comparing a photo with our face and that it has not been forged.

UNIX users (usually) represent real world subjects (us). A UNIX system decides what we are authorized to do based on our IDs. Every UNIX user name maps to a UNIX user ID (UID). Identification in computing is a process in which a user presents his user name. Authentication is establishing that the user is who she claims to be, which the user proves typically by typing her password.

In this chapter, we talk about identification and authentication mechanisms of AIX 5L™ Version 5.2 and Linux, in which ways they are similar and different, and how to make the best use of them.

1.1 User security mechanisms

Traditional UNIX security depends on a single map file known as /etc/passwd. Probably just about any UNIX administration basics class begins with the description of this file, so we skip a detailed explanation.

Historically, security mechanisms were developed in a race between system designers and programmers on the one side and “crackers” on the other. In the beginning, our passwords were present in /etc/passwd in a special form called password hash. A password hash is a function that does not have an inverse function and, for all practical purposes, it is impossible to reverse its result. Hence, it was deemed safe to keep password hashes in a world readable file. In time, the computers were becoming faster and mischievous users more skilled. Password hashes had to be moved out from /etc/passwd. The new way of storing passwords is usually called password shadowing, because the now defunct password field in /etc/passwd is shadowed by a more or less elaborate scheme for password storage.

Both AIX and Linux have password shadowing and, by default, rely on /etc/passwd for identification. Other than that, their native identification and authentication mechanisms are rather different. Linux supports Name Service Switch (NSS) modules that enable other identification methods and PAM as an authentication mechanism. AIX has its own loadable authentication modules, which support both identification and authentication methods.

1.1.1 AIX security

The /etc/security/passwd file is the AIX version of the password shadow. The /etc/security/user file contains user attributes that supplement the traditional set in /etc/passwd. Every user has a stanza that defines additional attributes. To keep the file changes and the size minimal, only the attributes differing from the default stanza are listed.

AIX loadable authentication modules framework

The AIX loadable authentication modules framework has a well defined API with two sets of function calls: One for identification and the other for authentication. A module designer may choose to implement either one of these. For example, Kerberos is only an authentication method. Therefore, the AIX Kerberos module implements just the authentication functions. System managers may combine modules in an effective manner to bring together various authentication and identification methods.

Loadable authentication modules are configured in /usr/lib/security/methods.cfg. For example, the following stanzas describe one identification and authentication method:

```
LDAP:
   program = /usr/lib/security/LDAP

KRB5:
   program = /usr/lib/security/KRB5
   options = authonly

KERBEROS:
   options = auth=KRB5,db=LDAP
```

The KERBEROS method specifies that the user and group information is available through LDAP and the authentication mechanism is Kerberos. Use the BUILTIN reserved keyword for the db option to refer to /etc/passwd and the /etc/security files.

The SYSTEM attribute in /etc/security/user specifies which way the user is authenticated by referencing stanzas from /usr/lib/security/methods.cfg. This implies that authentication can be controlled on a per user basis, which is a rather rare feature. To set the authentication on the host level, you should just change the SYSTEM attribute of the default stanza. Furthermore, it is possible to make quite complex authentication configurations using the SYSTEM grammar:

```
joe:
   SYSTEM = "LDAP or LDAP[unavail] and compat"
```

This example says that joe should be authenticated using the LDAP authentication method, but in case the LDAP service is not available, he can authenticate against the local files just as well. A complete description of the SYSTEM grammar is in the /etc/security/user file reference.

Another related attribute in /etc/security/user is registry, which specifies where the user's information is administered. Thus, a complete stanza for user joe in /etc/security/user may look like this:

```
joe:
   SYSTEM = KERBEROS
   registry = KERBEROS
```

Note that the user administration **chuser**, **mkuser**, and **rmuser** commands may not be well suited to some user registries, depending on the site's security policy. This is because these commands assume superuser privileges that may not be appropriate if users are administered on a site level.

1.1.2 Linux security

Linux keeps shadow passwords in the /etc/shadow map. It contains users' passwords along with some other useful data. This is a much safer solution than having passwords in /etc/passwd, because the file is readable only by the root user. Furthermore, it is possible to apply a password policy using other fields stored in the shadow map. Here is the list of all available attributes:

- Date the password has been changed
- Minimum password age
- Maximum password age
- How much time before the password expiration should the user be warned
- How much time after password has expired should the account be disabled
- Date the account has been disabled

All values are expressed in days and counted, where applicable, since January 1, 1970. The chage utility may be used to manipulate the values.

Shadow passwords is the standard Linux authentication method. However, contemporary Linux distributions rarely use it directly for authentication purposes, but rather authenticate through the PAM authentication mechanism. We discuss PAM in more detail in the next section.

The name service switch (NSS)

The name service switch is the basic Linux identification mechanism. It supports many system databases, but our concern is only with the /etc/passwd and /etc/group files.

Linux NSS consists of support in the GNU glibc Version 2, various name service implementations in /lib/libnss_*service*.so libraries, and the /etc/nsswitch.conf configuration file. In Example 1-1 on page 5, we show a simple configuration file. The first line specifies that the host name resolution should be first attempted through /etc/hosts and, in case of failure (for whatever reason), through DNS. The next three lines specify that information found through an LDAP directory service takes precedence over local files. Note that the service failover is done automatically by default. Other configurations are possible, as the last line in this example illustrates: The map file will be searched only in the case NIS servers are not available. For detailed information about the NSS configuration, refer to the nsswitch.conf(5) man page.

Example 1-1 The Linux /etc/nsswitch.conf file

```
hosts:      files dns
passwd:     ldap files
shadow:     ldap files
group:      ldap files
netgroup:   nis [notfound=return] files
```

1.2 Pluggable Authentication Modules (PAM)

PAM secures a common ground for applications on one side and the operating system's security subsystem on the other. It is a system software consisting typically of a system library and a bunch of modules. The library implements a PAM API and provides a connection between PAM modules and applications. The modules look exactly the same way to applications, even though they may handle widely different authentication methods. It is of no importance to an application requesting an authentication service whether a PAM module is using plain ASCII files or using fancy protocols to “talk” to a server located somewhere in Wichita.

PAM has been the standard Linux authentication mechanism for many years now. AIX 5L Version 5.2 is the first AIX release having complete PAM support.

1.2.1 PAM configuration

PAM configurations come in two flavors: Single file and multiple files. We find the former on AIX: All PAM configuration is stored in /etc/pam.conf (see Example 1-2). The latter is typical for most Linux distributions: The configuration is stored in multiple files in the /etc/pam.d directory (see Example 1-3 on page 6). However, both flavors have the same functionality. Some PAM implementations even allow either flavor. The only difference, as shown in the examples, is that the single file configuration contains settings for all applications and services lumped together. Therefore, the first colon in this file specifies the service.

Example 1-2 AIX PAM configuration (/etc/pam.conf)

```
sshd    auth        required        /usr/lib/security/pam_aix
sshd    account     required        /usr/lib/security/pam_aix
sshd    password    required        /usr/lib/security/pam_aix
sshd    session     required        /usr/lib/security/pam_aix
```

Example 1-3 Linux PAM configuration (/etc/pam.d/sshd)

```
auth       required     /lib/security/pam_nologin.so
auth       required     /lib/security/pam_unix.so
account    required     /lib/security/pam_unix.so
password   required     /lib/security/pam_unix.so
session    required     /lib/security/pam_unix.so
```

PAM is a *very* complicated subject, and reading its associated manuals is a necessity. This is a particularly sensitive security area, and mistakes in this area will cost you dearly.

1.2.2 PAM keywords

The first column in Example 1-3 contains four distinct words. They are called management groups or types in PAM parlance and refer to various authentication and session establishment phases:

auth Authentication. In this phase, the user presents his credentials, typically the name and the password.

account Account verification. The system verifies if the user's account data is valid, for example, the password has not expired or the account has not been retired.

session Session setup. The user has already been allowed access to the system. This phase is auxiliary and consists of setup and various other tasks. Typical examples are making syslog entries, imposing system limits, and printing the message of the day.

password Password change. This is used only if the user is forced to change the password. The most common example is password expiration.

Even though our examples feature all four types, that is not a requirement. For example, if the application does not need a session setup, then the session phase can be omitted. Also, PAM modules are not required to support all groups.

A system administrator may deploy more than one module for a phase. This is referred to as *module stacking* and the evaluation is ordered from top to bottom.

The second column contains the module control and the third is a reference to the module's image. The control states how important a role the module plays in the phase. Things get a bit complicated here, so we turn to an example to explain how to use PAM controls.

PAM example: The ftp service

The ftp service has a few peculiarities that have to be handled in a special way. It lends itself nicely to this discussion. We present one possible PAM configuration for the ftp service in Example 1-4. It employs no less than six PAM modules.

Example 1-4 PAM configuration for the ftp service

```
auth    sufficient      pam_ftp.so
auth    required        pam_unix.so
auth    required        pam_listfile.so file=/etc/ftpusers
auth    required        pam_shells.so
account requisite       pam_time.so
account required        pam_unix.so
session required        pam_unix.so
session required        pam_limits.so
```

The authentication phase starts with the pam_ftp module, which returns success if the user is ftp or anonymous, the anonymous ftp has been enabled, and the user presents a valid e-mail address. As this module is marked *sufficient*, if it returns success, the whole authentication phase succeeds and the rest of the auth modules are skipped. Otherwise, the next module invoked is pam_unix, which checks user credentials. In this case, as this module is *required*, the user must present a good password. The pam_unix module hashes the password and compares it with the hashed value in /etc/shadow. The pam_listfile makes sure that the user is not listed in /etc/ftpusers, which is used to prohibit the service for users such as adm or root. Finally, the pam_shells module verifies if the user's shell is valid.

In short, the authentication phase succeeds either if pam_ftp returns success or if all three modules return success. Note that it is necessary to have pam_ftp, the one marked *sufficient*, listed as the first. Note also that all three *required* modules are evaluated regardless of the values they return. If one of them fails, even though the final result of the authentication phase is known, the other modules are evaluated.

In Example 1-4, the account phase is controlled by two modules. The first is pam_time, which checks if the user is allowed to use the service at that moment. This module is a *requisite*, and if it fails, the processing will stop and the service will be denied. Otherwise, the pam_unix module will check for standard situations, such as a retired account. In the last phase, the session is logged by pam_unix and user limits are set by the pam_limits module. Note that there is no password phase defined; the ftp application have no provision for password changing.

Here are the exact rules for PAM processing:

1. If a *requisite* module fails, the processing stops and an authentication failure status is returned to the service.
2. If a *sufficient* module returns success, the processing stops and an authentication succeeded status is returned to the service.
3. If no *required* modules failed and at least one yielded success, an authentication succeeded status is returned to the service. All required modules are always executed.
4. If none of the required modules reached a decision, *optional* modules are used to decide the user's fate. In this case, all optional modules specified are evaluated.

Note that PAM modules are allowed to return neither success or failure. This is essential in some situations. Most modules always either deny or grant, for example, a user either belongs to a certain group or he does not, or a user either presents a good password or does not. On the other hand, a PAM module depending on the network may return the ignore status if a server is unreachable or a network interface is down, in which case it may be desirable to fail over to another authentication method. You should always check the actual module's documentation for a precise description of the module.

Just to make things more complicated, PAM modules may accept arguments. In Example 1-4 on page 7, the pam_listfile module has its file parameter set to /etc/ftpusers. A typical option supported by most PAM modules is *debug*: Just put that word to the right of the module name and you will receive many error messages through syslog.

More recent PAM implementations support finer grained control instead of the four control flags just discussed. Instead of specifying a control flag, the administrator can put a list of retval=action pairs in the second field. This is out of scope of this text and we refer you to:

http://www.softpanorama.org/Security/pam.shtml

1.3 Linux PAM implementation

The Linux PAM implementation is essentially what we describe in 1.2, "Pluggable Authentication Modules (PAM)" on page 5. However, various Linux distributions have different ways of managing the PAM configuration. The Linux distribution we consider in this redbook is Red Hat Linux Version 8.0.

Red Hat Linux uses the pam_stack PAM module to provide a system level PAM configuration. The pam_stack module simply redirects the action to a specified

file, thereby allowing centralized control. We show a typical sshd PAM configuration in Example 1-5. Every evaluation of the pam_stack module will result in evaluation of the /etc/pam.d/system-auth file, as specified. This way, we may keep the authentication setup in just one place.

Example 1-5 /etc/pam.d/sshd on a Red Hat host

```
auth       required     /lib/security/pam_stack.so service=system-auth
auth       required     /lib/security/pam_nologin.so
account    required     /lib/security/pam_stack.so service=system-auth
password   required     /lib/security/pam_stack.so service=system-auth
session    required     /lib/security/pam_stack.so service=system-auth
session    required     /lib/security/pam_limits.so
session    optional     /lib/security/pam_console.so
```

1.4 AIX PAM implementation

Apart from the PAM system library and development support, AIX 5L Version 5.2 includes the PAM loadable authentication module and pam_aix PAM module. These modules provide a bridge between the standard AIX security subsystem and PAM. The AIX PAM support consists of the following:

- /usr/lib/libpam.a, /usr/include/security/pam_*.h

  A shared system library included in bos.rte.security, which includes support for PAM modules. A couple of modules include files for PAM development (pam_appl.h and pam_modules.h).

- /usr/lib/security/pam_aix

  A PAM module providing AIX authentication services. PAM applications such as ssh can use it.

- /usr/lib/security/PAM

  An AIX authentication load module that provides access to PAM modules for authentication.

- AIX applications

  Some AIX applications and services may use different PAM services through the AIX PAM authentication module. This is the list of programs and services: ftp, login, passwd, rexec, rlogin, rsh, su, and telnet.

1.4.1 PAM modules and AIX

AIX 5L Version 5.2 includes one simple PAM module: /usr/lib/security/pam_aix. This module maps PAM API calls to AIX security calls. It provides AIX security services to PAM applications.

It is also possible to install new PAM modules. At the time of writing this redbook, we were not able to find any pre-built AIX PAM modules, so the only option is to download the module source, compile, and install. To do that, we need the following packages:

- AIX development toolkit (bos.adt.* from the Base Operating System)
- Development tools and utilities from the Linux toolbox (autoconf, automake, binutils, bison, db, fileutils, flex, gcc, libtool, m4, make, patch, sh-utils, and wget)

Not all of the listed packages are necessary for compiling open source software, but it does not hurt to have them installed.

Compiling and installing a PAM module

We choose to compile and install the pam_ldap module. This module supports the LDAP authentication (rfc2307 schema) and it is standard on Linux. For more details, see:

http://www.padl.com/

First, we download and unpack the source code tarball:

```
$ wget -q ftp://ftp.padl.com/pub/pam_ldap.tgz
$ gzip -dc pam_ldap.tgz | tar xf -
$ ls -l
total 240
drwxr-xr-x   3 joe    staff          4096 Oct 25 16:01 pam_ldap-156
-rw-r--r--   1 joe    staff        115648 Oct 25 15:53 pam_ldap.tgz
$ cd pam_ldap-156
```

Note that the version may change. List the directory to find out about the current version.

The most complex task is configuring the source code for compilation:

```
$ LDFLAGS="-L/opt/freeware/lib" LIBS=-lc \
   CPPFLAGS="-I/opt/freeware/include -D_LINUX_SOURCE_COMPAT -DPAM_EXTERN=" \
   ./configure --with-ldap-lib=openldap \
   --with-ldap-conf-file=/etc/pam_ldap.conf --enable-ssl
```

This very long command line configures the source code for compilation. However, you have to make a few more changes in the makefile file:

- Add `-L/opt/freeware/lib` to the definition of pam_ldap_so_LDFLAGS
- Change all occurrences of `-g root` to `-g system`

- Create the exports.aix file with the following contents (it is essentially a copy of exports.linux):

```
pam_sm_authenticate
pam_sm_acct_mgmt
pam_sm_setcred
pam_sm_open_session
pam_sm_close_session
pam_sm_chauthtok
```

Now we may proceed to compile and link the module:

```
$ gmake
...
```

Before installing the module, we recommend making a backup copy of any existing /etc/ldap.conf file. Even though we specified another name, the installation procedure may still copy the configuration file to /etc/ldap.conf:

```
$ su
# cp -p /etc/ldap.conf /etc/ldap.conf.bak
# gmake install
...
/bin/sh ./mkinstalldirs /lib/security
./install-sh -c -o root -g system pam_ldap.so /lib/security/pam_ldap.so.1
(cd /lib/security; rm -f pam_ldap.so; ln -s pam_ldap.so.1 pam_ldap.so)
:
if test ! -f /etc/ldap.conf; then \
        /bin/sh ./mkinstalldirs /etc; \
        ./install-sh -c -m 644 -o root -g system ./ldap.conf /etc/ldap.conf; \
fi
gmake[1]: Leaving directory `/usr/local/src/pam_ldap-156'
# cd /etc
# mv ldap.conf pam_ldap.conf
# mv ldap.conf.bak ldap.conf
```

Note that the pam_ldap module was not supported on AIX at the time of the writing of this redbook. The configuration process is probably going to be in better shape in new versions.

We also find an example PAM configuration file (pam.conf) in the source directory. Of course, it is not possible to use it without extensive changes, but it may be instructive to review its contents. The configuration of pam_ldap itself is essentially the same as on Linux systems. Therefore, we defer it until the next chapter, where we discuss the use of directory services for authentication.

As you can see, compiling PAM modules on AIX is somewhat involved, because the PAM support on AIX is brand new. However, we are hopeful that this will change in near future.

Configuring a PAM module

The PAM LDAP module is installed in /usr/lib/security/pam_ldap.so. We show the relevant fragment of /etc/pam.conf and the PAM LDAP module's own configuration /etc/pam_ldap.conf in Example 1-6.

Example 1-6 Using the PAM LDAP module on AIX

```
/etc/pam.conf:
   sshd    auth            sufficient      /usr/lib/security/pam_ldap.so
   sshd    auth            required        /usr/lib/security/pam_aix
   sshd    account         sufficient      /usr/lib/security/pam_ldap.so
   sshd    account         required        /usr/lib/security/pam_aix
   sshd    password        required        /usr/lib/security/pam_ldap.so
   sshd    session         required        /usr/lib/security/pam_ldap.so
   sshd    session         required        /usr/lib/security/pam_aix

/etc/pam_ldap.conf:
   host r-linux.weeorg.com
   base dc=weeorg,dc=com
```

In this example, the sshd daemon utilizes both the LDAP and the AIX security subsystem. However, the latter will be used only in case the LDAP fails. This way, we may support both local users as well as those defined in the directory.

See Chapter 2, "Centralized user management" on page 15 for more details on LDAP.

Note: In case you use PAM with AIX, applications always specify the use_first_pass flag to PAM modules, as shown in the following example:

```
telnet  auth      required  /usr/lib/security/pam_ldap.so use_first_pass
telnet  account   required  /usr/lib/security/pam_ldap.so
telnet  password  required  /usr/lib/security/pam_ldap.so
telnet  session   required  /usr/lib/security/pam_ldap.so
```

The use_first_pass flag instructs the PAM module to use the password that has already been provided to the system. In other words, AIX prompts the user for their name and password *before* invoking PAM. PAM modules should not interact with users nor print any messages.

1.4.2 PAM applications and AIX

The new ssh package available at the IBM OSS site is a PAM enabled application. You can download it and find installation instructions at:

http://oss.software.ibm.com/developerworks/projects/opensshi

This ssh version supports only the PAM API for authentication. Therefore, in order to use the sshd service, you must create /etc/pam.conf and configure PAM properly. We show a configuration that is suitable for authentication with the AIX standard security subsystem in Example 1-2 on page 5.

If AIX applications are to use PAM authentication services through the AIX security mechanism by setting the SYSTEM grammar in /etc/security/user, they can choose the PAM service by setting the PAM_SERVICE environment variable. These AIX applications and services set the PAM_SERVICE variables: ftp, login, passwd, rexec, rlogin, rsh, su, and telnet. If the service is not present in /etc/pam.conf, the PAM service that will be used is OTHER.

Compiling PAM applications on AIX

AIX 5L Version 5.2 includes complete support for development of PAM applications. As a proof of concept, we compile and run a small PAM helper program whose purpose is to enable PAM authentication for the squid Web caching server. The author of the program is Henrik Nordstrom, and the helper source code and some documentation are available at this site:

http://squid.sourceforge.net/hno/software.html

The source code consists of a single source file in pam_auth.c. We download the file and produce an executable image:

```
$ wget http://squid.sourceforge.net/hno/pam_auth-2.0.c
$ gcc -o squid-pamauth pam_auth-2.0.c -lpam
$ ls -l squid-pamauth
-rwxr-xr-x   1 joe staff        250945 Nov 06 14:27 squid-pamauth
```

The default PAM service used by this helper is squid and it supports auth and account PAM phases. Therefore, we place the following two lines in /etc/pam.conf:

```
squid   auth    required        /usr/lib/security/pam_ldap.so
squid   account required        /usr/lib/security/pam_ldap.so
```

The interface to the squid server is very simple: Squid will invoke the squid-pamauth program and pass a user name and a password on the standard input separated by space. Hence, we can easily determine whether the PAM program works correctly:

```
$ ./squid-pamauth
joe catchme
ERR
catch me joe
ERR
joe secret
OK
```

2

Centralized user management

On UNIX, there are two popular ways to provide single-source sign-on and centralized user management: Network Information Service (NIS) and Lightweight Directory Access Protocol (LDAP). The former is a well known Sun Microsystems product and it is available on most UNIX platforms. Still, in the last several years, LDAP has become very popular, and for good reasons.

Network Information Service (NIS) has a very bad reputation when it comes to security and it is suitable for trusted environments only. NIS+, an intended NIS replacement, is much better in this respect, but it is rather difficult to set up and administer.

Single-source sign-on consists of an authentication method and one copy containing the user and group information. The main advantage is that users use the same password to authenticate to all services. Another obvious advantage of the single-source sign-on is that users are administered in one place. This can save time and resources, depending on the number of hosts at the site.

In this chapter, we discuss various ways to set up centralized user management and provide single-source sign-on. Our main objective is to deploy LDAP for this purpose and show how AIX and Linux hosts use LDAP for authentication.

2.1 Lightweight Directory Access Protocol (LDAP)

LDAP was designed to make accessing the ISO X.500 directories easier. The ISO standard and its Directory Access Protocol (DAP) are extremely complex specifications that define the protocol through all seven OSI layers. Initially, LDAP was a client application for accessing the X.500 directory services. It relied on the operating system networking stack (TCP/IP), supported only the most important DAP operations, and was therefore much easier to implement. Later, the LDAP specification was also applied to the server side. LDAP became rather distinct from X.500.

LDAP is based on an open standard. No single vendor has control over the LDAP specification, but it has been developed as a joint effort of many IT companies and professionals and coordinated by the Internet Engineering Task Force (IETF). LDAP is supported and implemented on almost any computer platform today. These are the main reasons to deploy LDAP as a directory service.

2.1.1 Introduction to LDAP

The most common example of a directory is a telephone book, which consists of names (entries), their telephone numbers, and perhaps addresses and other useful information. The most important trait of the telephone book is that it is organized in a simple fashion facilitating quick lookup by name. Data stored in LDAP directories is not much different; there are objects (entries) and their attributes. Incidentally, if you run into a computerized telephone book on the Internet, there is a good chance that it has been implemented using LDAP.

Directory schemas describe the data stored in a directory. They define which attributes must be present in an entry, which attributes are optional, what kind of value an attribute may take, and whether is allowed to have multiple values. Every entry in an LDAP directory has a unique name called the Distinguished Name (DN) and that is the LDAP counterpart of a name in the telephone book. Here is one example of a DN:

```
dn: cn=Joe R. User,dc=weeorg,dc=com
```

This format is known as the LDAP Data Interchange Format (LDIF) and you will probably encounter it every now and then. The DN consists of a list of attribute=value pairs. In this example, cn stands for Common Name and dc for Domain Component. All these attributes are defined in a schema. The distinguished name is not of much use alone. Let us see what else is in Joe's entry:

```
cn: Joe R. User
cn: Joe User
cn: Joey
```

```
sn: User
givenName: Jack
title: Dr
mail: joe@weeorg.com
telephoneNumber: 11 (that's eleven)
objectClass: top
objectClass: person
```

Attribute names are on the left side followed by a colon. The rest of the line is an attribute's value. Note that the values may contain various kinds of strings in a fairly free format (though it usually helps if it is meaningful). Indeed, it is possible to assign *any* value to an attribute, including things like photos, which consist of binary data. The point is that LDAP could not care less about what is stored in a directory; it is up to the application to decipher and understand the data. LDAP just provides access and may restrict values to some syntax. Also, as you can see, an attribute may have more than one value.

The objectClass attribute is special; it refers to a schema where attributes are defined. Schemas are sets of related object classes and attribute definitions. Every object class defines the attributes that may be present.

The value of the DN also signals that the data is organized in a hierarchical way. The DN is coded similarly to the Domain Name System (DNS) host names with commas as separators rather than dots.

LDAP does not specify how to store data in the directory, only how to access it. Of course, every LDAP server features a database (usually called a back end), which is used to store and retrieve information. The database should be able to retrieve data quickly. Furthermore, it is assumed that the directory data does not change frequently. If it does, then you probably want to use a general database facility and not a directory.

2.1.2 Using LDAP for authentication

In the previous subsection, we say that attributes may be assigned any value. That is not entirely true. Some attributes may have a special meaning in the LDAP server itself, that is, the LDAP server is sometimes an LDAP application and, when in this role, it may be selective about the data served. In particular, all LDAP servers implement some authentication method.

An experimental standard is described in the Request For Comments (RFC) 2307: An Approach for Using LDAP as a Network Information Service (sometimes also referred to as RFC2307bis). This standard describes object classes and attributes for the purpose of identification and authentication. Even though it has not been classified as an IETF standard, it has been around for several years now and it is most often the common denominator for many

platforms and supported by many LDAP server and client implementations. Now, if we want to use an LDAP directory service for various operating systems, identifying the common denominator sounds reasonable.

The following object classes are included in RFC2307 (there are more, but these are featured in this text):

- posixAccount
- posixGroup
- shadowAccount

The posixAccount and posixGroup classes describe fields present in the /etc/passwd file. The shadowAccount consists of attributes that correspond to the fields in /etc/shadow. In Example 2-1, we show the relevant excerpt from RFC2307. The object class definitions may be intimidating at first sight, but we refer you to the MAY and MUST sentences, which contain a list of optional and mandatory attributes. The uid and userPassword attributes are used for user authentication. In Example 2-2 on page 19, we show a typical user entry in the LDIF format. Incidentally, even though we write object names in mixed case, it does not matter, not even in attribute values, which obviously has some effect on authentication (note the user name given):

```
$ telnet r-linux
Trying...
Connected to r-linux.
Escape character is '^]'.
Red Hat Linux Version 8.0 (Psyche)
Kernel 2.4.18-14 on an i686
login: Joe
Password:
Last login: Tue Nov  5 10:54:21 from 9.3.5.25
[joe@localhost joe]$
```

Of course, the case is significant for passwords, because they are stored as hashes.

Example 2-1 Object class definitions

```
( nisSchema.2.0 NAME 'posixAccount' SUP top AUXILIARY
  DESC 'Abstraction of an account with POSIX attributes'
  MUST ( cn $ uid $ uidNumber $ gidNumber $ homeDirectory )
  MAY ( userPassword $ loginShell $ gecos $ description ) )

( nisSchema.2.1 NAME 'shadowAccount' SUP top AUXILIARY
  DESC 'Additional attributes for shadow passwords'
  MUST uid
  MAY ( userPassword $ shadowLastChange $ shadowMin
        shadowMax $ shadowWarning $ shadowInactive $
```

```
        shadowExpire $ shadowFlag $ description ) )

( nisSchema.2.2 NAME 'posixGroup' SUP top STRUCTURAL
  DESC 'Abstraction of a group of accounts'
  MUST ( cn $ gidNumber )
  MAY ( userPassword $ memberUid $ description ) )
```

Example 2-2 User LDIF entry

```
dn: uid=joe,ou=People,dc=weeorg,dc=com
uid: joe
cn: Joe R. User
objectClass: posixAccount
objectClass: shadowAccount
objectClass: top
userPassword: {crypt}h/WA58b9dz5nI
loginShell: /bin/ksh
uidNumber: 1000
gidNumber: 100
homeDirectory: /home/users/joe
gecos: Joe R. User
shadowLastChange: 11627
shadowMax: 99999
shadowWarning: 7
```

2.2 Planning for LDAP authentication

We suppose that everybody is already used to vendors implementing services in their own way even though the service itself may be described by a standard endorsed by everybody in the industry. It is important to check out all the options available and see how they relate to each other. Deploying LDAP is not an exception. Therefore, we examine many solutions for LDAP authentication on both AIX and Linux.

We need an LDAP server to offer access to the identification and authentication data and to keep it in good shape. We also need an LDAP client that will be sufficiently integrated into the operating system to be able to take over the identification and authentication tasks. We need a thing or two more, such as implementing the security policy and encrypting the communication (we discuss these matters later in this chapter).

AIX 5L Version 5.2 is delivered with both an LDAP server (IBM Directory Server) and an LDAP client. Of course, coming from the same vendor, these two cooperate nicely. If you are running an AIX-only shop and you plan to stick to it, this is probably the optimal solution. With the recently included support for PAM,

the AIX LDAP authentication module is not the only option, but it is the only way to preserve the full set of user attributes from the standard AIX security subsystem.

To authenticate to LDAP, Linux uses a PAM LDAP module. The most popular is an open source product from PADL (see http://www.padl.com/). They developed the Name Service Switch (NSS) module for LDAP as well, thereby offering a complete solution for the LDAP client side. The NSS LDAP module runs on AIX too.

As for an LDAP server, there is the OpenLDAP directory server. It is also open source and well supported on Linux. As Linux's role in corporate environments has become more important, so has the number of different LDAP servers running on it. Almost all commercial LDAP servers support Linux today. IBM Directory Server also runs on Linux and, though not open source, it can be deployed free of charge.

The PAM LDAP client on Linux supports the RFC2307 standard. AIX 5L Version 5.2 LDAP authentication features mapping between AIX attributes and RFC2307, so Linux and AIX LDAP client can share the same data.

2.3 LDAP servers

Many LDAP servers are available on the market. Most of them should be pretty stable, as they have already undergone so many releases (and owner changes). Here is a list of things to watch out for:

- LDAP standards

  One of the reasons for choosing LDAP for directory services is that it is based on IETF standards. Hence, you definitely want your LDAP software to follow the RFCs, and to have LDAP Version 3 support.

- Interoperability

  It is preferable to run LDAP servers that are capable of talking the same language to other LDAP servers. We do not mean LDAP here, but replication. All serious organizations use replication with LDAP for higher availability. Also, LDIF support is important for the directory data backup and transfer purposes.

- Secure Sockets Layer (SSL) and Transport Level Security (TLS) support

  Encrypting LDAP communication is a must for most organizations. LDAP is more and more often used for security sensitive applications rather than as a fancy phone book. Though all servers support SSL, beware: It is one of those notorious standards where nobody is actually sure how it was supposed to be

implemented. Almost all products supporting SSL has one or more oddities in the implementation.

- Access Control

  The server must support some kind of access control. The question is how easy it is to administer access and how flexible is the access control system. A simple error in access control and suddenly your (more enterprising) users may be able to read data like other people's salaries or other users's passwords.

- Security policy

  You want to make sure that your organization security policy may be implemented in the LDAP area too. Note that it is more convenient, logical, and sometimes necessary to implement the policy at the server and not on clients.

- Authentication methods

  Various authentication methods can be a life saver in environments with changing demands. That is even more true if LDAP is to be used for authentication. You should look for Simple Authentication and Security Layer (SASL), Generic Security Service (GSS) API, and SSL (TLS) support.

- Database

  You should verify if the back-end database can support the estimated amount of data and load. Support for transactions and database robustness are important for the data safety.

- Development kits

  If you expect to develop LDAP applications, it may be a good idea to have the LDAP Software Development Kit (SDK) from the same vendor. LDAP API has still not been standardized, and, even though various vendors' API support is similar, APIs may be different enough to become a serious difficulty.

Of course, the importance of items in this list will vary depending on your actual needs. Furthermore, the list may not be complete, and, due to a high change rate in the IT industry, it will some day definitely be out of date. LDAP is still a relatively recent standard, so expect to see many new features. Also, new features may be something to beware of; try not to depend on them, simply because they may come and go, or become a peculiar single-vendor supported extension.

In the next section, we take a look at a couple of LDAP servers: IBM Directory Server and open source OpenLDAP directory server.

2.3.1 IBM Directory Server

IBM Directory Server was formerly known as IBM SecureWay® Directory Server. At the time of the writing of this redbook, the current version is 4.1 and it is available on AIX, Linux, Solaris, HP-UX, Windows NT, and Windows 2000. Linux Turbolinux, Red Hat, and SuSE distributions are supported as well as Linux on zSeries™ platforms. This product is available free of charge.

Installation

The installp images of the IBM Directory Server are delivered with the AIX 5L Version 5.2 distribution. Alternately, they can be ordered on a CD-ROM media or downloaded from the IBM site at:

http://www.ibm.com/software/network/directory/server/index.html

The distribution consists of the following components:

- LDAP server
- DB2® server
- LDAP client
- HTTP server (recommended for administration)
- GSKit (recommended for SSL connections)

Installation and initial configuration is fairly simple. Install all ldap.* filesets, one HTTP server (such as IBM HTTP server http_server.*), and the AIX GSKit (gskkm.rte), in case you would like to use encrypted communication which we strongly recommend. Several kinds of HTTP servers are supported: IBM HTTP server, Domino™ Webserver, Apache Server, and iPlanet Webserver. The DB2 filesets will be installed automatically, as they are required for the LDAP server. In Example 2-3, we show a sample installation using the command line interface and the installp program.

Example 2-3 IBM Directory Server sample installation

```
# installp -acgXYd src ldap.server ldap.client ldap.html.en_US ldap.msg.en_US
# installp -acgXYd src http_server.base http_server.ssl \
        http_server.admin http_server.modules \
        http_server.msg.en_US http_server.html.en_US
# installp -acgXYd src gskkm.rte ldap.max_crypto_client ldap.max_crypto_server
```

Alternatively, you can use the SMIT interface or the Web-based System Manager to perform the software installation. We show how to use the Web-based System Manager Graphical User Interface (GUI). **Choose Software** -> **Installed Software** in the navigation menu on the left and then **Software** -> **New Software** -> **Install Additional Software** -> **Advanced Method** (see Figure 2-1 on page 23). In the dialog box that pops up, fill in the software source (a directory or other media), type “ldap” in the field below, and click on Add Entry

and then OK (see Figure 2-2 on page 24). The software installation will start. Repeat the same procedure with the appropriate source for the http_server, gskkm, ldap.max_crypto_client, and ldap.max_crypto_server software subsets. For more details on Web-based System Manager, see 10.1, “Web-based System Manager” on page 226.

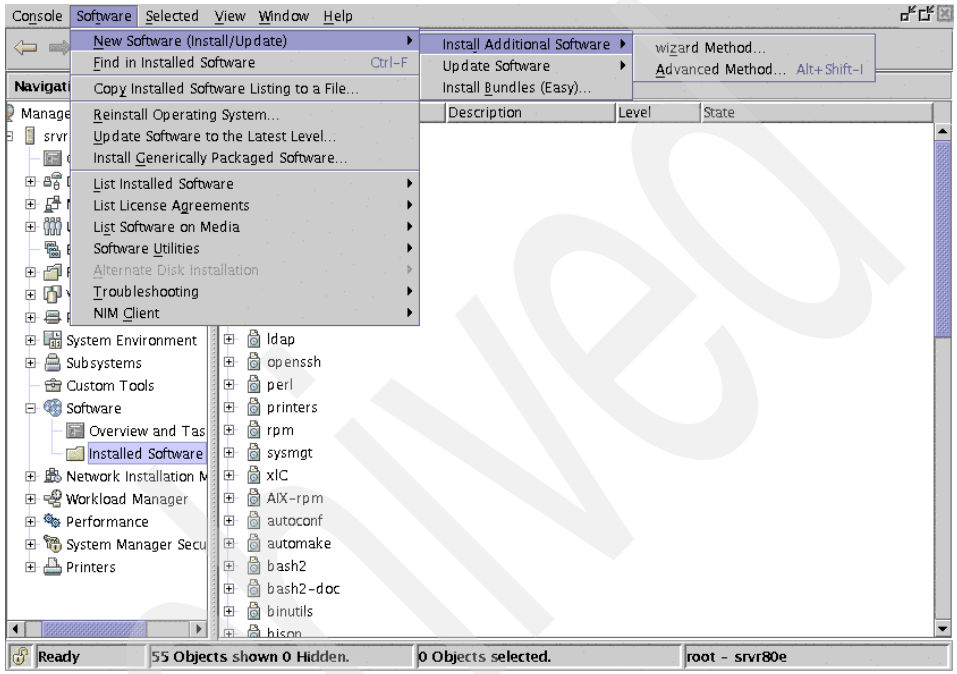

Figure 2-1 Web-based System Manager software install interface

Initial configuration may be done using ldapxcfg (graphical interface) or ldapcfg (command line interface). In this step, you choose the administrator’s DN (default is cn=root), the password, a few database create options, and an HTTP server type. In Example 2-4 on page 24, we show one example initial configuration. Having an HTTP server is optional, but we strongly recommend you install one. Note that it is necessary to have the HTTP server in place before starting the initial configuration utility (ldapxcfg or ldapcfg) and that you must restart the server after the configuration. The database settings should be fine for most requirements. If you expect a big database, refer to the *IBM Directory Server Version 4.1 Installation and Configuration Guide for Multiplatforms*, available at:

http://www-3.ibm.com/software/network/directory/library/publications/41/config/ldapinst.htm

Example 2-4 IBM Directory Server initial configuration

```
# ldapcfg -u cn=root -p secret -c \
        -s ibmhttp -f /usr/HTTPServer/conf/httpd.conf \
        -l /home/ldapdb2
```

After the initial configuration is complete, it is mandatory to restart the HTTP server:

```
# /usr/HTTPServer/bin/apachectl stop
# /usr/HTTPServer/bin/apachectl start
```

Note: The default database disk requirements for version 4.1 are at least 80 MB on the /home file system. Make sure that there is enough space or the ldapcfg program will fail.

Figure 2-2 Installing LDAP software with Web-based System Manager

Configuration

Configuration of the IBM Directory Server is stored in /usr/ldap/etc/slapd32.conf with a convenient link in /etc. However, you should use a standard browser with support for frames for configuration and administration. Just point it to `http://server/ldap/index.html` and log in with the DN you set with ldapcfg (default is cn=root). The navigational area is in the frame on the left side. A sample window of the Web browser is shown in Figure 2-3.

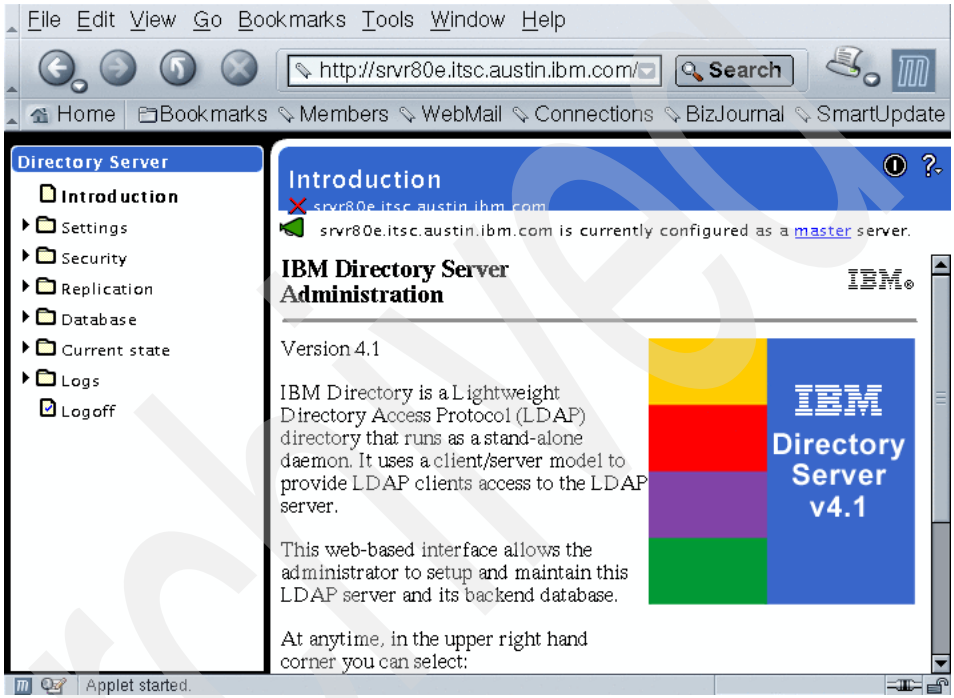

Figure 2-3 IBM Directory Server Web administration interface

Password encoding

In Example 2-2 on page 19, the userPassword attribute begins with the "`{crypt}`" string. This string denotes the password encoding format (password hash function). The server supports three hashes: *crypt*, *sha-1*, and *imask*. The default setting is imask, which is a two-way encoding function, unlike crypt and sha-1. If your applications require clear text passwords, this is the option you should choose. Otherwise, crypt or sha-1 are better choices. The latter uses a hash with more cryptographic strength, but the former has an advantage of being used on most UNIX systems. Therefore, if you are migrating users and want to keep their passwords, the crypt hash is probably the best choice. To change the password encoding, select **Settings** -> **General** and pick one of the Password encryption choices.

Note: After every change, you make you must click on the **Update** button to actually update the configuration. Furthermore, the server must be restarted for the changes to take effect.

Choosing the suffix

All DNs administered at one site share one or more suffixes. That is typically a DN referring to the site. The one we are going to use throughout this redbook is “dc=weeorg,dc=com”. Select **Settings** -> **Suffixes** to input the suffix in the **Suffix DN** box. Directory servers commonly feature multiple suffixes and databases, but one suffices in this text.

Starting and stopping the server

At this point, you can start the server by clicking on the power button, which resides in the upper right corner of the web page (see Figure 2-3 on page 25), or by using the command line:

```
# /usr/bin/slapd
```

To restart the server, click on the power button again. To stop the server, either go to **Current state** -> **Start/Stop** and click on the Stop button, or use the command line:

```
# kill ‘cat /etc/slapd.pid‘
```

2.3.2 The OpenLDAP directory server

The OpenLDAP software has its beginnings in the first LDAP implementation from the University of Michigan. The UoM still has some Web presence regarding the LDAP directory services, but the primary site for software and documentation is at:

http://www.openldap.org/

The OpenLDAP software supports LDAP Version 3 and there are currently two code trees maintained in parallel:

- Release 2.1.x. This tree is used for most of the new features. Still, it should be safe for general use.
- Stable Release 2.0.x. This release is the most stable.

Current code from OpenLDAP is reasonably stable and of production quality. Many organizations entrust their directory services to the OpenLDAP directory server.

OpenLDAP, like most open source software, runs on a myriad of platforms. The preferred installation method for AIX is to install the RPM packages from the AIX toolbox for Linux site:

ftp://ftp.software.ibm.com/aix/freeSoftware/aixtoolbox/RPMS/ppc/openldap/

At the time of the writing of this redbook, the available version is 2.0.21. Note that this package does not include support for encrypted communication (TLS/SSL). In order to include support for encryption, the source code has to be compiled.

In this subsection, we will cover installation on a Linux platform. Even though we cover Red Hat Linux Version 8.0, most of the following procedure applies to any Linux distribution.

Installation

We need to install the following packages: openldap, openldap-clients, openldap-servers, and nss_ldap. In addition, the following packages are required: OpenSSL (openssl), Kerberos support (krb5-libs), and Cyrus SASL (cyrus-sasl and cyrus-sasl-md5). It is somewhat difficult to track the RPM dependencies, so this list may not be complete and may change in the future.

The RPM package for AIX has no support for SASL nor SSL, so we install only one image:

```
$ wget -q ftp://ftp.software.ibm.com/aix/freeSoftware/aixtoolbox/RPMS/\
        ppc/openldap/openldap-2.0.21-3.aix4.3.ppc.rpm
# rpm -i openldap-2.0.21-3.aix4.3.ppc.rpm
```

Note that the backslash ("\") at the end of the line means that the indented text that follows should be typed in the same line.

We perform essentially the same procedure on Linux, but installed many more packages:

```
$ wget ftp://ftp.redhat.com//pub/redhat/linux/current/en/os/i386/RedHat/RPMS/\
        openldap*2.0.25*rpm
$ wget ftp://ftp.redhat.com//pub/redhat/linux/current/en/os/i386/RedHat/RPMS/\
        cyrus-sasl*rpm
$ wget ftp://ftp.redhat.com//pub/redhat/linux/current/en/os/i386/RedHat/RPMS/\
        krb5-*rpm
$ wget ftp://ftp.redhat.com//pub/redhat/linux/current/en/os/i386/RedHat/RPMS/\
        pam_krb5-*rpm
$ ls *.rpm
cyrus-sasl-2.1.7-2.i386.rpm         krb5-server-1.2.5-6.i386.rpm
cyrus-sasl-devel-2.1.7-2.i386.rpm   krb5-workstation-1.2.5-6.i386.rpm
cyrus-sasl-gssapi-2.1.7-2.i386.rpm  openldap-2.0.25-1.i386.rpm
cyrus-sasl-md5-2.1.7-2.i386.rpm     openldap-clients-2.0.25-1.i386.rpm
cyrus-sasl-plain-2.1.7-2.i386.rpm   openldap-devel-2.0.25-1.i386.rpm
```

```
krb5-devel-1.2.5-6.i386.rpm         openldap-servers-2.0.25-1.i386.rpm
krb5-libs-1.2.5-6.i386.rpm          pam_krb5-1.56-1.i386.rpm
# rpm -i openldap*2.0.25*rpm cyrus-sasl*rpm krb5-*rpm pam_krb5-*rpm
```

Configuration

OpenLDAP still does not have any kind of graphical user interface for configuration and administration. The main configuration utility is vi or your favorite text editor. The configuration is kept in slapd.conf. It is a flat ASCII file and usually resides in /etc/openldap. We show a sample configuration in Example 2-10 on page 42.

The OpenLDAP configuration consists of two parts: The global options and the database (or back-end) options. Global options apply to the slapd directory server. They are followed by zero or more back-end definitions. We discuss the most used options. See the slapd.conf(5) man page for more details.

Global options

The OpenLDAP distribution includes several LDAP schemas. Those used in back ends must be specified in the global options part. Actually, object class and attributes definitions are global options, but typically kept in separate files, which are included in slapd.conf. It is common to start the configuration file with included schemas. In case an attribute or object class is referenced in data, but not defined in any schemas, slapd refuse to start.

Transport Level Security (TLS) options

Availability of a secure communication channel is a must no matter how trusted your networks may be. The TLS options in the second paragraph of Example 2-10 on page 42 specify the location of the Certificate Authority (CA) and server certificates, the server key, and the minimum cryptographic strength. In this example, we use certificates created and locally managed with OpenSSL utilities. Even though it is possible to use commercially available certificates, it is cheaper to create certificates using OpenSSL for the purpose of serving local LDAP clients. TLS is a new protocol similar to SSL and is described in RFC2246. Note that when the security option for the minimum cryptographic strength is specified, slapd does not accept non-encrypted connections.

Database options

The database part starts with the database keyword. The slapd server is modular, for example, additional back ends may be implemented and dynamically loaded without the core code change. Naturally, the database configuration options depend on the back-end's nature. However, many options are common to all back-end types.

The database most commonly used is ldbm, which is the Berkeley DB implementation. The suffix keyword defines the common DN suffix for the

database. The rootdn and rootpw keywords specify the administrators DN and the password. Note that the rootdn must contain the database suffix. The password can be in clear, but it is preferable to use the slappasswd program to create a password hash:

```
# slappasswd -s joejoe
{SSHA}7bJFkS/BfENSoxkOk2+MT4KyXgtIRTgO
```

The directory keyword specifies the location where database and index files will be stored. The index keyword specifies which attributes are to be used as indexing keys. Indexing the database is very important and sometimes overlooked, resulting in bad directory server performance. All attributes on which a search is going to be regularly performed should be indexed. In our example, the most important ones are user and group names and IDs. Of course, your applications govern the choice of indexes to maintain.

Finally, access control is very important in most applications. The default access in OpenLDAP slapd is read. That is definitely not appropriate for user passwords, so we allow only the authentication operation on the userPassword attribute. The self keyword denotes the entry itself. In our example, that means that the user who has already authenticated himself is authorized to change the password.

Starting and stopping the server

The openldap-servers RPM package includes a standard system V startup script. The script can be run from /etc/init.d directory:

```
# /etc/init.d/ldap start
# /etc/init.d/ldap stop
```

2.4 Migrating user information to LDAP

It is very seldom the case that an organization is deploying LDAP and creating the users' database from scratch. Usually, the existing users should be migrated to the LDAP directory.

Only users and groups that refer to real people should be migrated to LDAP. It does not make much sense to migrate users such as sys or daemon or groups such as bin or system for the following reasons:

- System users are local by their nature. Some applications and services that use them may be started before the network is available.
- Powerful users such as root should not be distributed; the security implications are too serious.
- System users and groups may have the same names on different platforms, but most often they do not have the same UIDs. No standard has been

adopted yet. These UIDs and GIDs own various system files and it would be virtually impossible to change them without jeopardizing the system's functionality.

Both AIX and Linux feature tools to convert various system maps to LDIF, which can be imported to an LDAP server. The tools are significantly different though. AIX utilities are part of the operating system itself and they will not only migrate the users but also configure the LDAP server and prepare the local security subsystem for LDAP authentication. On the other hand, the Linux utilities only produce the LDIF files.

2.4.1 Migrating users on Linux

A set of utilities for converting standard UNIX security maps such as /etc/passwd and /etc/group is available from the PADL site:

`ftp://ftp.padl.com/pub/`

and look for the MigrationTools.tar.gz file.

It is a suite of Perl scripts that produces LDIF compliant files ready for import to an LDAP server. We describe the conversion process in more detail in 2.6.1, "OpenLDAP server setup" on page 38.

2.4.2 Migrating users on AIX

The tools we use on Linux to migrate users may be used on AIX too. Some preprocessing is in place to bring data from the /etc/security/user and /etc/security/passwd files into the shadow form and that should not be difficult to do. However, AIX has its own set of utilities for LDAP configuration. These utilities are specific to the AIX LDAP authentication client and the IBM Directory Server.

AIX 5L Version 5.2 supports attribute mapping between the standard AIX security and the RFC2307 schema. Previous AIX versions have support only for the AIX specific schema. Linux supports only the RFC2307 schema, so we will use the same for AIX too. Populating LDAP with AIX specific data is fine if you are running an AIX only shop.

The all-purpose program for LDAP authentication setup is mksecldap. It not only converts local databases to LDIF, but also configures the IBM Directory Server, creates the DB2 instance, and configures the LDAP client for authentication. In case you use mksecldap to configure the server, there is no need to run ldapcfg or ldapxcfg to configure the server password and the DB2 instance. However, the mksecldap program does not configure the HTTP server for LDAP administration, so you may still want to run the following command:

```
# ldapcfg -s ibmhttp -f /usr/HTTPServer/conf/httpd.conf
```

Configuring the IBM Directory Server

The mksecldap program has two modes of execution: One for the server and another for the client. In the server mode, it creates the DB2 instance (if necessary), processes the server configuration (/etc/slapd32.conf), and optionally migrates users and groups. It converts the maps to LDIF and invokes ldif2db to put them into the database. Let us see what does the command line look like:

```
# mksecldap -s -a cn=root -p secret -S rfc2307 -d dc=weeorg,dc=com \
   -k /etc/security/ldap/server.kdb
```

Creating the SSL keys

The -k option specifies the absolute path to the database of SSL keys. The SSL keys must be created prior to using the gsk5ikm program from the GS kit (gskkm.rte). Use the -w option to specify the key database password. However, it is preferable to have the password stashed, just make sure that you use the same location and base file name with the extension ".sth". In this example, it is /etc/security/ldap/server.sth.

To manage SSL key databases, start the gsk5ikm program:

```
# gsk5ikm&
```

To create the new key database, select **Key Database File** -> **New** and fill in the fields, as shown in Figure 2-4 on page 32. When you click on **OK**, an additional pop-up dialog appears asking for the password. Type in the password and select the "Stash the password to a file" option. Using password stashes is a preferred method, because otherwise the password has to be typed whenever a program using the key database starts.

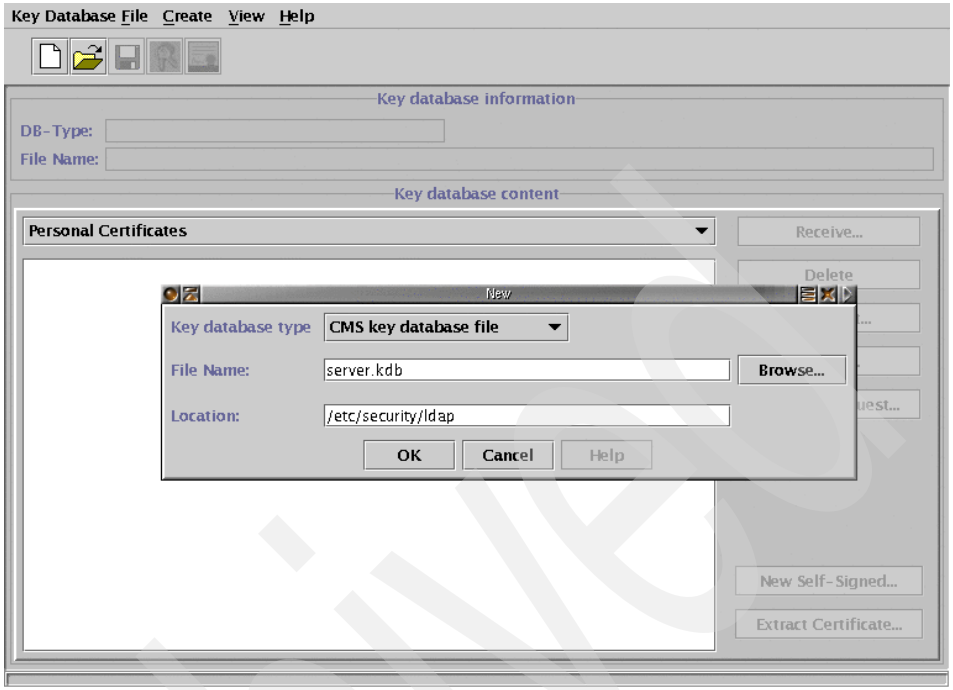

Figure 2-4 Creating the key database using gsk5ikm

The next step is to create a certificate request for the server. Select **Personal Certificate Requests** in the pull-down menu in the middle of the window and click on the **New** button. The Key Label field should describe the owner of the certificate and the Common Name is the server's Fully Qualified Domain Name (FQDN). When you click **OK**, the certificate requests will be stored in a file that should be sent to the Certificate Authority to be signed. This process is illustrated in Figure 2-5 on page 33.

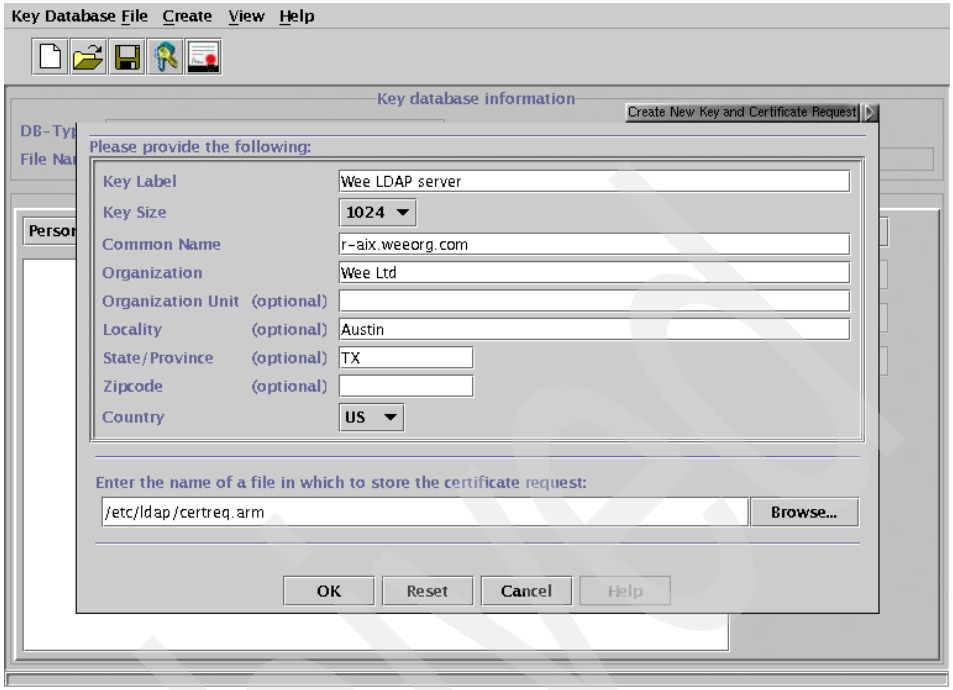

Figure 2-5 Creating a certificate request with gsk5ikm

To import the signed certificate, select the **Personal Certificates** portion of the key database and click on the **Receive...** button on the right. Select the appropriate data type (usually the Base64-encoded ASCII) and the file containing the signed certificate. Later in this chapter, we show how to create the OpenSSL certificate authority. The same utility may be used to sign the GS kit produced certificate requests.

Converting system maps to LDIF

The mksecldap program may print a message informing the administrator to restart ldif2db. In this case, one can use the sectoldif program to convert user and group maps to the LDIF format. Here is a sample session:

```
# sectoldif -d dc=weeorg,dc=com -S rfc2307 > all.ldif
# vi all.ldif # remove system users and groups
# db2ldif -i all.ldif
```

We recommend, as in the previous subsection, to edit the LDIF file and remove all systems and unnecessary users and groups.

2.5 LDAP authentication clients

In this section, we configure AIX and Linux hosts to use LDAP for authentication.

2.5.1 AIX LDAP authentication client

The AIX LDAP client support consists of the LDAP loadable authentication module and the client side secldapclntd daemon. The secldapclntd daemon accepts requests from the LDAP authentication module and retrieves and caches information from the LDAP server.

The mksecldap program that we encountered in the previous section is used to set the standard AIX LDAP client. In its client mode, it processes the /usr/lib/security/methods.cfg and /etc/security/user files to allow for LDAP authentication, and the /etc/security/ldap/ldap.cfg file used by the client side caching daemon secldapclntd. The tasks are quite different from those configuring the server, but invocation is remarkably similar:

```
# mksecldap -c -h r-linux.weeorg.com -a cn=root -p secret -S rfc2307 \
   -d dc=weeorg,dc=com -k /etc/security/ldap/client.kdb
```

The mksecldap program starts the secldapclntd daemon and adds an appropriate line to /etc/inittab.

Several programs are available for the secldapclntd control: start-secldapclntd, stop-secldapclntd, restart-secldapclntd, flush-secldapclntd, and ls-secldapclntd. The **flush** command flushes the daemon's cache, which may at times be convenient. We show a sample output from ls-secldapclntd in Example 2-5. This output resembles the /etc/security/ldap/ldap.cfg file, with the exception of attribute maps, which we show here:

```
userattrmappath:/etc/security/ldap/2307user.map
groupattrmappath:/etc/security/ldap/2307group.map
```

The map configuration is a consequence of the -S rfc2307 option. You can find available maps in the /etc/security/ldap directory.

Example 2-5 A sample output from ls-secldapclntd

```
ldapservers=r-linux.weeorg.com
ldapport=389
ldapversion=3
userbasedn=ou=People,dc=weeorg,dc=com
groupbasedn=ou=Group,dc=weeorg,dc=com
idbasedn=
usercachesize=1000
usercacheused=1
groupcachesize=100
```

```
groupcacheused=0
cachetimeout=300
heartbeatT=300
numberofthread=10
alwaysmaster=no
userobjectclass=posixaccount
groupobjectclass=posixGroup,top
```

The configuration may be done on a per-user basis or system wide, as we discuss in 1.1.1, “AIX security” on page 2. In Example 2-6, we show the relevant stanzas.

Example 2-6 LDAP authentication with the AIX LDAP authentication module

```
/usr/lib/security/methods.cfg:
   LDAP:
            program = /usr/lib/security/LDAP

/etc/security/user:
   joe:
            admin = false
            registry = LDAP
            SYSTEM = LDAP
```

2.5.2 Linux LDAP authentication client

Linux uses PAM for authentication and the name service switch mechanism for identification. The standard modules for LDAP are PAM LDAP and NSS LDAP. Since both of them are from the same source (PADL at http://www.padl.com), the configuration and setup are similar, and they can even share the LDAP configuration file. We illustrate the Linux security subsystem in Figure 2-6 on page 36.

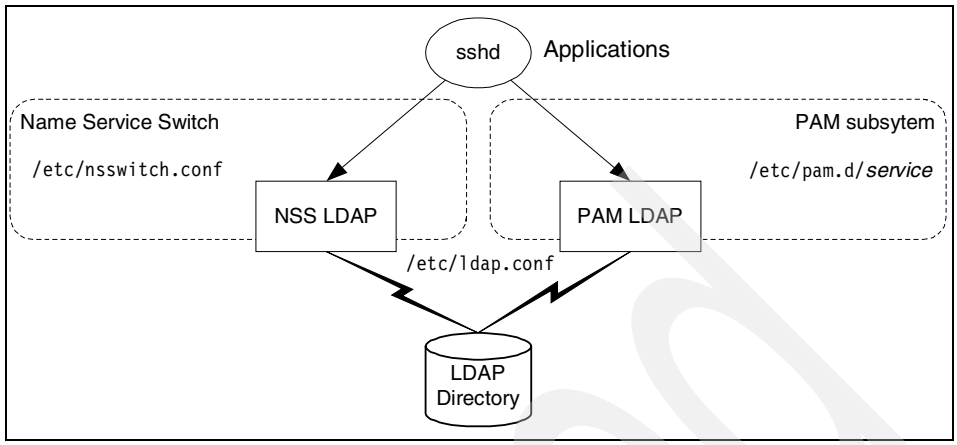

Figure 2-6 Linux security subsystem

2.5.3 PAM and NSS LDAP modules on AIX

The PAM LDAP module and AIX LDAP loadable authentication module authenticate users in very different ways. On the one hand, the PAM LDAP module authenticates users by trying to perform the LDAP bind operation as the given user. On the other hand, the AIX LDAP module (or, more precisely, the secldapclntd client side daemon) binds to the LDAP server as a privileged user, reads the user password, computes the hash of the user supplied password, and finally compares the two. The most significant consequence is that the AIX LDAP method requires privileged access to the LDAP directory. That may not always be in accord with the site's security policy. That is why we discuss an alternative LDAP authentication method here.

We show, in 1.4.1, "PAM modules and AIX" on page 9, how to compile and install the PAM LDAP module.

The PAM LDAP module on AIX is accessible either directly through PAM enabled applications or through the AIX PAM loadable authentication module. For the former, such as the ssh daemon, it is sufficient to edit the /etc/pam.conf configuration file. The latter, however, requires appropriate settings in the /etc/security/user and /usr/lib/security/methods.cfg files. We illustrate the differences in authentication between AIX and PAM applications in Figure 2-7 on page 37.

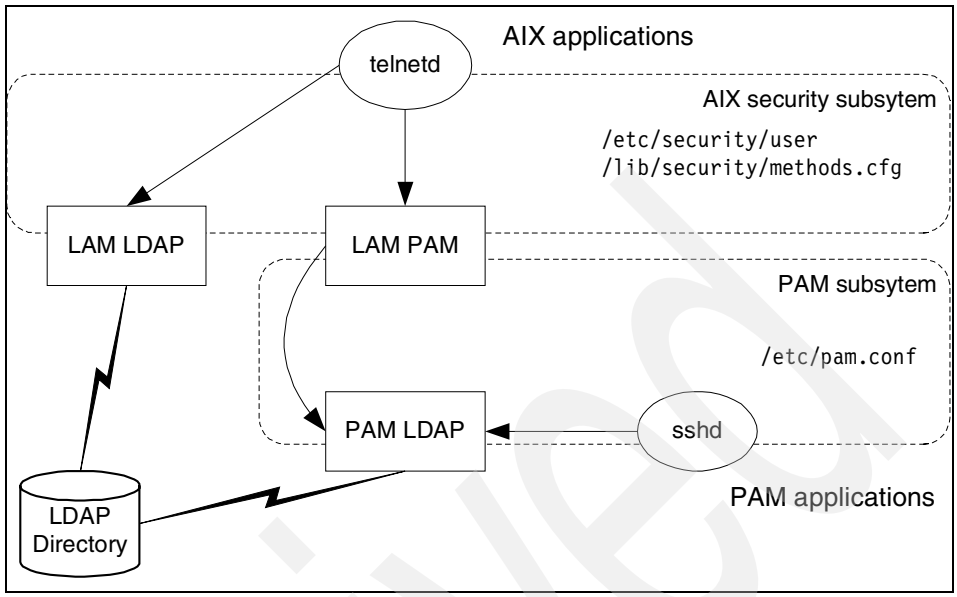

Figure 2-7 Different authentication methods on AIX 5L Version 5.2

In order to use PAM LDAP for ssh or any other PAM application, we have to reconfigure /etc/pam.conf. We show one possible configuration in Example 2-7. This configuration allows LDAP authentication and, if that fails, forces authentication through the AIX security subsystem. See the discussion about PAM in 1.2, “Pluggable Authentication Modules (PAM)” on page 5.

Example 2-7 PAM LDAP configuration on AIX

```
sshd    auth        required /usr/lib/security/pam_ldap.so use_first_pass
sshd    account     required /usr/lib/security/pam_ldap.so
sshd    password    required /usr/lib/security/pam_ldap.so
sshd    session     required /usr/lib/security/pam_ldap.so
```

Configuration of the PAM LDAP module itself is the same as on Linux (see Example 2-13 on page 46).

PAM is an authentication mechanism only, so the host still depends on local files for the name resolution. The NSS LDAP may be compiled as an AIX loadable module and installed in /usr/lib/security. Stanzas from Example 2-8 on page 38 should be included in /usr/lib/security/methods.cfg and referenced appropriately in /etc/security/user. For example, we may change the identification and authentication method for a user using the chuser program:

```
# chuser =R files SYSTEM=PAMLDAP registry=PAMLDAP joe
```

Example 2-8 Alternative LDAP identification and authentication

```
PAM:
        program = /usr/lib/security/PAM
        options = authonly

NSSLDAP:
        program = /usr/lib/security/NSS_LDAP
        options = dbonly

PAMLDAP:
        options = db=NSSLDAP,auth=PAM
```

The NSS_LDAP module has its own configuration file. Like the PAM LDAP pam_ldap.conf configuration file, it describes how to access the directory data. The file location may be specified during the source configuration process and it is convenient to use /etc/nss_ldap.conf. The syntax and semantics of nss_ldap.conf are the same as for the PAM LDAP configuration file. If /etc/pam_ldap.conf has already been configured, one can just create a soft link to nss_ldap.conf:

```
# cd /etc; ln -s pam_ldap.conf nss_ldap.conf
```

2.6 Deploying LDAP for authentication

In this section, we make a real example of LDAP system authentication for AIX and Linux. The choice we make is to use the OpenLDAP server, AIX LDAP authentication module for AIX applications, and PAM LDAP module for Linux and PAM applications on AIX, such as Secure shell. We need NSS LDAP module on Linux for the name resolution. We also use SSL and TLS to encrypt communication over the network, so there must be a working DNS. At the very least, the IP addresses of hosts running an LDAP server must be resolvable.

2.6.1 OpenLDAP server setup

The OpenLDAP server runs on a Red Hat Linux Version 8.0 host called r-linux.weeorg.com. We perform the following tasks:

1. Create CA and SSL server certificates.
2. Create the slapd configuration.
3. Migrate users.

Creating SSL certificates

We show a sample session where we create the CA and server certificates using the OpenSSL utilities in Example 2-9. In this example, the assumed location of the OpenSSL configuration directory is /usr/share/ssl, because it was run on the Red Hat Linux Version 8.0 host, but it depends on the actual software distribution.

Example 2-9 Creating CA and server certificates

```
# misc/CA -newca
CA certificate filename (or enter to create)
Making CA certificate ...
Using configuration from /usr/share/ssl/openssl.cnf
Generating a 1024 bit RSA private key
.....................................................................
.................++++++
.................................................++++++
writing new private key to './demoCA/private/cakey.pem'
Enter PEM pass phrase:
Verifying password - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [US]:
State or Province Name (full name) [Texas]:
Locality Name (eg, city) [Austin]:
Organization Name (eg, company) [Weeorg Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []:r-linux.weeorg.com
Email Address []:ca@weeorg.com
# misc/CA -newreq
Using configuration from /usr/share/ssl/openssl.cnf
Generating a 1024 bit RSA private key
..........................++++++
.................++++++
writing new private key to 'newreq.pem'
Enter PEM pass phrase:
Verifying password - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
```

```
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [US]:
State or Province Name (full name) [Texas]:
Locality Name (eg, city) [Austin]:
Organization Name (eg, company) [Weeorg Ltd]:
Organizational Unit Name (eg, section) []:LDAP server
Common Name (eg, your name or your server's hostname) []:r-linux.weeorg.com
Email Address []:joe@weeorg.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Request (and private key) is in newreq.pem
```

The OpenSSL distribution includes a simple CA utility. In our example, we first create the CA key and certificate with the **-newca** command. The CA utility will create several directories. At the top level, you will find a file called cacert.pem, which is the CA certificate.

To create a server certificate, we first create a certificate request using the **-newreq** command. Make sure to enter the FQDN of the host that will own the certificate as its Common Name, or SSL clients will have problems with the certificate. Finally, the **-signreq** command signs our request. When the CA resides on the same host as one of the LDAP servers, make sure that the server's certificate subject does not match exactly the CA certificate's subject. Otherwise, some SSL implementations may refuse to cooperate. In this example, we give "LDAP server" as the Organizational Unit and an e-mail address different from the one in the CA certificate. Either of them is enough to make the difference.

Using **openssl rsa**, we remove a password that protects the key; otherwise, the password has to be supplied whenever the LDAP server starts, which may be somewhat inconvenient (OpenSSL does not support password stashes):

```
# openssl rsa -in newreq.pem -out ldapkey.pem
read RSA key
Enter PEM pass phrase:
writing RSA key
# rm newreq.pem
```

The server certificate can be tested using the openssl s_server and s_client programs. To do this we need two terminals. In one, we start the server:

```
# openssl s_server -CAfile demoCA/cacert.pem -cert ldapcert.pem \
   -key ldapkey.pem -accept 2222
...
```

In the other, we connect to it using the client:

```
# openssl s_client -connect r-linux.weeorg.com:2222 -CAfile demoCA/cacert.pem
...
   Timeout   : 300 (sec)
   Verify return code: 0 (ok)
---
```

Note that the server key must be readable by the slapd process, but other users should be denied read access. The slapd server on Red Hat Linux Version 8.0 runs as user ldap and group ldap. We distribute the certificate and the key to appropriate locations and fix the permissions:

```
# mv newcert.pem /usr/share/ssl/certs/ldapcert.pem
# chmod og+r /usr/share/ssl/certs/ldapcert.pem
# mv ldapkey.pem /etc/openldap
# chown ldap /etc/openldap/ldapkey.pem
# chmod 400 /etc/openldap/ldapkey.pem
```

The CA certificate that resides in the /usr/share/ssl/demoCA/cacert.pem file must be available to all SSL clients and it should be transferred using ftp or a similar utility. In the case of the OpenSSL clients, just copy the PEM file to a convenient location.

The AIX GS kit client requires a slightly different procedure. The CA certificate should be added to the list of “Signer Certificates.” In the gsk5ikm key management application, create a new key database file. Then choose the **Signer Certificates** section of the database and click on the **Add...** button. The PEM format is a MIME encoded certificate and it may be imported by choosing the “Base64-encoded ASCII data” for data type. Remove all other CA certificates from the key database file. This key database file may be distributed to all AIX clients.

> **Note:** To use SSL or TLS, the server’s name and its IP address must be resolvable through the DNS. It is a requirement because the server’s name stored in its certificate is compared to a result from the DNS reverse lookup operation done on the server’s IP address. This is one of the most common omissions during the SSL setup.

Creating slapd configuration

The slapd configuration on Red Hat Linux Version 8.0 is kept in /etc/openldap/slapd.conf. Example 2-10 on page 42 shows a complete configuration of the slapd server. It is logically divided into four parts: Schema, TLS, back end, and access control. Even though we use term TLS, the OpenLDAP server supports both SSL and TLS connections.

Example 2-10 OpenLDAP /etc/openldap/slapd.conf configuration file

```
include         /etc/openldap/schema/core.schema
include         /etc/openldap/schema/cosine.schema
include         /etc/openldap/schema/inetorgperson.schema
include         /etc/openldap/schema/nis.schema

loglevel        256

TLSCipherSuite          HIGH:MEDIUM:+SSLv3
TLSCertificateFile      /usr/share/ssl/certs/ldapcert.pem
TLSCertificateKeyFile   /etc/openldap/ldapkey.pem
TLSCACertificateFile    /usr/share/ssl/demoCA/cacert.pem
security                tls=112

database        ldbm
suffix          "dc=weeorg,dc=com"
rootdn          "cn=admin,dc=weeorg,dc=com"
rootpw          {SSHA}voNuRGGB+LaiwCZ8SRbBCqOlBnOBeG1W
password-hash   {crypt}
directory       /var/lib/ldap
index           objectClass             eq
index           uid,uidNumber,gidNumber eq
index           cn                      eq,subinitial

access to attribute=userPassword
        by dn="cn=admin,dc=weeorg,dc=com" write
        by self write
        by anonymous auth
access to *
        by dn="cn=admin,dc=weeorg,dc=com" write
        by self read
        by * read
defaultaccess read
```

Note: The slapd.conf file contains sensitive information, in particular, the server administrator's password. Even though the password is encrypted, slapd.conf should be well protected and readable by root only. The authentication data is extremely sensitive and having directory administrator privileges stolen may have disastrous consequences.

We explain, in part, the OpenLDAP configuration in 2.3.2, "The OpenLDAP directory server" on page 26. You can find more information in the slapd.conf(5) manual page and *The SLAPD Administrator's Guide*, available at:

http://www.umich.edu/~dirsvcs/ldap/doc/

Migrating users

The openldap-servers RPM package for Red Hat Linux Version 8.0 includes a set of utilities in the /usr/share/openldap/migration directory. These migration utilities originate from the home of the pam_ldap and nss_ldap software packages:

http://www.padl.com/

Before running the migration utilities, you have to edit the migrate_common.ph script and set the $DEFAULT_BASE variable to the DN suffix. Also, do not forget to edit the passwd.ldif and group.ldif files and remove all system users and groups. The utilities support migrating all standard NIS maps, but we migrate only the user and group databases. The procedure is shown in Example 2-11. The **slapadd** command creates the database files and you must make sure that the slapd server is not running before using it. Otherwise, the database files may be corrupted and the slapd server would be useless.

> **Note:** Prior to migrating users, you should set the file creation mask to 066, because files with sensitive information like user passwords are created.

Example 2-11 Migrating the user and group information on Linux

```
# cd /usr/share/openldap/migration
# ./migrate_base.pl > /var/lib/ldap/top.ldif
# ./migrate_passwd.pl /etc/passwd > /var/lib/ldap/passwd.ldif
# ./migrate_group.pl /etc/group > /var/lib/ldap/group.ldif
# cd /var/lib/ldap
# vi passwd.ldif group.ldif # remove users and groups
# cat top.ldif passwd.ldif group.ldif |
        slapadd -b "dc=weeorg,dc=com"
# chown -R ldap /var/lib/ldap
# chmod -R og-rw /var/lib/ldap
```

Once the user and group information is stored in the LDAP server, we can start it. The slapd server on Red Hat Linux is configured by default to listen on both the ldap (389) and ldaps (636) ports, so both TLS and SSL connections are supported. Try to use the ldapsearch program to make a simple test:

```
# service ldap start
$ ldapsearch -x -LLL -h r-linux.weeorg.com -b dc=weeorg,dc=com uid=joe
ldap_bind: Confidentiality required
        additional info: TLS confidentiality required
# ldapsearch -x -ZZ -LLL -h r-linux.weeorg.com -b dc=weeorg,dc=com uid=joe
dn: uid=joe,ou=People,dc=weeorg,dc=com
uid: joe
cn: Joe R. User
objectClass: account
```

```
objectClass: posixAccount
objectClass: top
objectClass: shadowAccount
shadowLastChange: 11990
shadowMax: 99999
shadowWarning: 7
loginShell: /bin/bash
uidNumber: 501
gidNumber: 100
homeDirectory: /home/joe
gecos: Joe R. User
```

As you can see, the LDAP server does not allow non-encrypted connections. Note that the userPassword attribute is not present in the output. That is because of the access control, which we impose at the LDAP server. In order to see the password, we must authenticate to the server:

```
# ldapsearch -x -ZZ -LLL -D "uid=joe,ou=people,dc=weeorg,dc=com" -W \
   uid=joe userpassword
Enter LDAP Password:
dn: uid=joe,ou=People,dc=weeorg,dc=com
userPassword:: e2NyeXB0fXpjbmtNRO5yTGxoUGc=
```

The reason for the strange userPassword value format is that it could contain binary data and is therefore MIME encoded. Use a MIME utility such as mimencode to decode the value.

2.6.2 AIX LDAP client setup

The AIX LDAP client may cooperate with the OpenLDAP server. Some persuasion is needed, but it is not overly complicated. We start with **mksecldap**:

```
# mksecldap -c -h r-linux.weeorg.com -a cn=admin,dc=weeorg,dc=com -p secret \
        -S rfc2307 -d dc=weeorg,dc=com -k /etc/security/ldap/key.kdb -u NONE
```

The -u option prevents the program from configuring users in the /etc/security/user to use LDAP. We prefer to do it manually, on a per user basis:

```
# chuser SYSTEM=LDAP registry=LDAP joe
```

We need just a few more adjustments in the /etc/security/ldap/ldap.cfg file. Edit the file using a text editor such as vi and make sure that the contents matches the one we show in Example 2-12 on page 45.

Example 2-12 Contents of /etc/security/ldap/ldap.cfg

```
ldapservers:r-linux.weeorg.com
ldapadmin:cn=admin,dc=weeorg,dc=com
ldapadmpwd:secret
useSSL:yes
ldapsslkeyf:/etc/security/ldap/key.kdb
userattrmappath:/etc/security/ldap/2307user.map
groupattrmappath:/etc/security/ldap/2307group.map
userbasedn:ou=People,dc=weeorg,dc=com
groupbasedn:ou=Group,dc=weeorg,dc=com
userclasses:account,posixaccount,shadowaccount
groupclasses:posixgroup
ldapversion:3
ldapport:389
ldapsslport:636
```

The /etc/security/ldap/key.kdb file is the key database we created previously. It must include our local CA certificate in order to be able to verify the LDAP server's identity. Now we can start the secldapclntd client side LDAP caching daemon:

```
# restart-secldapclntd
```

If everything went all right, you can try to connect to this host and the authentication will be done through LDAP.

Note: The AIX LDAP authentication client has to have elevated privileges on the LDAP server. This has the added convenience that many user management commands will continue to function, as with local files. The price for this convenience is that the non-encrypted LDAP administrator password is in a file on every client host. This is a questionable security practice, especially if there are many clients, thus multiplying the risk of compromising the LDAP administrator credentials. See 2.5.3, "PAM and NSS LDAP modules on AIX" on page 36 for an alternative authentication setup.

2.6.3 Linux LDAP client setup

Red Hat Linux features a very useful utility called authconfig. It can resolve a number of difficult tasks:

```
# authconfig --kickstart --enableshadow --enablemd5 \
   --enableldap --enableldapauth --enableldaptls \
   --ldapserver r-linux.weeorg.com --ldapbasedn dc=weeorg,dc=com
```

Some interesting excerpts from the relevant files are shown in Example 2-13 on page 46. The /etc/ldap.conf file is read by both PAM and NSS LDAP modules.

Though authconfig does a good job, it still cannot use multiple LDAP servers or SSL certificates. The /etc/ldap.conf sample in Example 2-13 shows the changes in the last four lines.

Example 2-13 Linux LDAP client configuration

```
/etc/pam.d/system-auth:
   #%PAM-1.0
   # This file is auto-generated.
   # User changes will be destroyed the next time authconfig is run.
   auth        required      /lib/security/pam_env.so
   auth        sufficient    /lib/security/pam_unix.so likeauth nullok
   auth        sufficient    /lib/security/pam_ldap.so use_first_pass
   auth        required      /lib/security/pam_deny.so

   account     required      /lib/security/pam_unix.so
   account     [default=bad success=ok user_unknown=ignore \
      service_err=ignore system_err=ignore] /lib/security/pam_ldap.so

   password    required      /lib/security/pam_cracklib.so retry=3 type=
   password    sufficient    /lib/security/pam_unix.so nullok use_authtok md5
   shadow
   password    sufficient    /lib/security/pam_ldap.so use_authtok
   password    required      /lib/security/pam_deny.so

   session     required      /lib/security/pam_limits.so
   session     required      /lib/security/pam_unix.so
   session     optional      /lib/security/pam_ldap.so

/etc/nsswitch.conf:
   passwd:     files ldap
   shadow:     files ldap
   group:      files ldap

/etc/ldap.conf:
   host r-linux.weeorg.com
   base dc=weeorg,dc=com
   ssl start_tls
   tls_cacertfile /usr/share/ssl/demoCA/cacert.pem
   tls_checkpeer yes
   pam_password crypt
```

Incidentally, the AIX LDAP client connects to port 636 using LDAP over SSL, while the PAM LDAP client uses TLS on the standard LDAP TCP port (389).

2.7 Security considerations

Centralized user management and authentication raise a few security concerns. We list here the most important changes to the environment:

- Users authenticate at one source.
- Users' authentication information is stored in one place.
- The authentication information is transferred over the network for LDAP replication and LDAP authentication requests.

A single-source sign-on is convenient not only for managers, but also for users. They do not have to think about different passwords or expirations; there is always just one password and just one identity. Therefore, password management is much easier. Convenience somehow never reflects well on security (we are still looking for an explanation). The problem is twofold:

- Applications may differ widely in security requirements. Some applications may require that user present different kind of credentials, for example, a password and a PIN. In that case, either the application has to be coerced to require only a password, or it has to be moved to another authentication realm. Neither scenario is a good solution.
- The other problem is that data accessed is seldom of the same importance. Some may be extremely sensitive. If the user's password is compromised, access to all resources is compromised. As there is just one password, there is just one level of security. Again, there is no good solution.

2.7.1 Host access control

There exists, however, a crude access control on a per user basis and on the host level. An LDAP user entry may have additional attributes that specify hosts to which the user is allowed access. Both the AIX LDAP client and the PAM LDAP module support this kind of access control, but in a different manner.

AIX 5L Version 5.2 introduces a couple of attributes for the purpose: hostsdeniedlogin and hostsallowedlogin. Their meanings are evident. The values these attributes can take may be a list of host names or subnets. Then, if the client host finds itself in the hostsallowedlogin attribute, user is granted access. Otherwise, if the host is in the hostsdeniedlogin list, user is denied access. Both attributes may take multiple values. For a full explanation, see the "LDAP exploitation of the security subsystem" section in the *AIX 5L Version 5.2 Security Guide*.

On the other hand, PAM LDAP utilizes only one attribute: host from the account object class. If the pam_check_host_attr option in the configuration file is set, this attribute controls the host access in the account PAM phase. It is multi valued,

but values may be only host names or an asterisk, which is a wildcard character meaning that access is allowed to any host. Therefore, this kind of access control may be difficult to arrange if access is granted for most but not all hosts.

If the host access control on a per user basis is desired, and an RFC2307 style authentication and the AIX LDAP authentication clients are in use, then the /etc/security/ldap/2307user.map attribute map file must be edited to include mapping from hostsallowedlogin to the host attribute:

```
hostsallowedlogin       SEC_LIST        host    m
```

The hostsdeniedlogin attribute has no matching attribute. However, never set “*” for a host attribute, because AIX LDAP client is not going to treat it the same way as PAM LDAP.

2.7.2 LDAP servers access and database backup

LDAP servers store and have full access to the authentication information. They should be running up to date software, with minimal installations, and limited access. The only access that should be allowed is for the management, backup, and data replication. LDAP servers should be physically secure as well.

The LDAP servers backup tapes should be accessible only to system administrators. It may be even better to separate backup of the LDAP directory from the rest of data. Typically, directory data is small and compresses well, so a small size media such as floppy diskette should suffice.

It is worth noting that the actual backup procedure depends on the back-end database requirements. For example, if the OpenLDAP server uses the Berkeley DB, it is recommended to bring it down before performing backup. Naturally, backup practice for the DB2 databases should be followed in case of the IBM Directory Server.

2.7.3 Encryption and PKI

We show all information paths in the centralized authentication environment in Figure 2-8 on page 49. Apart from the path marked “remote access”, all other paths are specific to this environment.

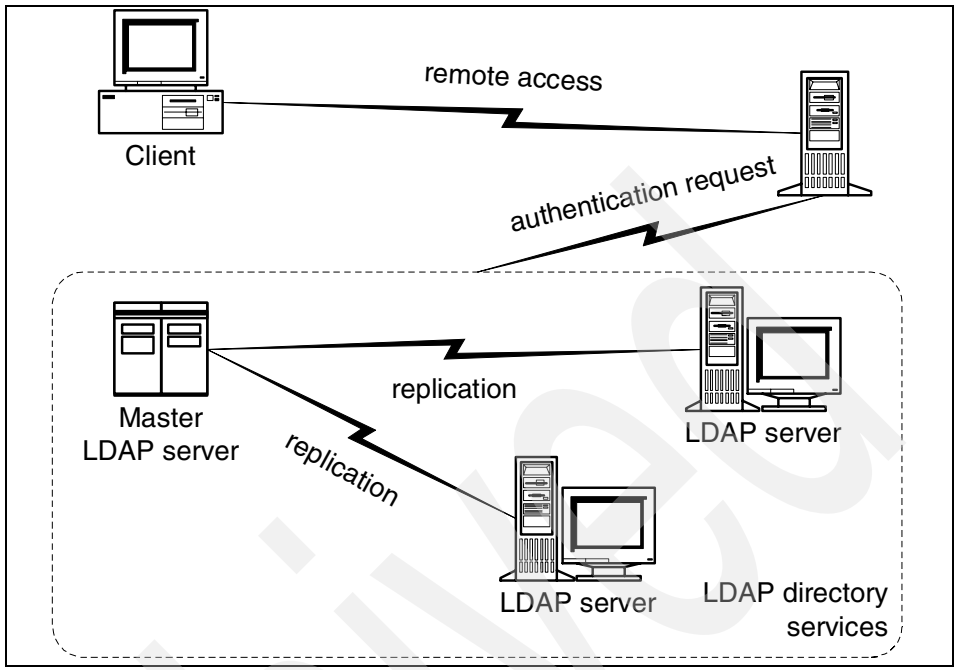

Figure 2-8 LDAP authentication environment

The LDAP protocol supports public key style cryptography (SSL/TLS) for secure connections. Having encrypted communication is important, and with SSL, we are well protected. However, in the case of LDAP authentication, it is as important to be sure about the other party's identity, or we could be giving away passwords to just about anybody. Public Key Infrastructure (PKI) offers a mechanism to establish identity, but for all practical purposes, it is lacking in many respects. We list here the two main problems:

- An SSL certificate contains a public key and a name. It is signed by a Certificate Authority (CA). In other words, there is a third (hopefully) trusted party to claim that the certificate is good and that it really belongs to the named organization. Hence, the certificate validity depends on how well the CA is doing its business. If a server offers a self-signed certificate, one cannot conclude anything, unless that certificate is available in the client's certificate database.
- There is also a private key that corresponds to the public key contained in the certificate. The private key should be known only to its owner. Like anything else, private keys may be compromised. In that case, the corresponding certificate should be *revoked*. Every CA keeps a list of revoked certificates that is made publicly available. Some SSL clients, such as OpenSSL, do not have the capability to check the CRL for revoked certificates. At any rate, it is

simply impractical to check the Certificate Revocation List (CRL) whenever an SSL connection is to be established, because the communication and processing overhead significantly hinder performance.

Given that all LDAP servers are controlled by the same management team, we may be able to deal with these PKI deficiencies. The main point is that users have no responsibilities regarding the PKI management. The management staff is hopefully knowledgeable and diligent enough to keep up with the PKI.

The PKI we recommend is to create a local certificate authority and use it to sign the server certificate requests. Note that using certificates for authentication is quite a bit different from using certificates on the Internet. It simply makes no sense for authentication clients to trust anybody with the certificate issued by some commercial CA. Therefore, certificates for authentication purposes have to be prepared within the organization. Furthermore, clients should trust only certificates signed by the local CA. In other words, all commercial root CA certificates must be removed from clients' certificate databases.

If one of the servers' private keys is compromised, the only solution is to replace the certificate databases on all hosts. Since not all SSL clients have a capability to use the Certificate Revocation Lists (CRL), it is of no use to just revoke the certificate. Even if they were able to consult the CRL, the performance would probably be so sluggish that you would soon turn the feature off. Anyway, if the CRL is not used, the only option is to create a new CA, produce new server certificate requests, sign new server certificates, and replace old certificates with the new ones.

This sounds like a lot of work and, if done manually, it would definitely be. Still, one may be able to automate the process to a reasonable extent. For example, all clients of the same type get the same copy of the key database and it is possible to organize file transfer in a batch mode. The most time consuming task is producing certificates for servers and restarting the LDAP servers, but the number of servers is usually small.

3

Single sign-on

To computer users, it is only logical to prove their identity to the system once. It is also annoying to type your user name and password repeatedly whenever there is a need to use another application. And it becomes unmanageable when applications require different kinds of identities and authentication methods. Just take the Internet as an example; proliferation of services and companies wishing to know our names and habits would make the whole mess unusable if it were not for browsers' capabilities to save all these identities on our computer. Even though it is a poor example, that is one example of a single sign-on. The browser identifies us to various services, thus relieving the user from remembering numerous user names and passwords. We say that it is poor because our browsers and computers are typically poorly managed and secured, though users are not to be solely blamed.

The single sign-on paradigm is a direct consequence of incompatible applications and identification methods. A fair question to ask is how we got here. The answer is probably somewhere between human nature, poor planning, and market pressure. Worse, security and authentication are often given little attention and implemented late and hastily in a project. Finally, a single sign-on system that supports all applications is very difficult, if not impossible, to get right and usually expensive to deploy.

Though they sound similar, single-source sign-on, which we described in Chapter 2, "Centralized user management" on page 15, and single sign-on (SSO) are two very different animals. While a migration of the environment from

standard authentication methods to a single-source sign-on is smooth and non-intrusive, the single sign-on is not only an effort on the part of system administrators, but also on the users' side, sometimes requiring even a user training program. A heavy client involvement in the SSO infrastructure and the incurred price are the main reasons for a very slow adoption of SSO environments.

The idea behind single sign-on is that users prove their identity once and from there on their credentials are automatically forwarded to whichever application they want to use. There are several ways to achieve this. The one which we describe in this chapter is the broker-based Kerberos.

Kerberos has a long history and it has been proven to be a solid technology. One weak point is that in order to join the Kerberos environment, applications must be Kerberized. In other words, they have to be taught Kerberos protocols. The standard remote access (telnet and rlogin) and file transfer (ftp) protocols are supported. There is optional support for encryption. Kerberos is available for free and runs on most platforms, including both AIX and Linux.

3.1 The Kerberos way

Kerberos is a broker based single sign-on technology. This means that there is an authentication server running somewhere on the network maintaining a central storage of user credentials. There is also a specialized protocol for moving security information around. Both servers and clients have to know this protocol.

The Kerberos server maintains a database of principals and corresponding passwords. The term *principal* is a part of the Kerberos semantics, and it may represent any subject involved in communication, such as a user or an application server. A set of principals is organized in a realm that is typically a domain name in uppercase letters. For example, the ftp/r-linux.weeorg.com@WEEORG.COM service (in other words, the ftp service running on host r-linux.weeorg.com) and user joe@WEEORG.COM are both principals belonging to the WEEORG.COM realm.

Another important term to get acquainted with is *ticket*. All authentication in Kerberos is based on tickets. At the start of a session, a user gets a kind of a superticket called the Ticket Granting Ticket (TGT). A TGT is obtained from the Kerberos server by passing a principal name. The ticket is encrypted using the user's password and only this user can decrypt it.

In order to prevent password guessing attacks, the Kerberos server may require users to pre-authenticate before getting the TGT. The Kerberos server sends a time stamp encrypted with the user's password and if the user can decrypt it and send back the time stamp, it may get the TGT in return. That is one commonly used pre-authentication method, but another may be employed depending on your needs.

With this ticket, the user can get other tickets with which he may communicate with network services, such as login or ftp servers. Naturally, all this happens behind the stage, without any user intervention. For more details on how Kerberos (really) works visit:

http://web.mit.edu/kerberos/www/

We show here one Kerberos session:

```
$ kinit
Password for joe@WEEORG.COM:
$ klist
Ticket cache:  FILE:/var/krb5/security/creds/krb5cc_201
Default principal:  joe@WEEORG.COM

Valid starting     Expires            Service principal
11/09/02 19:04:04  11/10/02 05:03:44  krbtgt/WEEORG.COM@WEEORG.COM
```

The kinit program obtains the TGT. It is good for the next 10 hours, as we see in the output of the klist program above. There are other ways to obtain a TGT, such as the login service: When a user logs in to their terminal, the login program automatically obtains the ticket.

Now we can use all the services that are registered with the Kerberos server. The business of obtaining service tickets is done by the Kerberos clients without bothering the user, as in the following:

```
$ rlogin r-linux.weeorg.com -l joe
Last login: Sat Nov  9 18:57:01 from r-aix
[joe@r-linux joe]$ exit

$ telnet -l joe r-linux.weeorg.com
Trying...
Connected to r-linux.weeorg.com.
Escape character is '^]'.
[ Kerberos V5 accepts you as ``joe@WEEORG.COM'' ]

    r-linux.weeorg.com (Linux release 2.4.18-14 #1 Wed Sep 4 13:35:50 EDT 2002)
(2)

Last login: Sat Nov  9 19:04:29 from r-aix
[joe@r-linux joe]$
```

The exchange between the Kerberos servers, services, and clients is done in a secure way.

Note: Kerberos relies on time stamps to protect from replay attacks. Therefore, all participants in Kerberos authentication (clients and servers) must have their clocks synchronized. To tell the truth, a skew of five minutes is allowed, but we see no reason for not deploying the Network Time Protocol (NTP) on the network. It is very convenient to have clocks synchronized in many other situations.

3.2 Kerberos configuration

The Kerberized environment consists of a Kerberos Distribution Center (KDC) and Kerberized clients. The KDC is represented by the master KDC server and one or more slave KDC servers. It is not obligatory to install the slave servers, but we strongly recommend having the KDC service available at all times. Another important role in a Kerberos environment is that of the administration server. This server is called kadmind, and it runs on the same host that keeps the master database. Apart from offering management services, kadmind also allows for password change requests coming from plain users.

3.2.1 Kerberos configuration files

All Kerberos servers and clients have at least one configuration file: /etc/krb5.conf. This file describes the Kerberos environment. We give three examples in Example 3-1.

Example 3-1 Kerberos krb5.conf configuration file

```
Server krb5.conf:
   [logging]
    kdc = FILE:/var/log/krb5kdc.log
    admin_server = FILE:/var/log/kadmind.log
    default = FILE:/var/log/krb5libs.log

   [libdefaults]
    default_realm = WEEORG.COM
    ticket_lifetime = 24000

   [realms]
    WEEORG.COM = {
     kdc = kerberos.weeorg.com:88
     kdc = kerberos-1.weeorg.com:88
     admin_server = kerberos.weeorg.com:749
     default_domain = weeorg.com
    }

   [domain_realm]
    .weeorg.com = WEEORG.COM
    weeorg.com = WEEORG.COM

Client krb5.conf (with dns support):
   [libdefaults]
    dns_lookup_realm = true
    dns_lookup_kdc = true

Client krb5.conf (without dns support):
   [libdefaults]
    default_realm = WEEORG.COM

   [realms]
    WEEORG.COM = {
     kdc = kerberos.weeorg.com:88
     kdc = kerberos-1.weeorg.com:88
     admin_server = kerberos.weeorg.com:749
     default_domain = weeorg.com
    }

   [domain_realm]
    .weeorg.com = WEEORG.COM
```

```
    weeorg.com = WEEORG.COM
```

The krb5.conf file consists of sections that you may recognize by names enclosed in brackets. The *libdefaults* section defines the site-wide defaults, such as realm or ticket issuing details. The most important one is the name of the realm. Kerberos applications use the default realm name to form principal names. For example, we can reference principal joe@WEEORG.COM by specifying only the part to the left of the @ sign. In some Kerberos versions, it is possible to use DNS to lookup the default realm name (see 3.6, “Discovering Kerberos services” on page 72).

The *realms* section specifies locations of Kerberos servers for one or more realms. Every realm is represented by a tag, which is the realm name and a list of service locations. In our example, the KDC services are available at hosts kerberos and kerberos-1 and the administration server runs on the host kerberos. We can conclude that kerberos.weeorg.com is the master server. The default_domain tag is used to convert host names in Kerberos Version 4 principals to the FQDN host names.

The *domain_realm* section defines mapping between host names and realms. If the domain tag begins with a dot, which means that all names ending with this domain name belong to the specified realm. Alternatively, the new DNS method may be used to discover the realm name to which a host belongs (see 3.6, “Discovering Kerberos services” on page 72).

The *logging* section defines where Kerberos applications send error messages. Several destinations are supported, including plain files, devices, and syslog. The syslog destination may be used for centralized logging. For example, the following line specifies that the kdc messages starting at the notice severity level are to be sent to the LOG_AUTH syslog facility:

```
kdc = SYSLOG:NOTICE:AUTH
```

Finally, the *kdc* section may specify the kdc.conf file location. We show one in Example 3-2. Whereas krb5.conf is read by every Kerberized program, kdc.conf is used by the KDC and administration server only on the master server.

Example 3-2 The kdc.conf file

```
[kdcdefaults]
    kdc_ports = 88

[realms]
    WEEORG.COM =  {
        admin_keytab = /var/krb5/krb5kdc/kadm5.keytab
        acl_file = /var/krb5/krb5kdc/kadm5.acl
        dict_file = /var/krb5/krb5kdc/kadm5.dict
```

```
        key_stash_file = /var/krb5/krb5kdc/.k5.WEEORG.COM
        kadmind_port = 749
        kdc_ports = 88
        max_life = 10h 0m 0s
        max_renewable_life = 7d 0h 0m 0s
        master_key_type = des-cbc-crc
        supported_enctypes = des3-cbc-sha1:normal des-cbc-md5:normal
des-cbc-crc
:normal
        kdc_supported_enctypes = des3-cbc-sha1:normal des-cbc-md5:normal
des-cbc
-crc:normal
    }
```

3.2.2 Kerberos database

Kerberos maintains a database of principals and their passwords. All participants in the Kerberos authentication system are principals.

The database must be initialized before any data is stored in it:

```
# kdb5_util -r WEEORG.COM create -s
Initializing database '/var/kerberos/krb5kdc/principal' for realm 'WEEORG.COM',
master key name 'K/M@WEEORG.COM'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:
Re-enter KDC database master key to verify:
```

This command creates the master Kerberos key, several administration principals, and the krbtgt principal. This way, the Kerberos database is bootstrapped. When the -s option is specified, the kdb5_util program stores the master key in the stash file:

```
# kadmin.local -q "ktadd -k /var/kerberos/krb5kdc/kadm5.keytab kadmin/admin"
Authenticating as principal root/admin@WEEORG.COM with password.
Entry for principal kadmin/admin with kvno 3, encryption type Triple DES cbc
mode raw added to keytab WRFILE:/var/kerberos/krb5kdc/kadm5.keytab.
Entry for principal kadmin/admin with kvno 3, encryption type DES cbc mode with
CRC-32 added to keytab WRFILE:/var/kerberos/krb5kdc/kadm5.keytab.

# kadmin.local -q "ktadd -k /var/kerberos/krb5kdc/kadm5.keytab kadmin/changepw"
...
```

Just like users, Kerberos applications obtain tickets as well. To decrypt them, they use keys. Incidentally, user passwords are also keys, but for some reason we are used to referring to them as passwords. Since applications typically do not use a keyboard, keys are retrieved from files called key tables. For example,

the kadmind administration server is a Kerberos application, which serves requests for principal manipulation and password changes. To be able to do this, it reads all principals and corresponding keys from the kadm5.keytab file. That is why we use the **ktadd** command to store the special kadmin/admin and kadmin/changepw principals in kadm5.keytab.

To access the administration server, we use the admin/admin principal:

```
# kadmin.local -q "addprinc admin/admin@WEEORG.COM"
Authenticating as principal root/admin@WEEORG.COM with password.
WARNING: no policy specified for admin/admin@WEEORG.COM; defaulting to no
policy
Enter password for principal "admin/admin@WEEORG.COM":
Re-enter password for principal "admin/admin@WEEORG.COM":
Principal "admin/admin@WEEORG.COM" created.
```

3.2.3 Controlling access to Kerberos

Kerberos administration server controls access to the Kerberos database using the Access Control List (ACL). The list is kept in a flat ASCII file (kadm5.acl). To start, we create the file with the following line to allow all principals with the admin instance unlimited access:

```
*/admin@WEEORG.COM        *
```

An asterisk in the principal name is a global character standing for any string. An asterisk in the second field is a shortcut for the *admcil* set of privileges. Every letter in this string denotes one type of access:

a	Add principals and policies
d	Delete principals and policies
m	Modify principals and policies
c	Change password of principals and policies
i	Query database (inquire)
l	List principals and policies

For example, the following line allows the user jack to inquire about principals:

```
jack@WEEORG.COM        i
```

The uppercase letters denote that the principal is denied the corresponding access. The order of the ACL is significant and the administration server applies controls for the first match.

The third field in the access control list may specifies a target. Like the first field, the target is a principal specification and may include the * wildcard.

3.2.4 Starting Kerberos

Kerberos has three services: The Key Distribution Center (KDC), the administration service, and the password change service. The first is implemented in the krb5kdc daemon and the latter two in the kadmind daemon. Once we have Kerberos configured, starting the services may be done by simply invoking these two programs on the command line. However, this should be done by invoking another program, depending on the platform.

Starting Kerberos services on AIX

On AIX 5L Version 5.2, both Kerberos daemons are started using the start.krb5 script:

```
# /usr/krb5/sbin/start.krb5
```

To stop the services, use the **stop.krb5** command:

```
# /usr/krb5/sbin/stop.krb5
```

Starting Kerberos services on Linux

On Red Hat Linux Version 8.0, Kerberos services are started or stopped using the standard Red Hat service program:

```
# service krb5kdc start
# service kadmin start
```

Alternatively, you can turn to the somewhat more standard system V type startup scripts:

```
# /etc/init.d/krb5kdc start
# /etc/init.d/kadmin start
```

3.3 Kerberos administration

The Kerberos database consists of principals and policies. The kadmin program is used to administer the database. It connects to the administration server running on the Kerberos master server.

3.3.1 Kerberos principals

The format of a Kerberos principal is:

```
component/component/component@REALM
```

However, commonly, only one or two components are used and the format is:

```
primary/instance@REALM
```

For example, ordinary users have just a single component (primary), which is normally the same as their UNIX user name. Services have two components, of which the primary is the service type and the instance is the host name. Here are a few examples of principal names:

```
joe@WEEORG.COM
joe/admin@WEEORG.COM
host/lettuce.weeorg.com@WEEORG.COM
ftp/weed.weeorg.com@WEEORG.COM
ldap/marguerite.weeorg.com@WEEORG.COM
krbtgt/WEEORG.COM@WEEORG.COM
```

The first two are principals used by humans. By convention, all principals with the *admin* instance are members of the administration team. In this case, user joe may be the same person as administrator joe, but the use of these two principals is very different. Whereas the first principal refers to an ordinary user, the second is commonly used only with the kadmin program.

The next three principals are Kerberos services. The *host* service refers to the low level remote access protocol, such as telnet or rlogin. Do not forget that Kerberos does not distinguish application servers from users: A principal for *every* Kerberized service must be defined in the Kerberos database. Furthermore, the keys have to be accessible to the processes that implement the services. Just like users who give passwords to prove their identities, application servers pass keys read from the key table for the same purpose. Key table files should be readable by the root user only. In case the service process is running with another user ID, it is best to create a separate key table file that will hold just this one principal and be owned by the same user ID.

The last principal krbtgt/WEEORG.COM, which refers to the Kerberos TGT, is a special ticket granting principal used by the KDC exclusively.

The kadmin program is used to administrate principals and key tables. It has the same functionality like the kadmin.local program. Normally, the local variant is used only to bootstrap the database. Let us see how to create principals:

```
# kadmin -p admin/admin
Authenticating as principal admin/admin with password.
Enter password:
kadmin:  addprinc joe/admin
WARNING: no policy specified for joe/admin@WEEORG.COM;
  defaulting to no policy. Note that policy may be overridden by
  ACL restrictions.
Enter password for principal "joe/admin@WEEORG.COM":
Re-enter password for principal "joe/admin@WEEORG.COM":
Principal "joe/admin@WEEORG.COM" created.
```

Note that we connect using the admin/admin principal in order to be able to manipulate principals. Now we create some host keys:

```
kadmin:  addprinc -randkey host/lettuce.weeorg.com
WARNING: no policy specified for host/lettuce.weeorg.com@WEEORG.COM;
  defaulting to no policy. Note that policy may be overridden by
  ACL restrictions.
Principal "host/lettuce.weeorg.com@WEEORG.COM" created.
```

The -randkey option of the **addprinc** command is a way to ask the program to generate a random key instead of asking for a password. The processes on the host must have this key available, so we start kadmin on host lettuce and run the following command:

```
kadmin:  ktadd host/lettuce.weeorg.com
Entry for principal host/lettuce.weeorg.com with kvno 6, encryption type Triple
DES cbc mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal host/lettuce.weeorg.com with kvno 6, encryption type DES
cbc mode with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
```

If, for some reason, it is not possible to run kadmin on the target host, we may produce the key table file on any host and transfer it securely to the target host:

```
kadmin:  ktadd -k /tmp/krb5.keytab-lettuce host/lettuce.weeorg.com
Entry for principal host/lettuce.weeorg.com with kvno 3, encryption type Triple
DES cbc mode with HMAC/sha1 added to keytab WRFILE:/tmp/krb5.keytab-lettuce.
Entry for principal host/lettuce.weeorg.com with kvno 3, encryption type DES
cbc mode with RSA-MD5 added to keytab WRFILE:/tmp/krb5.keytab-lettuce.
kadmin: exit
# scp -p /tmp/krb5.keytab-lettuce joe@lettuce:/tmp/krb5.keytab
# rm -f /tmp/krb5.keytab-lettuce
lettuce# mv /tmp/krb5.keytab /etc
lettuce# chown root /etc/krb5.keytab; chmod 600 /etc/krb5.keytab
```

You have to place the key table file in a location where applications are expecting it. On AIX 5L Version 5.2, it is /etc/krb5/krb5.keytab, and on “vanilla” Kerberos implementations, including Linux, it is usually /etc/krb5.keytab. Key tables may be managed using the ktutil program.

3.3.2 Kerberos policies

You may have noticed in the output produced by kadmin that a policy has been mentioned several times. A policy is a set of attributes that govern the password quality. Every principal in the Kerberos database is assigned a policy. This is a list of password attributes:

maxlife	Maximum lifetime
minlife	Minimum lifetime
minlength	Minimum length

minclasses	Minimum number of character classes
history	Number of passwords kept in a principal history

Policies are created, deleted, and modified using the kadmin program. We create the default policy with the maximum password age of 180 days, a minimum length of six, and passwords must contain character belonging to at least two different character classes:

```
kadmin:  addpol -maxlife "180 days" -minlength 6 -minclasses 2 default
kadmin:  getpol default
Policy: default
Maximum password life: 15552000
Minimum password life: 0
Minimum password length: 6
Minimum number of password character classes: 2
Number of old keys kept: 1
Reference count: 0
```

The policy named default is assigned to all created principals, if not specified otherwise. If the default policy does not exist, newly created principals will have no policy assigned.

The time specification may be pretty elaborate, as we show in the following examples:

```
“3 days”
“next Sunday”
“Jan 1, 2003”
“12/31/2003 10:00:00 CST”
fortnight
“6 months”
“1 year 6 months”
"next monday 10pm"
```

We will not list all rules, but do not forget to enclose the specification in double quotes if it contains spaces. You may have fun experimenting with odd time specifications. Once swallowed by the software, the time is converted to seconds, so what you type will look very different from what you see:

```
kadmin:  modprinc -expire "next fortnight" host/lettuce.weeorg.com
Principal "host/lettuce.weeorg.com@WEEORG.COM" modified.
kadmin:  getprinc host/lettuce.weeorg.com
Principal: host/lettuce.weeorg.com@WEEORG.COM
Expiration date: Thu Dec 12 11:44:28 CST 2002
Last password change:  Wed Nov 13 18:31:51 CST 2002
Password expiration date: [none]
Maximum ticket life: 0 day 10:00:00
Maximum renewable life: 7 days 00:00:00
Last modified: Thu Nov 14 11:44:28 CST 2002 (admin/admin@WEEORG.COM)
```

```
Last successful authentication: [never]
Last failed authentication: [never]
Failed password attempts: 0
Number of keys: 2
Key: vno 3, Triple DES cbc mode with HMAC/sha1,
no salt
Key: vno 3, DES cbc mode with RSA-MD5,
no salt

Attributes:

Policy: [none]
```

As you can see, principals may be set to expire just like their passwords. Furthermore, there is a number of principal attributes, most of which specify the nature of tickets issued for the principal. See the Kerberos reference documentation for details. Incidentally, it is probably not advisable to let service principals expire in any way. Therefore, we recommend creating service principals in the following manner:

```
kadmin:  addprinc -randkey -clearpolicy host/bigserver.weeorg.com
Principal "host/bigserver.weeorg.com@WEEORG.COM" created.
```

3.3.3 Kerberos database management

We use the kdb5_util program to initialize the Kerberos database. This program implements a couple of symmetric pairs of operations that manipulate the database as a whole: Create and destroy, and dump and load. In particular, the **dump** command exports the database to a file that may be transferred to another host and then loaded into another database. Note that the load operation destroys the existing contents of the database.

The **create** command is used only once, in the beginning, to initialize the database. If the database is corrupted for some reason, we must initialize the database and load data from the backup.

It is preferable to perform backup of the Kerberos database separately, because the data is very sensitive and it is sometimes difficult to limit access to the backup media. The Kerberos database is typically small, so it usually fits on a diskette. As for the format, perhaps it is better to use a file from the **kdb5_util dump** command, because it uses the ASCII character set.

3.3.4 Kerberos database replication

Kerberos deploys a very simple replication method. The database is dumped using the kdb5_util program, and the resulting file is transferred to the slave

server and loaded into the target database. All this is usually done from a cron job. The rule of thumb is to run the procedure on an hourly basis, but that depends on your needs and the database size. The kprop client and the kpropd server manage the database transfer. Sending a dump file to the host running the slave Kerberos server is simple:

```
# kprop -f dump_file slave_server
```

Of course, the kpropd daemon must be running on the slave server host. It is usually through the inetd superserver, so we put a line like this one in /etc/inetd.conf:

```
krb5_prop stream tcp nowait root /usr/sbin/kpropd kpropd
```

The kpropd daemon receives the database, stores it in a file, and then runs the kdb5_util program with the **load** command to update the active database.

3.4 AIX Network Authentication Service (NAS)

AIX 5L Version 5.2 has full support for Kerberos. The telnet, “r” commands, and ftp services have been Kerberized for both servers and clients. Actually, they are equipped with more than one authentication method.

The system administrator uses the chauthent program to choose authentication methods and the order of precedence. In this example, it is Kerberos Version 5 followed by the standard authentication method:

```
# chauthent -k5 -std
# lsauthent
Kerberos 5
Standard Aix
```

For example, the ftp client will try each of the listed methods in the given order.

AIX 5L Version 5.2 includes the Network Authentication Service Version 1.3 on the Expansion Pack CD. It is a Kerberos server implementation compliant with the Kerberos protocol described in RFC1510: The Kerberos Network Authentication Service (V5). It sports several additions and a particularly interesting extension is the LDAP directory plug-in. On the one hand. this plug-in uses an LDAP server to store the Kerberos database of principals and policies. On the other hand, Kerberos clients may use LDAP service to look up the KDC and administration servers.

3.4.1 Installing required packages

The following packages include the Kerberos server, some clients, and the development kit:

```
krb5.client.rte
krb5.doc.en_US.html
krb5.doc.en_US.pdf
krb5.msg.en_US.client.rte
krb5.server.rte
krb5.toolkit.adt
```

You can install them from the expansion pack CDROM using the installp program:

```
# installp -acgXYd src krb5.client krb5.server krb5.toolkit krb5.msg.en_US \
>       krb5.doc.en_US
```

3.4.2 AIX Kerberos master server

There are two ways to set up the Kerberos master server, depending on the preferred database location: A local file system or an LDAP server. It may not be easy to decide which path to take.

On the plus side, using the file system to store the database is the way the “vanilla” MIT Kerberos works, so there should be less interoperability issues. It is also very easy to manage and the environment is quite simple. Typically, there are no performance issues, because database updates seldom occur, and KDC servers perform read and cryptography operations only when dealing with tickets that are normally significantly less then the number of user sessions.

On the other hand, LDAP may have already been used to store other user data, as we show in Chapter 2, “Centralized user management” on page 15. This way, both the identification and authentication data are stored in one place. In spite of the LDAP open nature, the administrators should have the expertise to protect the Kerberos data properly. This seems to open many new possibilities and may offer better data control. LDAP servers have their own way to replicate directories in a synchronous manner. Kerberos has a very simple replication mechanism which, depending on your environment, may be an issue.

If you choose to use LDAP to store Kerberos data, then all KDC servers must know how to get to it. Currently, there is no way to export the Kerberos data in the format suitable for the kdb5_util program. The effect of this is that you can use only the AIX platform to host Kerberos servers.

There exists, however, an LDAP back-end for the Heimdal Kerberos package and it is available from:

http://www.padl.com/Research/Heimdal.html

This back-end uses an LDAP scheme different from the one installed by AIX Network Authentication Service (NAS). It may prove to be very difficult to adapt one or the other package in such a way they can use the same schema and share data.

NAS with LDAP support

Prior to dealing with Kerberos, the LDAP server has to be prepared. AIX NAS officially supports two brands of LDAP servers: IBM Directory Server and SunONE Directory Server. We show how to prepare the IBM Directory Server. The general idea is to have the LDAP server understand the Kerberos data by loading an appropriate set of schemas. The next task is to create a realm object, which is going to be the top of the Kerberos tree.

Refer to 2.3.1, “IBM Directory Server” on page 22 for information on how to install the IBM Directory Server. The server should be properly configured using the ldapcfg program and running. To continue our example, we will use the dc=weeorg,dc=com suffix, as in Chapter 2, “Centralized user management” on page 15.

Loading Kerberos schema definitions

The schema description is in the IBM.KRB.schema.ldif file in the /usr/krb5/ldif directory. We need to feed this file to the LDAP server using the ldapmodify program:

```
# cd /usr/krb5/ldif
# ldapmodify -c -h host -D rootdn -w secret IBM.KRB.schema.ldif 2>/tmp/errs
```

The host, rootdn, and secret parameters refer to the LDAP server host, the administrator’s DN and password.

> **Note:** It may happen that certain attributes, such as userPassword or uid are already present in the server. By using the -c option, we ask the ldapmodify program not to stop on error. However, you should examine the error output to see if there were other errors reported.

Note that there is no need to use the standard Kerberos database replication method; the LDAP replication must be set up and all Kerberos data will be automatically propagated to all LDAP slave servers. It is practical to have all LDAP slave servers prepared for Kerberos and replication. In that case, you will have Kerberos data available in slave LDAP servers at the time of configuring the

master server. We suggest that you apply the same schema changes on the slave LDAP servers at this time. That way, the Kerberos realm that we create in the following text is automatically replicated to the slave servers. You should verify that the replication is in good shape.

Creating the Kerberos realm

The Kerberos realm is the top of the Kerberos data tree. It is an object described in the KrbRealm-V2 and KrbRealmExt object classes. A template file called realm_add.ldif is in the same directory as the schema file. Copy this file to another name, such as ourrealm.ldif, and edit it to reflect your Kerberos configuration. We show how our realm in the LDIF format appears in Example 3-3. Then, use the **ldapadd** command to put the data in the directory:

```
# ldapadd -h host -D rootdn -w secret ourrealm.ldif
```

Example 3-3 Kerberos realm in the LDIF format

```
dn: krbrealmName-V2=WEEORG.COM,dc=weeorg,dc=com
objectclass: KrbRealm-V2
objectclass: KrbRealmExt
krbrealmName-V2: WEEORG.COM
krbprincSubtree: krbrealmName-V2=WEEORG.COM,dc=weeorg,dc=com
krbDeleteType: 4

dn: cn=principal, krbrealmName-V2=WEEORG.COM,dc=weeorg,dc=com
objectclass: container
cn: principal

dn: cn=policy, krbrealmName-V2=WEEORG.COM,dc=weeorg,dc=com
objectclass: container
cn: policy
```

This command may fail if there is no data for the suffix. If that happens, insert the following snippet at the top of the LDIF file:

```
dn: dc=com
dc: com
objectClass: top
objectClass: domain

dn: dc=weeorg,dc=com
dc: weeorg
objectClass: domain
```

Of course, you should change it appropriately if your suffix has different attributes or more levels.

Verify that the realm data is in the directory using the ldapsearch program:

```
# ldapsearch -h r-aix.weeorg.com -b dc=weeorg,dc=com objectclass=krbrealm-V2
krbrealmName-V2=WEEORG.COM,dc=weeorg,dc=com
objectclass=KrbRealm-V2
objectclass=KrbRealmExt
objectclass=top
krbrealmname-v2=WEEORG.COM
krbprincsubtree=krbrealmName-V2=WEEORG.COM,dc=weeorg,dc=com
```

The krbDeleteType attribute governs how Kerberos objects are removed from the LDAP directory or, more precisely, to which extent they are removed. This schema and, accordingly, the data it supports is shared with the Distributed Computing Environment (DCE). If that is the case, find out what is the proper setting in the *AIX 5L Version 5.2 NAS Administrator's and User's Guide for AIX*, which can be found in the /usr/lpp/krb5/doc/pdf/en_US/ADMINGD/ADMINGD.PDF file, so that the two packages do not conflict. Otherwise, if NAS is the sole user of data in LDAP, you should set it to 4, or it will cause complications by leaving old principal containers around.

The LDAP server is ready now. As you can see, there are two objects of class container that will "hold" all Kerberos data.

User principals

The previously described Kerberos tree is sufficient to store all principals. However, if there are already users defined in LDAP, as we describe in Chapter 2, "Centralized user management" on page 15, it may be beneficial to store the principal data along with the other user information. The LDAP entry, which is the parent of the users' entries, must be the Kerberos principal sub tree. The krbPrincSubtree attribute may contain multiple values, so we simply add another value containing the DN of the parent. Continuing our example from Chapter 2, "Centralized user management" on page 15, we send the following LDIF file to the ldapmodify program:

```
dn: krbrealmname-v2=WEEORG.COM,dc=weeorg,dc=com
changetype: modify
add: krbprincsubtree
krbprincsubtree: ou=people,dc=weeorg,dc=com
```

The next step is to add the krbPrincipal object class to users' entries and then set the krbPrincipalName attribute to user@REALM, like this:

```
dn: uid=joe,ou=people,dc=weeorg,dc=com
changetype: modify
add: objectclass
objectclass: krbprincipal
-
```

```
add: krbprincipalname
krbprincipalname: joe@WEEORG.COM
-
```

If you prefer the GUI, use the Directory Management Tool (dmt) or similar LDAP utility to do the same task. We show the changes in the LDIF format because it lends itself well to scripting.

After these changes, the kadmin program will store the data for the principal joe in the entry we just modified. This way, we may separate the services' principals from the users' principals, thus making our principal tree cleaner.

Configuring the master server

At this point, you should consider who will administrate the Kerberos database. In previous examples we use the LDAP administrator DN to edit the LDAP data. If the LDAP administrator will be the Kerberos administrator as well, it is alright to continue to use the LDAP administrator DN. Otherwise, you should seriously consider other options for administering the Kerberos database. For example, you may create another LDAP user such as cn=krbadmin,dc=weeorg,dc=com. In that case, you should substitute all occurrences of the cn=root string to the DN of your Kerberos administrator. Do not forget to change the LDAP Access Control List (ACL) on the Kerberos containers (realm, principal, and policy) prior to using the new DN.

AIX uses the config.krb5 program, which does most of the server setup. You may run it without arguments to see the usage. As you can see from the output in Example 3-4, the config.krb5 script does what we discuss in 3.2, "Kerberos configuration" on page 54. It also starts the krb5kdc and kadmind servers. The great difference, however, is that all data is stored in the LDAP server.

Example 3-4 Running the config.krb5 script

```
# config.krb5 -d weeorg.com -r WEEORG.COM \
   -l r-aix.weeorg.com -u cn=root -p secret
Initializing configuration...
Creating /etc/krb5/krb5.conf...
Creating /var/krb5/krb5kdc/kdc.conf...
Creating database files...
Initializing database 'LDAP' for realm 'WEEORG.COM'
master key name 'K/M@WEEORG.COM'
Attempting to bind to one or more LDAP servers. This may take a while...
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter database Master Password:
Re-enter database Master Password to verify:
Attempting to bind to one or more LDAP servers. This may take a while...
WARNING: no policy specified for admin/admin@WEEORG.COM;
  defaulting to no policy. Note that policy may be overridden by
```

```
  ACL restrictions.
Enter password for principal "admin/admin@WEEORG.COM":
Re-enter password for principal "admin/admin@WEEORG.COM":
Principal "admin/admin@WEEORG.COM" created.
Creating keytable...
Attempting to bind to one or more LDAP servers. This may take a while...
Creating /var/krb5/krb5kdc/kadm5.acl...
Starting krb5kdc...
Attempting to bind to one or more LDAP servers. This may take a while...
krb5kdc was started successfully.
Starting kadmind...
Attempting to bind to one or more LDAP servers. This may take a while...
kadmind was started successfully.
The command completed successfully.
```

Even though the Kerberos data is accessible through standard LDAP browsers, you should modify it with Kerberos tools, such as kadmin and ksetup. An important exception to the rule is the LDAP access controls. Regardless of how the LDAP server sets the ACLs initially, you should apply appropriate control, given the sensitivity of the Kerberos data. In particular, both read and write access to the KrbMstrKey and KrbKey object classes should be restricted to the KDCs and the administration server. Note that this is different from the Kerberos ACLs, which are manipulated using the kadmin program.

The configuration script may not set all desired options for LDAP access. For example, if you access LDAP servers in different ways, then you must edit the /var/krb5/krb5kdc/.kdc_ldap_data configuration file using a text editor. We show one possible configuration in Example 3-5. Even though LDAP has a referral mechanism to deal with updates sent to replica servers, it is better to use the replica_type tag and specify whether the server's data may be updated. The Kerberos servers keep connections open to all specified LDAP servers, but the preference tag governs which server is queried. The default preferences are 4 for the master LDAP server and 5 for the slave servers.

Example 3-5 LDAP access configuration in /var/krb5/krb5kdc/.kdc_ldap_data

```
[ldapdefaults]
        realm = WEEORG.COM
        bind_dn = cn=root
        bind_dn_pw = secret
        ldapserver = r-aix.weeorg.com
        ldapserver = ldap-1.weeorg.com
        bind_type = simple

[servers]
        r-aix.weeorg.com = {
                port = 389
```

```
                replica_type = readwrite
                preference = 4
        }
        ldap-1.weeorg.com = {
                bind_dn = cn=krbadm
                bind_dn_pw = secret
                ssl_port = 636
                keyring = /etc/ldap/key.kdb
                replica_type = readonly
                preference = 5
        }
```

At this point, you may use the kadmin program to add principals or define and apply policies.

Note: In case any data changes in the realm, the KDC and administration servers have to be restarted in order for changes to have effect.

The config.krb5 script should not be used to configure the slave KDC servers. However, given that there is no need to propagate the database using Kerberos tools, setting up the slave servers is easy. You should copy /etc/krb5/krb5.conf and /var/krb5/krb5kdc/.kdc_ldap_data from the master server and, optionally, edit them to suit the slave KDC server's needs.

3.5 Linux Kerberos support

As opposed to AIX, Linux sports a "vanilla" MIT Kerberos edition. Heimdal is another Kerberos package available as open source and is free, but we discuss the MIT version. Heimdal is not available from Red Hat, but is included with other distributions, such as Debian Linux. Of course, it may be compiled on any computer running Linux.

3.5.1 Red Hat Linux Kerberos packages

On a Red Hat Linux Version 8.0 host, you should install the following RPM packages:

```
krb5-libs-1.2.5-6
krb5-workstation-1.2.5-6
krb5-server-1.2.5-6
pam_krb5-1.56-1
cyrus-sasl-gssapi-2.1.7-2
```

Note that version numbers may change in the future. Installation is simple; after you obtain the packages, just use the rpm program:

```
# rpm -i cyrus-sasl-gssapi-2.1.7-2.i386.rpm krb5-libs-1.2.5-6.i386.rpm \
        krb5-server-1.2.5-6.i386.rpm krb5-workstation-1.2.5-6.i386.rpm \
        pam_krb5-1.56-1.i386.rpm
```

3.5.2 Configuring Kerberos on Linux

Linux distributions typically do not include any kind of configuration application for Kerberos, Red Hat Linux notwithstanding. Being the standard Kerberos implementation, you should not encounter any problems setting up a Kerberized environment. It is enough to follow the instructions we give in 3.2, “Kerberos configuration” on page 54.

3.6 Discovering Kerberos services

Normally, Kerberos clients rely on the krb5.conf file to discover the Kerberos configuration. A couple of mechanisms were recently introduced that may be used by Kerberos clients to find out the locations of the KDC servers automatically. One is implemented in the original Kerberos source code distribution from MIT and it makes use of the new DNS SRV records. The other one is a part of the AIX NAS and utilizes LDAP for this purpose.

Apart from the convenience of having clients automatically look up services’ locations, this kind of Kerberos clients’ setup is also important as a way to decouple them from the actual environment. For example, if we add another slave Kerberos server, we just need to change the configuration in one place, either the primary DNS server or the master LDAP server, and all the clients will automatically learn about it.

3.6.1 Discovering Kerberos services using AIX NAS and LDAP

If the LDAP plug-in is configured, NAS Kerberos clients may use LDAP to discover KDC and administration servers. Note that it is possible to use LDAP for services discovery if NAS is configured to use the standard (legacy) database on a local file system. The ksetup management program is used to configure the NAS entries in the LDAP server. The following ksetup session describes our environment:

```
# ksetup -h localhost -n cn=root -p secret
ksetup> addkdc r-kerberos.weeorg.com
ksetup> addkdc kerberos-1.weeorg.com
ksetup> addadmin kerberos.weeorg.com
ksetup> addpwd kerberos.weeorg.com
```

```
ksetup> listkdc
  kerberos.weeorg.com:88
  kerberos-1.weeorg.com:88
ksetup> exit
```

The commands that start with add, such as addkdc or addhost, install the names of the computers hosting the corresponding service: kdc for Kerberos Distribution Center (KDC), admin for the administration server (kadmind), and pwd for the password change service (also served by kadmind). As you may have guessed, commands that start with list and del, such as listkdc, listhost, delkdc, and delhost, are available as well. The complete description of the program is in the *AIX 5L Version 5.2 NAS Administrator's and User's Guide for AIX*, which can be found in the /usr/lpp/krb5/doc/pdf/en_US/ADMINGD/ADMINGD.PDF file.

Of course, all programs should know how to access an LDAP server. The config.krb5 script may configure the client hosts as follows:

```
# config.krb5 -C -r WEEORG.COM -d weeorg.com -l ldap.weeorg.com
Initializing configuration...
Creating /etc/krb5/krb5.conf...
The command completed successfully.
```

The script will create the krb5.conf file from scratch and include the following lines, which are of particular interest at the moment:

```
[libdefaults]
   use_ldap_lookup = 1
   ldap_server = ldap.weeorg.com
```

3.6.2 Discovering Kerberos services using DNS

The Kerberos distribution available on Linux includes support for locating the Kerberos service through DNS. Given that the DNS zone is configured properly, every Kerberos client host may be easily set up, as we show in Example 3-6. Nevertheless, one should still specify the administration server in krb5.conf, because support for it has not been completed yet.

Example 3-6 Client krb5.conf file with the DNS discovery support

```
[libdefaults]
 dns_lookup_realm = true
 dns_lookup_kdc = true
```

We provide a fairly complete example of the weeorg.com zone in Example 3-7 on page 74. The _kerberos TXT record resolves to WEEORG.COM, which is the default Kerberos realm. The SRV records represent the locations of various Kerberos services. Furthermore, they illustrate nicely the Kerberos environment. Note that the SRV records must point to the canonical host names.

Example 3-7 The DNS zone with support for Kerberos services

```
weeorg.com.              IN      NS      r-linux.weeorg.com.
;
r-linux                  IN      A       9.3.5.32
ldap                     IN      CNAME   r-linux.weeorg.com.
kerberos                 IN      CNAME   r-linux.weeorg.com.
r-aix                    IN      A       9.3.4.34
ldap-1                   IN      CNAME   r-aix.weeorg.com.
kerberos-1               IN      CNAME   r-aix.weeorg.com.
boticelli                IN      A       9.3.5.25
;
_kerberos                IN      TXT     "WEEORG.COM"
_kerberos-master._udp    IN      SRV     0 0 88  r-linux.weeorg.com.
_kerberos-adm._udp       IN      SRV     0 0 749 r-linux.weeorg.com.
_kpasswd._udp            IN      SRV     0 0 464 r-linux.weeorg.com.
_kerberos._udp           IN      SRV     0 0 88  r-linux.weeorg.com.
                         IN      SRV     0 0 88  r-aix.weeorg.com.
_ldap._tcp.weeorg.com    IN      SRV     0 0 389 r-linux.weeorg.com.
                         IN      SRV     0 0 389 r-aix.weeorg.com.
```

Note: When an important security protocol relies on DNS, then you should make sure that DNS cannot be compromised easily. The Kerberos and DNS combination is not an exception in this respect: DNS corruption may lead to a serious security breach or a denial of service, depending on your luck and their skills. We strongly recommend keeping the name servers in good shape and using DNSSEC for authenticated and secure zone transfers. The latter is a must if the transfers are done over a decidedly untrusted network, such as the Internet.

If you are running a particularly sensitive operation, then it may be better not to use the SRV records at all.

3.7 Integrating Kerberos authentication

We describe how to set up and integrate Kerberos in a heterogeneous environment in this section. The master KDC server is NAS on an AIX 5L Version 5.2 host. The slave KDC server is MIT Kerberos hosted on Red Hat Linux Version 8.0. Given such a division of roles, we use the local file systems to store Kerberos data.

3.7.1 KDC setup

AIX NAS has the config.krb5 configuration script, which may be used to initialize the master KDC and the database. However, you should use the mkkrb5srv

program instead of config.krb5. Whereas the latter configures the system for the native Kerberos authentication, the former takes care of the AIX standard authentication system as well. The mkkrb5srv script invokes config.krb5 to perform the server configuration. It is enough to specify the Kerberos realm and the domain:

```
# mkkrb5srv -d weeorg.com -r WEEORG.COM -s r-aix.weeorg.com
  Fileset                      Level  State      Description
  ----------------------------------------------------------------------------
Path: /usr/lib/objrepos
  krb5.server.rte            1.3.0.0  COMMITTED  Network Authentication Service
                                                 Server

Path: /etc/objrepos
  krb5.server.rte            1.3.0.0  COMMITTED  Network Authentication Service
                                                 Server
The -s option is not supported.
The administration server will be the local host.
Initializing configuration...
Creating /etc/krb5/krb5.conf...
Creating /var/krb5/krb5kdc/kdc.conf...
Creating database files...
Initializing database '/var/krb5/krb5kdc/principal' for realm 'WEEORG.COM'
master key name 'K/M@WEEORG.COM'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter database Master Password:
Re-enter database Master Password to verify:
WARNING: no policy specified for admin/admin@WEEORG.COM;
  defaulting to no policy. Note that policy may be overridden by
  ACL restrictions.
Enter password for principal "admin/admin@WEEORG.COM":
Re-enter password for principal "admin/admin@WEEORG.COM":
Principal "admin/admin@WEEORG.COM" created.
Creating keytable...
Creating /var/krb5/krb5kdc/kadm5.acl...
Starting krb5kdc...
krb5kdc was started successfully.
Starting kadmind...
kadmind was started successfully.
The command completed successfully.
Restarting kadmind and krb5kdc
```

As you can see, the KDC and administration servers are also started. The next step is to create the service principals and keys. We define the remote access (host) and file transfer (ftp) service principals for our two hosts. The service keys must also be added to key tables on both hosts. Of course, only keys for service

principals that are running on a particular host should be stored there. For this purpose run the script that we show in Example 3-8 on every host.

Example 3-8 Generating service principals and the host key table

```
R=WEEORG.COM
CACHE=my.cache
ADM=admin/admin
HOST=`hostname`
SERVICES="host ftp"
kinit -c $CACHE -S kadmin/admin $ADM
for s in $SERVICES; do
        s=$s/$HOST
        if kadmin -c $CACHE -p $ADM -q "getprinc $s@$R" 2>/dev/null |
                grep -qws $s
        then
                echo "WARNING: principal $s@$R already exists"
        else
                kadmin -c $CACHE -p $ADM -q "addprinc -randkey $s/$R"
        fi
        kadmin -c $CACHE -p $ADM -q "ktadd $s/$R"
done
```

We need to generate the krb5.conf file on the slave server. We just copy the file from the master server and edit it to remove unnecessary tags. Alternatively, you may take Example 3-1 on page 55 as a template.

Before starting the slave server the database must exist there. We now set up the Kerberos propagation to transfer the database from the master server. Slave KDC servers run the kpropd daemon to receive the database from the master server. The kpropd server has a simple ACL file to control access. This file contains a list of principals that will be sending the database. This is typically just the master server: host/r-aix.weeorg.com@WEEORG.COM. The file is named kpropd.acl and its location depends on the local setup. On Linux Red Hat, it is /var/kerberos/krb5kdc/kpropd.acl. One usually runs the kpropd daemon using the inetd Internet superserver. There is not much difference if we run it as a stand-alone daemon, so we just use the startup script provided with the Red Hat Linux distribution:

```
# service kprop start
Starting Kerberos 5 Propagation Server:                    [  OK  ]
```

We now create a simple custom script, which will run on the master server, to propagate the database now and then (see Example 3-9 on page 77). This script should be run as a cron job at least once an hour. We also use it to propagate the database manually. Note that if it produces no output, the database has been successfully transferred to the slave servers.

Example 3-9 Kerberos database replication script

```
#!/bin/sh
#
PATH=/usr/krb5/sbin:/usr/krb5/bin:$PATH
SLAVES="r-linux.weeorg.com"
DUMPF=/var/krb5/krb5kdc/slave_transfer
#
kdb5_util dump $DUMPF
for h in $SLAVES; do
        kprop -f $DUMPF $h | grep -v SUCCEEDED
done
```

We may now start the slave Kerberos server:

```
# service krb5kdc start
Starting Kerberos 5 KDC:                                   [  OK  ]
```

3.7.2 Standard Kerberos services

It is not very useful to run the Kerberos authentication system without any user services. In a Kerberos environment, every application has to understand the underlying protocol. Thus, we must start Kerberos enabled application servers. The exact procedure, of course, depends on the platform.

The standard AIX 5L Version 5.2 telnet, rlogin, and ftp servers include support for Kerberos. It is only necessary to include the Kerberos authentication method:

```
# chauthent -k5 -std
```

Other UNIX platforms use the inetd Internet superserver to start these servers. Red Hat Linux Version 8.0 is furnished with a similar superserver xinetd. The xinetd daemon keeps its configuration in files stored in the /etc/xinetd.d directory. Every service has its own file in a stanza format. For example, this is how the telnet service may be defined:

```
# default: off
# description: The Kerberized telnet server accepts normal telnet sessions, \
#              but can also use Kerberos 5 authentication.
service telnet
{
        flags           = REUSE
        socket_type     = stream
        wait            = no
        user            = root
        server          = /usr/kerberos/sbin/telnetd
        log_on_failure  += USERID
        disable         = no
}
```

You should find other Kerberos servers in klogin, kshell, and gssftp files. If necessary, change the disable option in each of them to “no” and then issue the following command:

```
# service xinetd reload
Reloading configuration:                                   [  OK  ]
```

The main reason for running Kerberos is security. Hosts allowing only Kerberos authentication are called *secure*. Hosts that run non-Kerberos service applications are called *insecure*. Naturally, it is preferable to run only secure hosts. However, some services do not support Kerberos, either because it is difficult to Kerberize them or because there was simply no interest to include the Kerberos support. If such a service transfers user credentials in the clear over the network, then you should consider replacing it with another service that offers a similar functionality. If that is not an option, then you may try tunneling its data through an SSL channel using one of the popular utilities. There is not much point in running Kerberos in an environment where other applications may send passwords in the clear.

It is possible to install other Kerberized applications, such as ssh, imap, and pop. Many applications support the Generic Security Service API (GSSAPI), which may be used to authenticate users with the Kerberos authentication service.

The Washington University imap server supports the GSSAPI authentication. On the client side, fetchmail supports almost all authentication mechanisms. The WU imap and fetchmail RPM packages on Red Hat Linux Version 8.0 are compiled with the GSSAPI support. All one has to do is to create an imap ‘service principal and to add it to the hosts key table file. The authentication method for fetchmail should be set to gssapi.

3.7.3 Kerberos authentication clients

The authentication clients accept requests from the non-Kerberos applications and authenticate users through the Kerberos authentication mechanism. Note that this is different from using Kerberos applications. In this subsection, we discuss the AIX Kerberos authentication module and the Linux PAM Kerberos module. This kind of authentication is typically used only once at the beginning of the session, for example, when the user logs in to his graphical workstation. The Kerberos authentication clients also obtain the TGT for the user.

AIX Kerberos authentication module

The AIX Kerberos authentication module is /usr/lib/security/KRB5. To use it, the system should be configured using the mkkrb5clnt program. In case the master server was not configured using the mkkrb5srv script, you should add the following line to the /var/krb5/krb5kdc/kadm5.acl file:

```
host/*@WEEORG.COM         i
```

To support our WEEORG.COM realm, we can issue the following command:

```
# mkkrb5clnt -r WEEORG.COM -l r-aix.weeorg.com -d weeorg.com -i LDAP -K
```

This command says that the host belongs to the WEEORG.COM realm, the information about KDC and administration server may be found at the r-aix.weeorg.com LDAP server, our domain is weeorg.com, and the users' information is administered in LDAP.

If you want to administer users using standard AIX user management commands such as **mkuser**, then this is not enough. In that case, you have to specify the -A option:

```
# mkkrb5clnt -r WEEORG.COM -l r-aix.weeorg.com -d weeorg.com -A -i LDAP -K
Password for admin/admin@WEEORG.COM:
Configuring fully integrated login
Authenticating as principal admin/admin with existing credentials.
WARNING: no policy specified for host/r-aix.weeorg.com@WEEORG.COM;
  defaulting to no policy. Note that policy may be overridden by
  ACL restrictions.
Principal "host/r-aix.weeorg.com@WEEORG.COM" created.

Administration credentials NOT DESTROYED.
Authenticating as principal admin/admin with existing credentials.

Administration credentials NOT DESTROYED.
Configuring Kerberos as the default authentication scheme
Making root a Kerberos administrator
Authenticating as principal admin/admin with existing credentials.
WARNING: no policy specified for root/r-aix.weeorg.com@WEEORG.COM;
  defaulting to no policy. Note that policy may be overridden by
  ACL restrictions.
Enter password for principal "root/r-aix.weeorg.com@WEEORG.COM":
Re-enter password for principal "root/r-aix.weeorg.com@WEEORG.COM":
Principal "root/r-aix.weeorg.com@WEEORG.COM" created.

Administration credentials NOT DESTROYED.
Cleaning administrator credentials and exiting.
```

The -A option will instruct the script to create a new administration principal root/*hostname*, such as root/marguerite.weeorg.com. This principal should have administrative rights, so you should have the following line in kadm5.acl:

```
root/*@WEEORG.COM         *
```

The mkkrb5clnt script configures a new method in /usr/lib/security/methods.cfg:

```
KRB5LDAP:
        options = db=LDAP,auth=KRB5
```

All users authenticating through Kerberos should have KRB5LDAP specified for the SYSTEM and registry attributes in /etc/security/user. The mkseckrb5 script takes care of that task for the existing users:

```
# mkseckrb5 joe
Please enter the admin principal name: admin/admin
Enter password:
Importing joe
Enter password for principal "joe@WEEORG.COM":
Re-enter password for principal "joe@WEEORG.COM":
```

For more details about the AIX and Kerberos integration, see the *AIX 5L Differences Guide Version 5.2 Edition*, SG24-5765.

PAM Kerberos module

Like many other Linux distributions, Red Hat Linux Version 8.0 delivers the PAM Kerberos module as part of the Kerberos support. This module is capable of authenticating users, obtaining TGT, and changing passwords. The Red Hat authconfig program may be used to configure Kerberos authentication, which includes setup for both Kerberos clients and the PAM Kerberos module:

```
# authconfig --kickstart --enableshadow --enablemd5 \
    --enableldap --ldapserver r-aix.weeorg.com --ldapbasedn dc=weeorg,dc=com \
    --enablekrb5 --krb5realm WEEORG.COM \
    --krb5kdc kerberos.weeorg.com,kerberos-1.weeorg.com \
    --krb5adminserver kerberos.weeorg.com
```

One can also stack the LDAP authentication method along with the Kerberos authentication, but we do not recommend mixing authentication methods for security reasons. Of course, the standard authentication method must be enabled for the sake of system users such as root.

3.8 Migrating users to Kerberos

In Chapter 2, “Centralized user management” on page 15, we describe how to move users from local map files to the LDAP directory. The effort needed to migrate a few tens of users is close to that for a few thousands of users. That is not the case with Kerberos.

Kerberos passwords are stored in a decryptable format. One simply cannot convert crypt or md5 hashes into the Kerberos format. In order to generate a user principal, the administrator has to have the corresponding password in clear text. On the other hand, the password hashes on UNIX cannot be decrypted. If the number of your users is small, you may invite all of them to stop by and negotiate new passwords. If there are many users, then this is probably not an option.

One possibility may be to harvest user passwords during the migration phase. To do this, you probably have to modify an existing authentication module or write another for this purpose. We are not aware of any such module at the moment, though rumor has it that there is a PAM module available for this purpose. It may also be possible to use strategically located network sniffers, but your security policy may prohibit that. The security policy may prohibit collecting user passwords as well, so you should discuss the matter with management. For example, if the new authentication module is named XPF, which simply appends the user name and the password to a file, and the authentication currently in use is LDAP, we can set up the following SYSTEM grammar:

```
SYSTEM = LDAP AND XPF
```

It is possible to run the kadmin program in batch mode. If there is a file with one user name and a password per line separated by white space, the following shell script may be used to create principals:

```
R=WEEORG.COM
CACHE=my.cache
ADM=admin/admin
kinit -c $CACHE -S kadmin/admin $ADM
while read user pass; do
        if kadmin -c $CACHE -p $ADM -q "getprinc $user@$R" 2>/dev/null |
                grep -qws $user@$R
        then
                echo "WARNING: principal $user@$R already exists"
        else
                kadmin -c $CACHE -p $ADM -q "addprinc -pw $pass $user@$R”
        fi
done < userpwd.txt
```

This is a very sensitive business and the file containing user data must be scrutinized before running the script.

3.9 Security considerations

Kerberos is one of the most secure authentication systems. Still, as with the LDAP authentication model, when user information is stored in a single place, migration to the SSO authentication introduces the same set of weaknesses. The Kerberos master database and application servers' key tables are sensitive information, which should be readable only by owners.

All KDC servers host the Kerberos data. These computers should have limited access and be physically secure. The key table files must be present on all hosts that offer Kerberized services. These files must be protected from unauthorized access and should be excluded from regular backups. Should the need arise, the service keys are easy to generate from scratch using a script, such as the one we show in Example 3-8 on page 76.

3.10 Enterprise Identity Mapping (EIM)

In a enterprise, it is common to have more than one user registry, which requires each person or entity within the enterprise to have a user identity in each registry. Today, the systems are becoming more complex than ever and you need to manage these multiples user registries. The need for multiple user registries quickly grows into a problem that affects users, system administrators, and application developers. Enterprise Identity Mapping (EIM) enables inexpensive solutions for more easily managing and working with multiple user identities in you enterprise.

3.10.1 EIM concepts

EIM is a mechanism for mapping a person or entity to the appropriate user identities in various registries throughout the enterprise. EIM provides APIs for creating and managing these identity mapping relationships, as well as APIs used by applications to query this information.

In a enterprise environment, it is common that a user has several users and passwords to access different systems and resources. Usually, the user sets the passwords to be the same one. However, this action compromises the security of the network. Using EIM and single sign-on enablement, you are able to manage this problem, because the users do not need to authenticate against every system that they want to access. Using EIM, this authentication occurs just once when the user is authenticated in the network. It is good for the users, because they do not need to remember many user names and passwords, and it is good for administrators, because they will not have so many problems related to users that forgot their password in a given system. It is good to the developers also,

because now they are free to write applications that use an appropriate existing user registry for authentication and a different user registry for authorization.

3.10.2 Using Enterprise Identity Mapping

EIM provides APIs for creating and managing the identity mapping relationships, as well as APIs used by applications to query this information.

The APIs provided by EIM allow applications to ask questions about the relationship between individuals and entities in an EIM architecture. So EIM architecture describes the relationship between individuals or entities in the enterprise and the many identities that represent them within an enterprise

Using the EIM APIs, you can determine a user identity in a given system if you know the identity of the user in any other system. For example, you have user A in one user registry. Now, you want know what user corresponds to user A in another user registry, so all you have to do is ask EIM what is the identity in a user registry that has the user name A in another giver user registry.

Identity mapping requires that administrators do the following:

1. Create EIM identifiers that represent people or entities in their enterprise.
2. Create EIM registry definitions that describe the existing user registries in their enterprise.
3. Define the relationship between the user identities in those registries to the EIM identifiers that they created.

No code changes are required to existing registries. Mappings are not required for all identities in a user registry. EIM allows one-to-many mappings (in other words, a single user with more than one identity in a single user registry). EIM also allows many-to-one mappings (in other words, a single user with more than one identity in a single user registry). EIM also allows many-to-one mappings (in other words, multiple users sharing a single identity in a single user registry, which, although supported, is not advised for security reasons). An administrator can represent any user registry of any type in EIM.

For more information about EIM, visit the following Web site:

http://publib.boulder.ibm.com/eserver/index.html

The EIM APIs are available on all IBM platforms. At the time of the writing of this redbook, the EIM development kit on Linux was not available yet. However, IBM will release it soon.

On AIX 5L Version 5.2, the EIM APIs are in the library libeim.a, which is part of the bos.eim fileset. The default location of libeim.a is /usr/ccs/lib. The bos.eim fileset is located on the second disk of the AIX 5L Version 5.2 installation disks.

4

Networking services

Different types of systems can be connected on a network. But in order for these systems to communicate with each other, they must speak the same language. The various languages computers use in communication are commonly called protocols.

In this chapter, the following topics are discussed:

- Protocols
- Data transfers
- Network management
- Network optimization

4.1 Protocols

Protocols define how information is delivered, how it is enclosed to reach its destination safely, and what path it follows. Protocols also coordinate the flow of messages and their acknowledgments.

Protocols exist at different levels within the kernel. They can be manipulated through the Application Programming Interface (API) level commands, for example, **no** and **nfso**, on the AIX operating system. The choices a user makes when invoking file transfer, remote login, or terminal emulation programs define the protocols used in the execution of those programs.

4.1.1 Domain Name System (DNS)

Although 32-bit Internet addresses provide machines with an efficient means of identifying the source and destination of data sent across a network, users prefer meaningful, easily remembered names. Transmission Control Protocol/Internet Protocol (TCP/IP) implementations of a BSD origin provide for a name to address resolution and vice versa through the well known resolver library interface. The flat organizations are supported with the /etc/hosts system map, whereas the DNS serves the hierarchical namespace.

The Domain Name System (DNS) uses delegated authority in a hierarchical context used by TCP/IP applications to map between host names and IP addresses. Each site maintains its own database of information and runs a server program that other systems across the network can query. The domain name hierarchy provides the assignment of symbolic names to networks and hosts that are meaningful and easier to remember. The DNS provides the protocol that allows clients and servers to communicate with each other. So, instead of each machine on the network keeping a file containing the name-to-address mapping for all other hosts on the network, one or more hosts are selected to function as name servers. These name servers translate (resolve) symbolic names assigned to networks and hosts into the efficient Internet addresses used by machines.

Configuring DNS setup

Before you configure a name server, decide which type or types best fit the network it will serve. Here are some of the types of name servers:

- Master name server

  A server that stores the database containing name-to-address mapping information. It loads its data from a file and delegates authority to other servers in its domain.

- Slave name server or stub name server

  A slave name server is a delegated authority. It receives its domain data files directly from a primary master server at system startup time for a given zone of authority and then periodically asks the master server to update its information.

- Hint name server

  A server that is not authoritative for any domain. It responds to requests to resolve names by querying other servers that have the authority to provide the information needed. All servers are caching servers. This means that the server caches the information that it receives for use until the data expires.

Note: Previous generations of the named name server specified the master name server as the primary name server, the slave name server as the secondary name server, and the hint name server as the caching-only name server. Any reference to the named.conf file in this documentation is specific to AIX 4.3.2 and later versions.

Keep in mind that a name server can function in different capacities for different zones of authority. For example, one name server host can be a master name server for one zone and a slave name server for another zone.

Implementation example

Now lets define our own DNS and domain setup. We're going to make the domain demo.com and define machines in it.

We have a 192.168.5/24 network that consists of the following setup:

- master.demo.com, an AIX system that serves as the master name server for the demo.com network 192.168.5.4 using Bind Version 9.
- slave.demo.com, a Linux system that serves as a slave name server for the demo.com network 192.168.5.5 using Bind Version 9.
- aix.demo.com, an AIX host machine, 192.168.5.10

Note: Not all characters are allowed in host names. Host names are restricted to the characters of the English alphabet (a-z), numbers (0-9), and the character "-" (dash). Upper and lower case characters are treated the same for DNS.

Configuring the master name server (AIX system)

To configure the master name server, use the following procedure:

1. Edit the /etc/named.conf file.

   This file is read each time the named daemon starts. It tells the server which type of server it is, the zone or zones for which it is responsible, and where to get its initial information.

   a. To use paths relative to a specified directory, we use the options configuration clause to specify the directory where the named data files are located (in this case, /usr/local/domain) and other attributes that we needed for this setup.

   ```
   options {
      directory "/usr/local/domain";
      allow-query { 192.168.5/24; 127.0.0.1; };
      allow-transfer { 192.168.5.5; };
      transfer-format many-answers;
   };
   ```

   In the `allow-query` line, we added the IP addresses in the network to allow it to make DNS queries to the server.

   In the `allow-transfer` line, we added the slave name sever (192.168.5.5) to allow it to receive zone transfers from the server. You must ensure that only real slave name servers are allowed to transfer zones from the master name server. This is one of the security features of BIND.

   The `transfer-format` specifies the way outbound zone transfers are formatted. The `many-answers` value means fit as many RRs into each DNS message as possible.

   b. To allow record data to be cached outside of defined zones (optional), specify the name of the hint zone file:

   ```
   zone "." in {
      type hint;
      file "/etc/named.ca";
   };
   ```

   > **Note:** The type hint declaration does not refer to the Hint name server. When the server starts up, it uses the root hints to find the root nameserver and get the most recent list of root nameservers.

   c. To configure the server as the master for a zone, we specify each zone (both domain and reverse zones) and its domain data file.

   ```
   zone "demo.com" in {
      type master;
      file "demo.com";
   ```

```
};
zone "5.168.192.in-addr.arpa" in {
   type master;
   file "5.168.192.in-addr.arpa";
};
```

d. The PTR record for localhost is also required:

```
zone "0.0.127.in-addr.arpa" in {
   type master;
   file "/etc/named.local";
};
```

2. Edit the /etc/named.local file.

   a. Specify the start of authority (SOA) of the zone and the default time-to-live information:

```
$TTL 345600
@      IN     SOA    master.demo.com. root.master.demo.com. (
                     20021030  ;serial
                     3600      ;refresh
                     600       ;retry
                     3600000   ;expire
                     86400 )   ;minimum
```

   b. Specify the name server (NS) record:

```
       IN     NS     master.demo.com.
```

   c. Specify the pointer (PTR) record:

```
1      IN     PTR    localhost.
```

3. Edit the domain data file (/usr/local/domain/demo.com file).

   For /usr/local/domain/demo.com, use the following procedures:

   a. Specify the start of authority of the zone and the default time-to-live information for the zone. This record designates the start of a zone. Only one start of authority record per zone is allowed.

```
$TTL 345600
@              IN     SOA    master.demo.com. root.master.demo.com. (
                             20021030  ;serial
                             3600      ;refresh
                             600       ;retry
                             3600000   ;expire
                             86400)    ;minimum
```

   b. Include name server records for all master and slave name servers in the zone:

```
               IN     NS     master.demo.com.
               IN     NS     slave.demo.com.
```

c. Include name-to-address resolution information on all hosts in the name server zone of authority:

```
localhost    IN    A     127.0.0.1
master       IN    A     192.168.5.4
slave        IN    A     192.168.5.5
aix          IN    A     192.168.5.10
```

4. Edit the reverse zone file (/usr/local/domain 5.168.192.in-addr.arpa).

   a. Specify the start of authority of the zone and the default time-to-live information for the zone. This record designates the start of a zone.

```
@            IN    SOA   master.demo.com. root.master.demo.com. (
                         20021030  ;serial
                         3600      ;refresh
                         600       ;retry
                         3600000   ;expire
                         86400)    ;minimum
```

   b. Include name server records for all master and slave name servers in the zone:

```
             IN    NS    master.demo.com.
             IN    NS    slave.demo.com.
```

   c. Include address-to-name resolution information on all hosts in the name server's zone of authority:

```
4            IN    PTR   master.demo.com.
5            IN    PTR   slave.demo.com.
10           IN    PTR   aix.demo.com.
```

5. Create an /etc/resolv.conf file.

   The presence of this file indicates that the host should use a name server, not the /etc/hosts file, for name resolution. This file must exist on a name server host and can contain the local host address, the loopback address (127.0.0.1), or be empty. In this case:

```
nameserver 127.0.0.1
domain demo.com
```

   The 127.0.0.1 address is the loopback address, which causes the host to access itself as the name server.

6. Run the named daemon.

   Initialize the named daemon by using **smit stnamed** or through the command line with:

```
# startsrc -s named
```

Then edit the /etc/rc.tcpip file and uncomment the line for the named daemon by removing the comment (#) symbol from the following line:

```
start /etc/named "$src_running"
```

This initializes and starts the daemon with each system startup.

Configuring the slave name server (Linux system)

To configure the slave name server, use the following procedures:

1. Edit the /etc/named.conf file.

   This file is read each time the named daemon starts. It tells the server which type of server it is, the zone or zones for which it is responsible, and where to get its initial information.

   a. To use paths relative to a specified directory, we use the options configuration clause to specify the directory (in this case, /usr/local/domain) in which the named data files can be located:

   ```
   options {
      directory "/usr/local/domain";
      allow-query { 192.168.5/24; 127.0.0.1; };
      allow-transfer { 192.168.5.4; };
      transfer-format many-answers;
   };
   ```

   b. To allow record data to be cached outside of defined zones (optional), specify the name of the hint zone file:

   ```
   zone "." in {
      type hint;
      file "/etc/named.ca";
   };
   ```

   c. To configure the server as the master for a zone, we specify each zone (both domain and reverse zones) and its data file:

   ```
   zone "demo.com" in {
      type slave;
      file "demo.com.bak";
      masters { 192.168.5.4; };
   };
   zone "5.168.192.in-addr.arpa" in {
      type slave;
      file "5.168.192.in-addr.arpa.bak";
      masters { 192.168.5.4; };
   };
   ```

d. To support resolving the loopback network address, specify a zone of type master with a source of /etc/named.local, as well as the domain for which the name server is responsible:

```
zone "0.0.127.in-addr.arpa" in {
   type master;
   file "/etc/named.local";
```

2. Edit the /etc/named.local file.

   a. Specify the start of authority (SOA) of the zone and the default time-to-live information:

```
$TTL 345600
@      IN     SOA    slave.demo.com. root.slave.demo.com. (
                     20021030  ;serial
                     3600      ;refresh
                     600       ;retry
                     3600000   ;expire
                     86400)    ;minimum
```

   b. Specify the name server (NS) record:

```
       IN     NS     slave.demo.com.
```

   c. Specify the pointer (PTR) record:

```
1      IN     PTR    localhost.
```

3. Create an /etc/resolv.conf file.

   The presence of this file indicates that the host should use a name server, not the /etc/hosts file, for name resolution. This file must exist on a name server host and can contain the local host address, the loopback address (127.0.0.1), or be empty. In this case:

```
nameserver 192.168.5.5
nameserver 127.0.0.1
domain demo.com
```

4. Run the named daemon.

   Initialize the named daemon through the command line with:

```
# /etc/rc.d/init.d/named restart
```

5. Check if the master server was able to do zone transfers to the slave server.

   List the files in the domain data directory. You should get an output similar to Example 4-1 on page 93.

Example 4-1 Domain data files

```
[root@slave root]# ls -l /usr/local/domain
total 8
-rw-------   1 named      sys       437 Nov   1 15:05 5.168.192.in-addr.arpa.bak
-rw-------   1 named      sys       450 Nov   1 15:05 demo.com.bak
```

Configuring an AIX host to use the name server

To configure the host (aix) to use the nameserver, use the following procedures:

1. Create an /etc/resolv.conf file.

   If this host uses more than one name server, add the names of the other name servers. In this example, the file will contain the following lines:

   ```
   nameserver 192.168.5.4
   nameserver 192.168.5.5
   domain demo.com
   ```

2. Test the communication between the host and the name server by typing the following command:

   ```
   # host aix
   ```

   The output you receive should appear similar to the following:

   ```
   aix.demo.com is 192.168.5.10
   ```

For security enhancements, such as setting up DNS security (DNSSEC) and Transaction Signature (TSIG) support, in BIND 9, refer to the *AIX 5L Differences Guide Version 5.2 Edition*, SG24-5765.

Testing your DNS setup

To test the DNS setup and to see if the master and slave can resolve the domain names properly, use the **nslookup** command. On the AIX client, aix, do the following steps:

1. Check if the master DNS server is resolving the host names correctly. Refer to Example 4-2.

Example 4-2 nslookup example for the master DNS server

```
[root@aix]: /> nslookup
Default Server: master.demo.com
Address:  192.168.5.4

> host aix
Server:   master.demo.com
Address:  192.168.5.4

Authorative answer:
Name:  aix.demo.com
```

```
Address:  192.168.5.10
```

2. Now, check the slave server. You should also get an authoritative answer if you were able to do zone transfers successfully. Refer to Example 4-3.

Example 4-3 nslookup example for the slave DNS server

```
[root@aix]: />  nslookup
Default Server: master.demo.com
Address:  192.168.5.4

>  server slave
Default Server: slave.demo.com
Address:  192.168.5.5

>  host aix
Server:   slave.demo.com
Address:  192.168.5.5

Authorative answer:
Name:  aix.demo.com
Address:  192.168.5.10
```

4.1.2 Dynamic Host Configuration Protocol (DHCP)

This is an application-layer protocol that allows network clients a way to obtain proper network configuration without knowledge or understanding of the network. Quite simply, a client on the network can easily obtain an IP address and other configuration parameters without knowing anything about the network *per se*. Typically, the DHCP server answers to broadcast packets sent by the client.

DHCP is a mechanism for the automatic network configuration of hosts through IP while IP addresses become “real estate” controlled by the server and are issued and leased to each client. It gives the network administrator a method to remove the end user from configuration problem and maintain the network configuration in a centralized location.

Configuring the DHCP server

There are numerous configurations and network topologies to consider when configuring a DHCP sever. By default, the DHCP server is configured by reading the /etc/dhcpsd.cnf file, which contains entries for logging information, options to return, machines to configure, and other items and information needed by the DHCP clients.

Configuring the DHCP server is usually the hardest part of using DHCP in your network. First, decide what networks you want to have DHCP clients on. Each

subnet in your network represents a pool of addresses that the DHCP server must add to its database. In Linux, the DHCP server is usually in the ISC package. This server does not use the numeric option, but symbolic, to specify the client option.

Implementation example

Lets define our own DCHP server and client setup. We are going to use the same domain demo.com from the DNS example.

We have a 192.168.5/24 network that consists of the following setup:

- Host name bayani, an AIX system that serves as the DHCP server and gateway for the demo.com network, 192.168.5.1.
- Host name clientA, a Linux client machine.
- Host name clientB, an AIX client machine.

Configuring the AIX DHCP server

To configure bayani as our DHCP server, set the /etc/dhcpsd.cnf file similar to the one in Example 4-4.

Example 4-4 /etc/dhcpsd.cnf

```
numLogFiles  5
logFilesSize 100
logFileName  /var/tmp/dhcpsd.log
logItem   SYSERR
logItem   OBJERR
logItem   PROTERR
logItem   ACNTING
logItem   ACTION
logItem   EVENT

leaseTimeDefault      30 minutes
leaseExpireInterval   3 minutes
supportBOOTP             No
supportUnlistedClients   Yes

network 192.168.5.0      24
{
   subnet 192.168.5.0    192.168.5.15-192.168.5.250
   {
      option    1  255.255.255.0
      option    3  192.168.5.1
      option    6  192.168.5.4 192.168.5.5
      option    15 demo.com
   }
```

```
}
```

Where:

option 1	Subnet Mask
option 3	Router option
option 6	DNS servers
option 15	Domain name

Then start the DHCP server daemon by using the command:

```
# startsrc -s dhcpsd
```

Alternatively, you can also use SMIT to start the DHCP server daemon. To do this, just execute the **`smit dhcpsd`** command and you should get a screen similar to Example 4-5.

Example 4-5 smit dhcpsd

```
                              dhcpsd Subsystem

Move cursor to desired item and press Enter.

  Start Using the dhcpsd Subsystem
  Stop Using the dhcpsd Subsystem



F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Configuring a Linux host to use the DHCP server

To configure the host (clientA) to use DHCP, use the following procedures:

1. Edit your Ethernet interface configuration file (in this case, we are using the eth0 interface). It should contain the following lines:

   ```
   DEVICE=eth0
   BOOTPROTO=dhcp
   ONBOOT=yes
   ```

2. Restart the network or the network interface with the following commands:

   ```
   # /etc/rc.d/init.d/network restart
   ```

   or

   ```
   # /etc/sysconfig/network-scripts/ifup eth0
   ```

3. Check if the client was able to receive an IP address and other information, passed by the DHCP server, using the command:

   ```
   # ifconfig eth0; netstat -rn
   ```

4. You should get an output similar to Example 4-6.

Example 4-6 ifconfig and netstat output in Linux client

```
[root@clientA root]# ifconfig eth0; netstat -rn
eth0      Link encap:Ethernet    HWaddr 00:04:AC:EE:37:DC
          inet addr:192.168.5.11    Bcasr:192.168.5.255    Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:7 errors:0 dropped:0 overruns:0 frame:0
          TX packets:13 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:1548 (1.5 Kb)    TX bytes:1746 (1.7 Kb)
          Interrupt:11 Base address:0xf000

Kernel IP routing table
Destination      Gateway         Genmask         Flags  MSS Window   irtt Iface
192.168.5.0      0.0.0.0         255.255.255.0   U       40 0           0 eth0
127.0.0.0        0.0.0.0         255.0.0.0       U       40 0           0 lo
0.0.0.0          192.168.5.1     0.0.0.0         UG      40 0           0 eth0
```

Configuring an AIX host to use the DHCP server

To configure the host (clientA) to use DHCP, use the following procedures:

1. Run **`smit tcpip`** and choose **Use DHCP for TCPIP Configuration & Startup**. Then select the network interface that the system is using and press Enter to run the process. Execution of this screen will start the dhcpcd client daemon process. Alternatively, you can use **`startsrc -s dhcpcd`** command to start the DCHP client daemon in the command line prompt.

2. Check if the client was able to receive an IP address and other information, passed by the DHCP server, using the command:

   ```
   # ifconfig en0; netstat -rn
   ```

3. You should get an output similar to Example 4-7.

Example 4-7 ifconfig and netstat output in AIX client

```
[root@clientB]: /> ifconfig en0; netstat -rn
en0:
flags=4e080863<UP,BROADCAST,NOTRAILERS,RUNNING,SIMPLEX,MULTICAST,GROUPRT,64BIT,
PSEG>
        inet 192.168.5.11 netmask 0xffffff00 broadcast 192.168.5.255
Routing tables
Destination      Gateway           Flags   Refs     Use  If   PMTU Exp Groups
```

```
Route Tree for Protocol Family 2 (Internet):
default            192.168.5.1        U         0        0  en0       -   -
192.168.5/24       192.168.5.12       U         4   209096  en0       -   -
192.168.5.12       127.0.0.1          UGHS     54   181254  lo0       -   -

Route Tree for Protocol Family 24 (Internet v6):
::1              ::1               UH        0        0  lo0 16896   -
```

4.1.3 Network Time Protocol (NTP)

Network Time Protocol is an Internet Protocol used to synchronize computer clocks. The reference time may be a single source, but it is usually multiple sources.

Having your clocks synchronized in all your systems is as important as having a solid network strategy, especially when administering distributed applications, Web services, or a distributed security monitoring tool. In these cases, it is a must to have all clocks synchronized.

NTP is a fault-tolerant protocol that will automatically select the best of several available time sources to synchronize to. In other words, an NTP server is able to tell good time sources from the bad ones given that it has been provided with enough servers. Then, it picks one of the servers deemed eligible for the synchronization purpose. Therefore, and also for backup, it is important that every NTP server is provided with at least three lower-stratum NTP servers. For increased availability, these servers should be on different network paths and administered by different entities.

Configuring the NTP setup

NTP needs a reference clock that defines the time to operate by and all clocks are set based on that time.

NTP use UDP. NTP servers such as xntpd relies on multiple time reference providers. NTP automatically selects the best of several available time sources to synchronize to. Multiple candidates can be combined to minimize the accumulated error. The NTP selection algorithm of one reference source eliminates all other references, not only those who are too far from the calculated mean value of them all.

Implementation examples

To configure our NTP setup, we need to define a reference clock that defines the *reference time*, and configure the clients clock based from that *reference time*.

We have a 192.168.5/24 network that consists of the following setup:

- Host name bayani, an AIX system that serves as the NTP server, 192.168.5.1.
- Host name clientA, a Linux client machine.
- Host name clientB, an AIX client machine.

Configuring the AIX NTP server

To configure bayani as our NTP server, do the following steps:

1. Set the /etc/ntp.conf file like the one in Example 4-8.

Example 4-8 ntp.conf file of NTP server

```
server 127.127.1.0
driftfile /etc/ntp.drift
tracefile /etc/ntp.trace
restrict default nomodify notrust
restrict 192.168.5.1
restrict 127.0.0.1
```

The 127.127.1.0 address is a special address that tells the AIX server to synchronize to its own system clock. Of course, a computer with a decent clock that does not “wander” much should be chosen as the private primary NTP server.

Note: For the purpose of simplicity, we are using the system clock or system time as the reference clock. In a real-world scenario, this setup is not recommended. To get a more precise and well-calibrated setup, use a cesium clock or refer to a public stratum 1 NTP server on the Internet.

The `restrict default nomodify notrust` line will only permit queries that return information and specifies treating hosts normally but never use them as synchronization sources.

The succeeding “restrict“ lines will prevent the server from attempting to synchronize to its own time at startup.

2. After editing the ntp.conf file, start xntpd using the command:

   ```
   # startsrc -s xntpd
   ```

   You can also start the xntpd daemon at each system restart by configuring it with **`smit xntpd`**.

3. Start querying the ntp server. Run:

   ```
   # lssrc -ls xntpd
   ```

If you get an output of `no peer, system is insane` in the Sys peer section, it means that xntpd has not synchronized yet with its system clock. The synchronization will take about 6-10 minutes. Check the status again after this time then the status should now be set to `127.127.1.0`.

Configuring the Linux NTP client

Do the following steps:

1. Start by querying the reference server first and make sure that you can synchronize to it by running the **`ntpdate`** command:

   ```
   # ntpdate -d 192.168.5.1
   ```

   In the output, you should get an offset in the last line. This tells you how far apart your system clock is with respect to the reference clock in the AIX server.

   > **Note:** If you get a `No server suitable for synchronization` message, it means that either the reference server is not accessible or configured as an NTP server or the reference has not synchronized yet to its system clock (127.127.1.0).

   If the offset is greater than 1000 seconds, you will not be able to synchronize to the NTP server. You can manually change this setting, so that the delta is less than 1000 seconds, or instantaneously synchronize with the NTP server. To do this, run the following command:

   ```
   # ntpdate 192.168.5.1
   ```

2. If the client was able to synchronize successfully with the NTP master, you can now configure the NTP client. The Linux NTP configuration will become a NTP server, not only a client. It will provide time services if requested by clients to do so.

   Edit the /etc/ntp.conf file to be similar to Example 4-9.

Example 4-9 ntp.conf file of NTP client

```
server 192.168.5.1
driftfile /etc/ntp.drift
tracefile /etc/ntp.trace
```

3. Then start the ntpd daemon:

   ```
   # /etc/rc.d/init.d/ntpd start
   ```

Configuring the AIX NTP client

Do the following:

1. Start by querying the reference server first and make sure that you can synchronize to it by running the **ntpdate** command:

   ```
   # ntpdate -d 192.168.5.1
   ```

   Again, if the offset is greater than 1000 seconds, instantaneously synchronize with the NTP server. To do this, run the following command:

   ```
   # ntpdate 192.168.5.1
   ```

2. If the client was able to synchronize successfully with the NTP master, you can now configure the NTP client. The AIX NTP configuration will become a NTP server, not only a client. It will provide time services if requested by clients to do so. By only using the `broadcastclient` statement, it will only listen to NTP broadcasts and set its own time.

   Edit the /etc/ntp.conf file to be similar to Example 4-10.

Example 4-10 ntp.conf file of NTP client

```
server 192.168.5.1
driftfile /etc/ntp.drift
tracefile /etc/ntp.trace
```

3. Then start xntpd using the command:

   ```
   # startsrc -s xntpd
   ```

   You can also start the xntpd daemon at each system restart by configuring it with **smit xntpd**.

4.1.4 Network Information Service (NIS)

The Network Information Service is a distributed service product designed by Sun Microsystems. Even though it is quite old, and some would say outdated, it is still in use in some environments. Unfortunately, NIS has a very poor security record and, depending on security requirements, it is probably not advisable to use it. The problem is not with implementations or coding errors, but with the design, which assumes a level of trust that is not easy to find in reality (at least not anymore). NIS+ is supposed to mend all the NIS deficiencies, which it does in many respects, but the price is a greater effort on the part of the system management.

NIS is provided for both AIX and Linux platforms. NIS+ is available on AIX, but Linux only has support for the API. Linux may join the NIS+ environment only as a NIS+ client. Not many sites deploy NIS+ on Linux, so the quality of the support may be a problem. It seems that the development of NIS+ for Linux has stopped.

On the other hand, NIS support on Linux has a longer history and should be better supported.

NIS distributes the UNIX /etc system maps, such as /etc/passwd, /etc/group, or /etc/services. The underlying communication technology is based on Remote Procedure Calls (RPC). Therefore, hosts in NIS environment must run portmapper to map RPC ports to TCP ports.

NIS maps are organized in a domain. Only one-level domains are supported. A NIS domain data resides on a host, which runs the NIS master server. For increased availability and to reduce the master server load, there should be one or more slave servers. Slave servers synchronize their copies of maps by pulling them from the master server. On the other hand, the master server may prompt the slave servers to pull the update maps. It may be done by the ypupdated daemon, which is available on AIX or manually by using the yppush program.

NIS servers keep databases in the DBM format. Therefore, all maps have to be converted from ASCII to this format using the makedbm program. The ypinit utility is typically used to initialize the database, either from the local files, in the case of the master server, or from the master server, if it is run on a slave server.

NIS servers are implemented in the ypserv program. It reads the maps and makes them available to the NIS clients. Every NIS client must bind to the server using the ypbind program. ypbind runs in the background and at startup tries to find a NIS server using a network broadcast. Broadcasting is convenient, but in this situation also a security issue, so it is better to give a hint to the ypbind program by specifying a server or a list of servers. On Linux, this is done by specifying NIS servers in /etc/yp.conf. On AIX, one should create a list of NIS servers in /var/yp/*nisdomain*/ypservers.

4.2 Data transfers

Although it is possible to send relatively short files using **`ftp`** and **`rcp`**, there are more efficient ways of transferring large files. **`ftp`** and **`rcp`** are usually designed to transmit relatively small amounts of text; therefore, other means are needed to transfer large files effectively and efficiently. This section will discuss some of the better tools that you can use for synchronizing and transferring files over the network. In this way, you can enable remote backup of data, as well as enabling redundancy features in case of a machine failure.

4.2.1 rsync

The rsync package provides a command for synchronizing files over a network. The **`rsync`** command uses a quick and reliable algorithm to quickly synchronize

remote and host files. The **rsync** command is fast because it just sends the differences in the files over the network (instead of sending the complete files). The **rsync** command is often used as a powerful mirroring process or just as a more capable replacement for the **rcp** command. To use **rsync** with SSL encryption, you will need to install the Secure Shell (ssh) package as a pre-requisite.

Implementation example

Here is an example setup. We have the following servers:

- An AIX server, which serves as our rsync server, 192.168.5.1
- A Linux server, which serves as our backup server, 192.168.5.5

The rsync package for AIX can be downloaded from AIX toolbox for Linux applications Web site:

http://www.ibm.com/servers/aix/products/aixos/linux/

You can download the latest rsync package for Linux at the following Web site:

http://samba.anu.edu.au/ftp/rsync/binaries

Configuring the AIX rsync server

When using rsync in a stand-alone daemon mode, it uses a single configuration file. By default, this file is named /etc/rsync.conf.

Here is a procedure for configuring the AIX rsync server:

1. Generate an /etc/rsync.conf file.

   This configuration file is similar to a Samba configuration file. Example 4-11 is how our rsync.conf file appears.

Example 4-11 rsync.conf file

```
motd file = /etc/motd
max connections = 5

[rsyncdir]
        comment = rsync test directory
        path = /home/rsyncdir
        read only = yes
        hosts allow = 192.168.5.5
```

> **Note:** The rsync daemon performs a chroot () system call to the path/s defined in the configuration file. For security reasons, all files within that path must be contained and should not contain symbolic links to other directories.

For more information on the rsync.conf file and its other options, read the man pages for rsync.

2. Launch the rsync daemon.

   To run the rsync daemon in stand-alone mode, use the command:

   ```
   # /usr/bin/rsync --daemon
   ```

   The rsync daemon will then fork and continue running in the background.

3. Do a local test.

   In the AIX machine, run the command:

   ```
   # /usr/bin/rsync localhost::
   ```

   You should get an output similar to Example 4-12.

Example 4-12 rsync local test

```
[root@aix]: /> /usr/bin/rsync localhost::
*******************************************************************************
*                                                                             *
*                                                                             *
*  Welcome to AIX Version 5.2!                                                *
*                                                                             *
*                                                                             *
*  Please see the README file in /usr/lpp/bos for information pertinent to    *
*  this release of the AIX Operating System.                                  *
*                                                                             *
*                                                                             *
*******************************************************************************

rsyncdir    rsync test directory
```

List the file contents in the rsyncdir directory, as in Example 4-13.

Example 4-13 Files in rsyncdir directory

```
[root@aix]: /> ls -l /home/rsyncdir
total 1
-rw-r--r--   1 guest    staff            15 Nov 07 15:48 test
```

Mirroring the rsync directory in Linux

To start mirroring the data defined in the rsync directory, you just need to run one command:

```
# rsync -a 192.168.5.1::rsyncdir/ /home/tmp
```

The motd is then shown, and the files should be transferred to /home/tmp. The archive mode attribute (**-a**) will ensure that all file information, such as, devices and attributes are preserved during the transfer.

List the file contents in the /home/tmp directory; it should contain the same content as the one in Example 4-13 on page 104 (refer toExample 4-14 for the output).

Example 4-14 Files in /home/tmp directory

```
[root@linux root]# ls -l /home/tmp
total 4
-rw-r--r--    1 100      bin            15 Nov  7 15:48 test
```

The mirroring will usually take less time than when using the normal **rcp** command. The rsync daemon uses the remote-update protocol, which allows rsync to transfer just the differences between two sets of files (if the destination file already exists) using a checksum-search algorithm. This is the reason why rsync is more efficient than rcp.

4.2.2 rdist

rdist is a program that will help maintain identical copies of files over multiple hosts on the network. It preserves the owner, group, mode, and mtime of files attributes, if possible, and at the same time updates programs that are executing. The new version of rdist does not need to be setuid to root. Instead of connecting to remote hosts directly, it now uses the remote shell (rsh) or secure shell (ssh) program to make its connections. The rdist package that we will use for this example was from the Toolbox, which you can download from AIX toolbox for Linux applications Web site:

http://www.ibm.com/servers/aix/products/aixos/linux/

Implementation example

Here is an example setup. We have the following servers:

- An AIX server, 192.168.5.1
- A Linux server, 192.168.5.5

We will copy files from the AIX server to the Linux server using the rdist program.

The rdist package for AIX can be downloaded from AIX toolbox for Linux applications Web site:

http://www.ibm.com/servers/aix/products/aixos/linux/

You can download the latest rdist package for Linux at the following Web site:

http://www.redhat.com/swr/i386/rdist-6.1.5-14.i386.html

Running the rdist command

In AIX, generate a distribution file (like a configuration file) and name it rdist.conf, as in Example 4-15.

Example 4-15 rdist.conf file

```
( /etc/motd) -> root@192.168.5.5
      install /home/backup/motd;
```

Then run the **rdist** command to start copying the file to the Linux server by executing:

```
# rdist -f /<path of the file>/rdist.conf
```

In the Linux server, list the file contents in the /home/backup/ directory and check if the file was transferred.

This procedure should work both on Linux and AIX. Also, make sure that rsh is working and configured properly to make rdist work.

4.3 Network management

In a working environment, where applications are running on a heterogeneous environment, the satisfaction of users is highly dependent on the response time. This is based not only on systems performance, but on how fast the information can be passed from one system to the other. This is why it is significant to periodically track network performance in your environment. Network management provides information about the current state of your network, and allows you to respond proactively to potential performance problems. In this chapter, you will get an overview of various protocols and tools that are available for tracking network performance and optimizing it.

A good book on network management is *UNIX Network Management Tools* by Maxwell.

4.3.1 SNMP

A very popular protocol to manage a network is called Simple Network Management Protocol, or SNMP. It is a simple and very useful protocol, because it is the common denominator for many managed systems. SNMP network management is based on the familiar client/server model that is widely used in TCP/IP-based network applications. It is an Internet standard for gathering statistics and managing devices on the Internet.

This protocol is like a language, with verbs and a vocabulary, and it is used for communication between a managed system called an *Agent* and a management station called a *Manager*. All the verbs use a special vocabulary, which is called the Management Information Base (MIB). This is the standard set of object definitions that all SNMP agents are required to support.

There are three requests that can be sent from the Manager:

get Retrieves the value of a particular variable and returns the answer to the requester (with response).

set Updates a variable on the managed system (and changes the system value that it represents); set is not available for all variables.

get-next Retrieves the next-variable. We do not know how many variables or instances we have for a specific agent. This powerful command allows you to scan the variables.

Two requests are used by the Agent:

response Answers the manager after receiving a get, get-next, or set.

trap Sends an alarm message to the manager, describing an extraordinary event.

It is possible to extend the MIB definition for an enterprise to define some specific variables concerning a special hardware, software, or a specific management strategy. To use an extended MIB, a subagent or a proxy-agent has to be set up.

In AIX 5L Version 5.2, SNMP Versions 2 and 3 is supported. To know more about SNMP and SNMP programming, consult the information provided in the *AIX 5L Version 5.2 Systems Management Guide: Communication and Networks* and *AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs* books.

4.3.2 IBM Tivoli® Netview

IBM Tivoli NetView® discovers TCP/IP networks, displays network topologies, correlates and manages events and SNMP traps, monitors network health, and gathers performance data. Tivoli NetView meets the needs of customers with large network infrastructure by providing the scalability and flexibility to manage mission-critical environments.

NetView has a functionality that provides a scalable management solution. It measures availability and provides fault isolation by quickly identifying the root cause of network failures.

The IBM Tivoli Netview supports AIX, Linux, Solaris, Windows NT, and Windows 2000.

A snapshot of Netview can be seen in Figure 4-1.

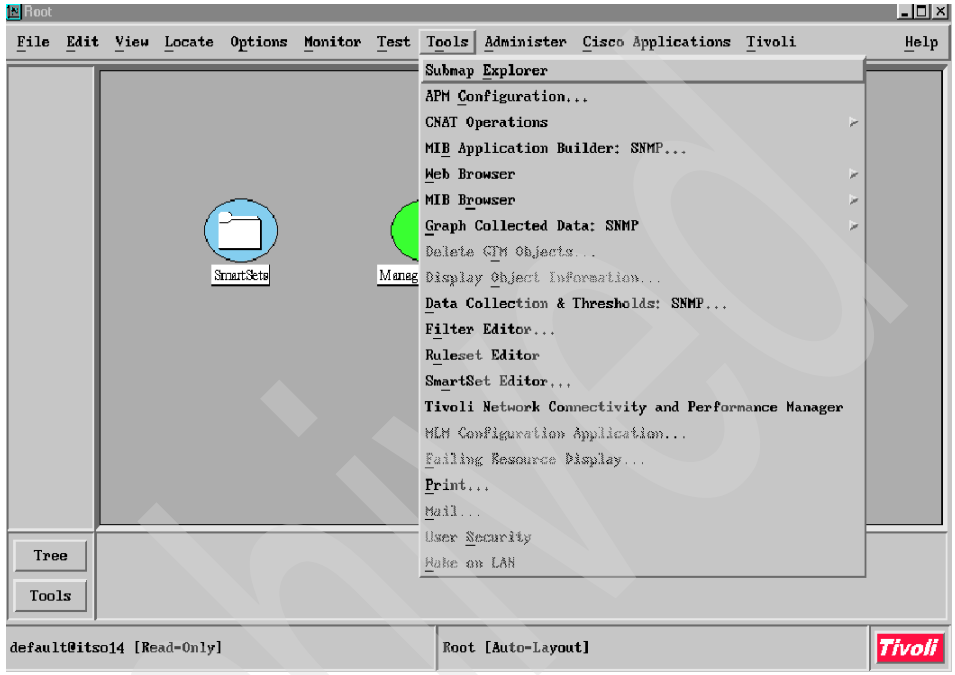

Figure 4-1 IBM Tivoli Netview

To get more information on IBM Tivoli Netview, go to the URL:

http://www.tivoli.com/products/documents/datasheets/netview.pdf

For Installation instructions and release notes on the latest version of Netview, point your browser to the URL:

http://www.ibm.com/software/sysmgmt/products/support/IBMTivoliNetView.html

Other Tivoli products related to network management

Here are some Tivoli products for network monitoring and management:

- IBM Tivoli Monitoring

  IBM Tivoli Monitoring provides monitoring for essential system resources, detecting bottlenecks and potential problems, and automatically recovering from critical situations.

- IBM Tivoli Netview Performance Monitor

  IBM Tivoli Netview Performance Monitor provides analyses of SNA-based traffic flows, request transit times, and component utilization, providing both critical real-time data and performance data for trend analysis.

- IBM Tivoli NetView for TCP/IP Performance

  IBM Tivoli NetView for TCP/IP Performance is a complementary product to IBM Tivoli NetView Performance Monitor, providing detailed information about TCP/IP performance from a Host perspective, as well as network connection information to selected critical Distributed Systems.

- IBM Tivoli Monitoring for Network Performance

  IBM Tivoli Monitoring for Network Performance packages two existing products, allowing you to properly manage the performance of hybrid network environments. The products packaged together are IBM Tivoli NetView for TCP/IP Performance and IBM Tivoli NetView Performance Monitor.

4.3.3 ntop

ntop is an open source software that works similar to the top tool, but measures and monitors network traffic. It supports various management activities, such as, network planning and optimization and network security. The newest version of ntop has included a quick access feature through its Web interface. ntop can both run in Linux and AIX by compiling the source or downloading the binary package from the Internet.

You can download various ntop formats from the following sources:

- ntop source package, found at:

  http://snapshot.ntop.org

- ntop rpm package, found at:

  http://www.rpmfind.net/linux/RPM

- ntop bff format, found at:

  http://www.bullfreeware.com

Before installing ntop, it is important to select the best host to run this program. The host should have an interface to the network to be monitored, since only the traffic captured through this interface can be analyzed. For a network environment that has several switches or bridge and routers, only the segment where the ntop host is connected will be monitored.

If you want to install and compile the source package rather than the binary formats, use the following steps:

```
# gunzip ntop-2.0-src.tgz | tar xvf -
# cd ntop
# ./autogen.sh -1
```

Running autogen.sh will recreate all the configure and make scripts from the .in and .am files, based on your systems specific configuration.

```
# ./configure
# make
# make install
```

After installation, you can run ntop through the command line and it will start capturing packets from the network. To activate the Web-based mode, just open a web browser and enter the following URL:

```
http://<ntop host>:3000
```

A snapshot of the ntop Web interface can be seen in the Figure 4-2 and Figure 4-3 on page 111.

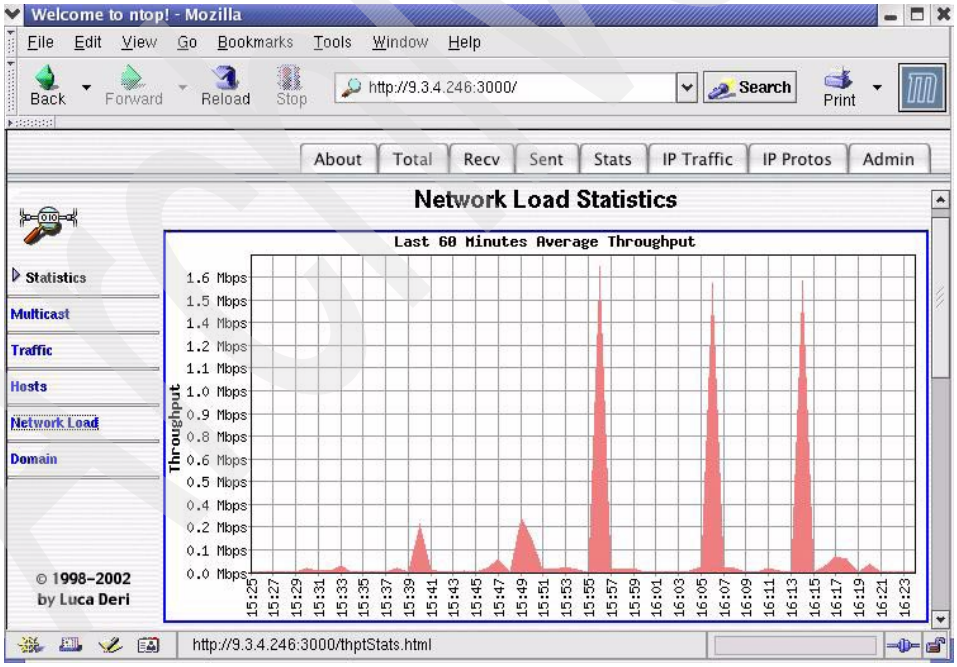

Figure 4-2 ntop Web interface: IP Protocol distribution

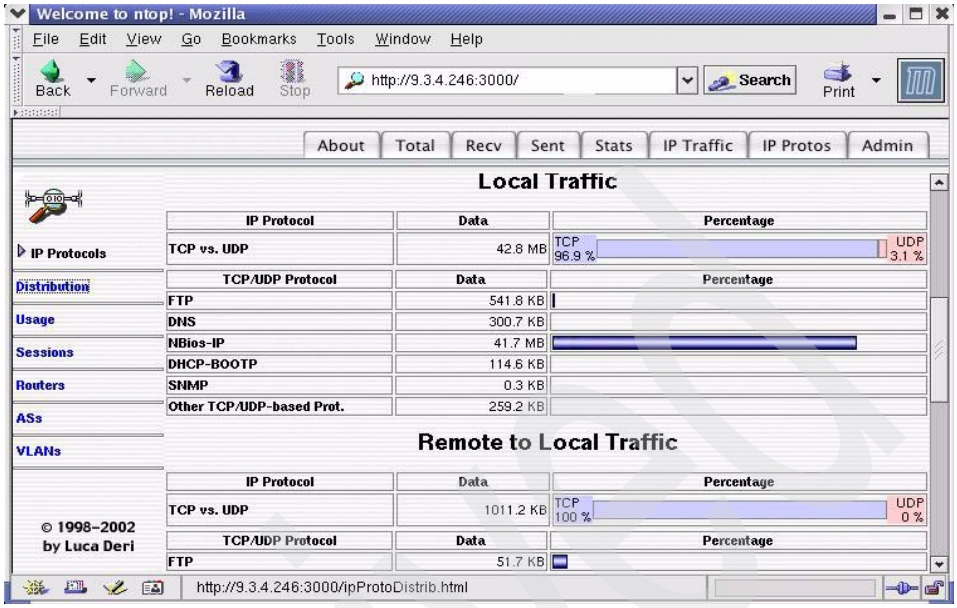

Figure 4-3 ntop Web interface: Network Load Statistics

Other Open Source Software related to network monitoring

Here is some of the Open Source Software that runs on Linux and AIX. For AIX installations, you can download these packages from the AIX toolbox for Linux applications Web site, found at:

http://www.ibm.com/servers/aix/products/aixos/linux/

- Ethereal network analyzer

  This is a graphical tool used to capture and analyze network traffic. To get more information and download this package, go to:

  http://www.ethereal.com/

- tcpdump utility

  This is a tool for network monitoring and data acquisition. A good reference for this tool can found at the following Web sites:

  http://www.tcpdump.org/
  http://www-iepm.slac.stanford.edu/monitoring/passive/tcpdump.html

4.3.4 UNIX network performance management commands

`netstat -i #`	Reports number of packets transmitted and received by interfaces.
`netstat -i -Ien0 #`	Reports number of packets transmitted and received by interface en0 (for AIX).
`netstat -m`	Reports streams allocations (for AIX).
`netstat -s`	Displays networking statistics, such as SNMP (for Linux).
`no`	Manages network tuning parameters (for AIX).
`/proc/sys/net/ipv4`	Includes parameters to tune TCP (Linux).
`tcpdump`	Captures packets off of a network interface, and interprets and prints out the headers.

5

Sendmail

The mail facility provides a method for exchanging electronic mail between the users on the same system or on multiple systems connected by a network. This chapter discusses configuring Sendmail on AIX and Linux systems.

We assume readers will have some familiarity with Sendmail and with mail administration in general. To learn about mail administration and Sendmail, you can refer to Web sites and documentations like:

- Sendmail Web site

  http://www.sendmail.org/

- AIX documentation: See the “Mail” section of the *AIX 5L Version 5.2 System Management Guide: Communications and Networks*

Sendmail in itself is a vast topic, and there are many books written just for Sendmail and the various options for configuring Sendmail. We do not go into the details of configuring Sendmail, as it is beyond the scope of this book. We however, cover basic issues of implementing Sendmail on AIX and Linux. The topics covered in this chapter are:

- Sendmail overview
  - Installing Sendmail
  - Configuration file: sendmail.cf
  - Start and stop Sendmail

  - Mail aliases file
  - Mail queue
  - Mail logging
- Mail server on AIX
  - Configuring mail server on AIX
  - Sending mail from Linux to AIX server
- Mail server on Linux

5.1 Sendmail overview

Sendmail is a mail transport agent developed by Eric Allman that handles e-mail from the moment a user submits a mail, transporting it across a network (LAN or Internet) to the proper destination system. The final e-mail delivery to the user's mailbox is done by a delivery agent. The Sendmail package includes the delivery agent, but its use is optional.

We discuss the installation and configuration of Sendmail on AIX and Linux in this section.

5.1.1 Installing Sendmail

Sendmail is the traditional UNIX transport agent and is installed and started at boot time by default on AIX and on most of the Linux distributions. Refer to the Sendmail Web site for instructions on how to install Sendmail from source code at:

http://www.sendmail.org

For Linux systems, you can install Sendmail through RPM package. You can download the Sendmail RPM package from:

http://speakeasy.rpmfind.net/linux/rpm2html/search.php?query=Sendmail

After downloading the appropriate RPM, install it using the **rpm -i** command. Refer to the man page for rpm for more details on RPM options.

In addition to installing Sendmail, you should also install the sendmail-cf package, which contains support for the m4 macro language based configuration.

The sendmail-cf package is installed by default on AIX, and is present at /usr/samples/tcpip/sendmail/ on AIX Version 5.2.0. Sendmail is started from the /etc/rc.tcpip script.

Most of the Linux distributions provide the sendmail-cf RPM on the installation CD. You can also download the RPM from:

http://www.rpmfind.net/linux/rpm2html/search.php?query=sendmail-cf

After you download the appropriate RPM for your Linux distribution and system architecture, install it using the **rpm -i** command.

5.1.2 Configuration file: sendmail.cf

Sendmail uses a configuration file called sendmail.cf, which is generally stored in /etc/sendmail.cf in AIX and in /etc/mail/sendmail.cf in Linux.The Sendmail program uses a complex proprietary macro language. An easier way to prepare a Sendmail configuration is to use the m4 macro language. The m4 sendmail macro package is delivered with most contemporary Sendmail installations, including both AIX and Linux. The sendmail m4 configuration files have the .mc extension. Platform specific sample files are usually included and they may be modified to suit the specific site's needs. To generate the Sendmail configuration file, it is sufficient to preprocess the .mc file using the m4 program. Find a macro file with .mc extension on your system and edit it according to your needs. After editing, generate the required sendmail.cf file using m4 macro. An overview of this process is described below:

1. Find a file with the .mc extension

   On AIX, you can find an example of .mc file at:

   ```
   # find / -name "*.mc"
   /usr/samples/tcpip/sendmail/cf/aixsample.mc
   ```

   On Linux, you can find the .mc files in a similar way.

2. Read the .mc file with its comments. Note that the order of items is significant. Some commonly used variables and macros are shown in Table 5-1.

Table 5-1 Variables and macros

Variable/Macro	Description
VERSIONID	This variable identifies the configuration file.
OSTYPE	Specifies the operating system type.
FEATURE	Specifies inclusion of a code representing some Sendmail's feature. For example, FEATURE(‘local_procmail’) says to make procmail the default local delivery agent.
define	Used to define the values of variables used by Sendmail.
MAILER	Lists supported mail delivery agents.

3. After editing the .mc file according to your configuration, you should run the following command to generate the sendmail.cf file:

   Important: It is a good idea to make a backup copy of the sendmail.cf file before generating a new sendmail.cf file, just in case you have to revert to your old configuration.

   ```
   # m4 ${CFDIR}/m4/cf.m4 config.mc > config.cf
   ```

Where CFDIR is the root of the cf directory and config.mc is the name of your configuration file. Refer to http://www.sendmail.org/m4/intro.html for more information on using the m4 macro language for Sendmail configuration.

A detailed example of setting up the sendmail.cf file from a macro file with .mc extension is shown in 5.2, “Mail server on AIX” on page 119.

After you edit the sendmail.cf file, you should restart Sendmail.

5.1.3 Start and stop Sendmail

When you make any changes to the sendmail.cf file, you should restart Sendmail to make those changes effective.

On AIX, execute:

```
# refresh -s sendmail
```

or

```
# stopsrc -s sendmail
# startsrc -s sendmail -a “-bd -q15m”
```

On Linux, depending on your Linux distribution, restart Sendmail. For example, on Red Hat Linux, execute:

```
# /etc/rc.d/init.d/sendmail restart
```

On SuSE Linux, execute:

```
# /etc/rc.d/sendmail restart
```

5.1.4 Mail aliases file

In the mail aliases file, we define the system-wide and domain-wide mail aliases.

AIX keeps mail aliases in /etc/aliases. In addition to aliases you provide, this file must contain an alias for each of the following:

- MAILER-DAEMON (usually set to root)
- postmaster (usually set to root)
- nobody (usually set to /dev/null)

To compile the aliases database file, type one of the following:

```
# sendmail -bi
# newaliases
```

This creates a dbm version of the aliases file stored at /etc/mail/aliases.db. For more information, refer to the “Managing Mail Aliases” section of the *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

On Linux, the aliases file is stored at /etc/aliases. After editing the /etc/aliases file, you should run:

```
# newaliases
```

For more information, run **man 5 aliases**.

5.1.5 Mail queue

On AIX, the mail queue is a directory that stores data and controls files for mail messages that the **sendmail** command delivers. By default, the mail queue is /var/spool/mqueue. You can read more in the “Managing the Mail Queue Files and Directories” section in the *AIX 5L Version 5.2 System Management Guide: Communications and Networks*.

On Linux, the mail queue directory is /var/spool/mqueue. You can use the **mailq** command to see the queue status on Linux.

5.1.6 Mail logging

The **sendmail** command logs mail system activity through the syslogd daemon. The syslogd daemon must be configured and running for logging to occur. Specifically, the /etc/syslog.conf file should contain the uncommented line:

```
mail.debug          /var/spool/mqueue/log
```

If it does not, use your favorite editor to make this change; be certain that the path name is correct and that the mail log file exists before you restart syslog. If you change the /etc/syslog.conf file while the syslogd daemon is running, restart the syslogd daemon.

On AIX, if the /var/spool/mqueue/log file does not exist, you must create it by typing the following command:

```
# touch /var/spool/mqueue/log
```

Type the following command at a command line:

```
# refresh -s syslogd
```

On Linux, go to the /etc/rc.d/init.d or /etc/rc.d directory, depending on your Linux distribution and restart the syslog service:

```
# Log all the mail messages in one place.
mail.*                                                  /var/log/maillog
```

5.2 Mail server on AIX

In this section, we describe a scenario in which AIX acts as a mail server or mail hub for an internal network of computers. All the mail from this network is handled by AIX.

5.2.1 Configuring mail server on AIX

We describe an example of generating a basic sendmail.cf file for a mail hub. The sendmail.cf file is generated using the m4 macro.

You can find an example of a macro file on AIX 5L Version 5.2.0 at /usr/samples/tcpip/sendmail/cf/aixsample.mc. You should edit this file or create a new .mc file.

We show a sample file (/etc/mail/sendmail_hub.mc) in Example 5-1.

Example 5-1 The configuration file in the .mc extension for the mail hub

```
VERSIONID(Config file for AIX hub)
OSTYPE(aix)dnl
FEATURE(use_cw_file)
MASQUERADE_AS(foo.com)
FEATURE(allmasquerade)
MAILER(smtp)dnl
MAILER(local)dnl
```

Important: Make sure you type the right OSTYPE macro value and change the file name in the /usr/samples/tcpip/sendmail/ostype directory. In our example, since we entered `aix` in the .mc file, we *must* rename the /usr/samples/tcpip/sendmail/ostype/aixsample.m4 to aix.m4.

The main options of the .mc file are described in the Table 5-1 on page 116.

The use_cw_file feature specifies an external file that contains the hosts and domains that this system will accept and deliver mail locally. The default file for this purpose is /etc/mail/local-host-names. The entries in this file can be similar to Example 5-2.

Example 5-2 Example of the /etc/mail/local-host-names file

```
localdomain.com
ibm.com
foo.com
```

The MASQUERADE_AS macro removes all references to any local computer and makes all the departing messages appear as if they are coming from the user at foo.com. The allmasquerade feature masquerades recipient names as well as sender names.

The MAILER macros activate the delivery agents, Simple Mail Transfer Protocol (SMTP), and the default local delivery agent. Putting all MAILERs at the end is important.

There are several more options that can be used to fine tune Sendmail according to your needs. You can find out about these options from http://www.sendmail.org/m4/intro.html or from the README included in the Sendmail documentation. For example, in AIX 5L Version 5.2.0, you can find the README in the /usr/samples/tcpip/sendmail directory.

We may produce the Sendmail configuration file by using the m4 macro preprocessor. This can be done as follows:

1. Make a backup of the current sendmail.cf file:

   ```
   # cd /etc/mail
   # mv sendmail.cf sendmail.cf.orig
   ```

2. Change the working directory to /usr/samples/tcpip/sendmail/m4/cf and execute the following commands:

   ```
   # cd /usr/samples/tcpip/sendmail/cf
   # m4 ../m4/cf.m4 /etc/mail/sendmail_hub.mc >/etc/mail/sendmail.cf
   ```

   Due to the coding style of m4 macros in the AIX's Sendmail package, it is necessary to preprocess the .mc files from the specified directory. Alternatively, one may specify the configuration directory when invoking the m4 program:

   ```
   -D_CF_DIR_=/usr/samples/tcpip/sendmail/
   ```

3. To make Sendmail read from the configuration file we created, restart Sendmail. Refer to 5.1.3, “Start and stop Sendmail” on page 117 for details.

We now configure the Linux system to use AIX as the mail hub.

5.2.2 Sending mail from Linux through an AIX hub

In this section, we describe the steps required to send mail from Linux to non-local domains using AIX as the mail hub.

The steps to do the same are as follows:

1. Edit or create a sample .mc file, for example, /etc/mail/sendmail_client.mc, that has the contents shown in Example 5-3 on page 121.

Example 5-3 Example of the /etc/mail/sendmail.mc file acting as a mail client

```
include(`/usr/share/sendmail-cf/m4/cf.m4')
VERSIONID(Config file for Red Hat Linux)
OSTYPE(`linux')
FEATURE(`smrsh')
define('PROCMAIL_MAILER_PATH',`/usr/bin/procmail')
FEATURE(`local_procmail')
FEATURE(`nocanonify')
define(`SMART_HOST',`aix_host.local-domain.com.')
MAILER(`smtp')
MAILER(`procmail')
```

The nocanonify feature tells Sendmail not to expand e-mail addresses to their fully qualified form on the local system.

The SMART_HOST feature forwards all non-local domain e-mail to the mail hub aix_host.local-domain.com.

Refer to Table 5-1 on page 116 for detailed explanations.

2. Make a backup of the current/etc/mail/sendmail.cf file:

   ```
   # mv /etc/mail/sendmail.cf /etc/mail/sendmail.cf.orig
   ```

3. Convert this /etc/mail/sendmail_client.mc file to /etc/sendmail.cf file by executing:

   ```
   # m4 ../m4/cf.m4 /etc/mail/sendmail_client.mc >/etc/mail/sendmail.cf
   ```

4. Restart Sendmail. Refer to 5.1.3, “Start and stop Sendmail” on page 117 to know how to restart Sendmail.

Now all the e-mail sent to users outside the local-domain is forwarded to the mail hub on AIX, which forwards those e-mails to the right mail servers, depending upon the domain.

5.3 Mail server on Linux

The scenario of Linux acting as a mail hub and AIX sending non-local domain mails through this mail hub is similar to that described in 5.2, “Mail server on AIX” on page 119 and 5.2.2, “Sending mail from Linux through an AIX hub” on page 120. The functions of AIX and Linux are interchangeable, but the steps to generate the Sendmail configuration file for the mail hub and mail client remain almost similar.

The main differences for this scenario are the OSTYPE macro in the .mc file used to generate the Sendmail configuration file and the location of the m4

macro files, which are generally installed in the /usr/share/sendmail-cf directory in Linux.

6

Samba file and print server

Samba is an Open Source/Free Software suite and allows clients to share file and print services via the Server Message Block (SMB) and Common Internet File Systems (CIFS) protocols. Samba is used mostly for Windows and UNIX integration, but can also be used between AIX and Linux. We describe Samba, because it is a very useful and popular software extensively implemented in a mixed operating systems environment. In this chapter, we discuss Samba as a file and print sharing example for Linux and AIX systems. We do not describe the details about Samba and the various configuration options associated with Samba, as they are already well documented and available on Samba's Web site at http://www.samba.org.

or in the redbook *Samba Installation, Configuration, and Sizing Guide,* SG24-6004, which can be found on the IBM Redbooks site at:

http://www.redbooks.ibm.com/pubs/pdfs/redbooks/sg246004.pdf

Also, refer to the redbook *AIX 5L and Windows 2000: Solutions for Interoperability,* SG24-6225, available at:

http://www.redbooks.ibm.com/pubs/pdfs/redbooks/sg246225.pdf

This chapter describes following topics:

- Installing Samba on AIX
- Installing Samba on Linux
- Samba file and print server on AIX and Samba client on Linux
- Samba file and print server on Linux and Samba client on AIX

6.1 Installing Samba on AIX

You can install Samba by using precompiled binary packages or by installing from the source code. We describe how to install Samba using RPM. For other options, refer to the *Samba Installation, Configuration, and Sizing Guide,* SG24-6004. Also, refer to the redbook *AIX 5L and Windows 2000: Solutions for Interoperability,* SG24-6225.

You can check if the Samba RPM package is installed on your system by executing the following command:

```
# rpm -qa | grep samba
samba-2.2.3a-2
samba-client-2.2.3a-2
samba-common-2.2.3a-2
```

If you get a similar output to the preceding one, then Samba is already installed. If not, then you can install Samba RPM packages from the AIX installation CD.

Once you locate the Samba RPMs on your CD, you can install Samba by doing the following steps:

```
# rpm -i /tmp/samba-2.2.3a-2.aix4.3.ppc.rpm
# rpm -i /tmp/samba-client-2.2.3a-2
# rpm -i /tmp/samba-common-2.2.3a-2

0513-095 The request for subsystem refresh was completed successfully.
```

In this case, the Samba RPMs are stored in the /tmp directory.

6.2 Installing Samba on Linux

Samba can be installed using RPM much like it is for AIX. You can also install Samba from source. Refer to the documentation available on the Samba Web site to know how to install from Samba from source:

http://www.samba.org

The RPMs for Samba on Linux are available at:

http://www.rpmfind.net/linux/rpm2html/search.php?query=Samba+

After downloading the appropriate RPMs according to your Linux distribution and system's architecture, you can install them by using the **rpm** command. For example:

```
# rpm -i /var/tmp/samba-common-2.0.10-2.i386.rpm
# rpm -i /var/tmp/samba-client-2.0.10-2.i386.rpm
```

```
# rpm -i /var/tmp/samba-2.0.10-2.i386.rpm
# rpm -i /var/tmp/samba-swat-2.0.10-2.i386.rpm
```

This installs all the required Samba utilities.

6.3 Samba file and print server on AIX and client on Linux

This section describes the setup of a Samba file and print server on AIX and how to access the SMB shares from a Linux client using commands like **smbclient** and **smbmount**, and how to print to a printer attached to AIX using a smbprint filter.

6.3.1 Configuring a Samba server on AIX

A Samba server can be started either through inetd or as a daemon. The choice depends upon how often the Samba server is required to provide the services. If the requests are *substantial*, then we recommend starting the Samba service as a daemon. In that case, ensure that the related smbd and nmbd lines are commented in the /etc/inetd.conf file and smbd is started with a -D option. A typical /etc/inetd.conf file entry for Samba looks like Example 6-1.

Example 6-1 /etc/inetd.conf entries to start Samba server as a daemon

```
#netbios-ssn stream tcp nowait root /usr/sbin/smbd smbd
#netbios-ns dgram upd wait root /usr/sbin/nmbd nmbd
swat      stream  tcp     nowait.400      root /usr/sbin/swat swat
```

To start the smbd server as a daemon, run the following command:

```
# /usr/sbin/smbd -D
```

Samba is configured using a single configuration file. It is a plain text file that can be edited manually or through the Web-based graphical user interface, Samba Web Administration Tool (SWAT). The file, generally stored at /etc/smb.conf or /etc/samba/smb.conf, has a number of options, and a great deal of fine tuning can be achieved. To know more about the various options, you can refer to the documentation provided by the SWAT interface or the manual pages by typing the **man smb.conf** command.

We describe an example of creating a file share and configuring a printer on AIX in the following steps:

1. Verify that the server is started and listening on the SWAT port (901). You can do that by typing the command shown below:

   ```
   # netstat -an | grep 901
   ```

Check for an output similar to:

```
tcp4       0      0  *.901                  *.*                    LISTEN
```

This shows that the server is listening on port 901.

2. Using a Web browser, go to:

   `http://hostname or IP address:901`

3. A window appears requiring an user ID and password. Type in the user ID as root and with root's password. You should see Figure 6-1.

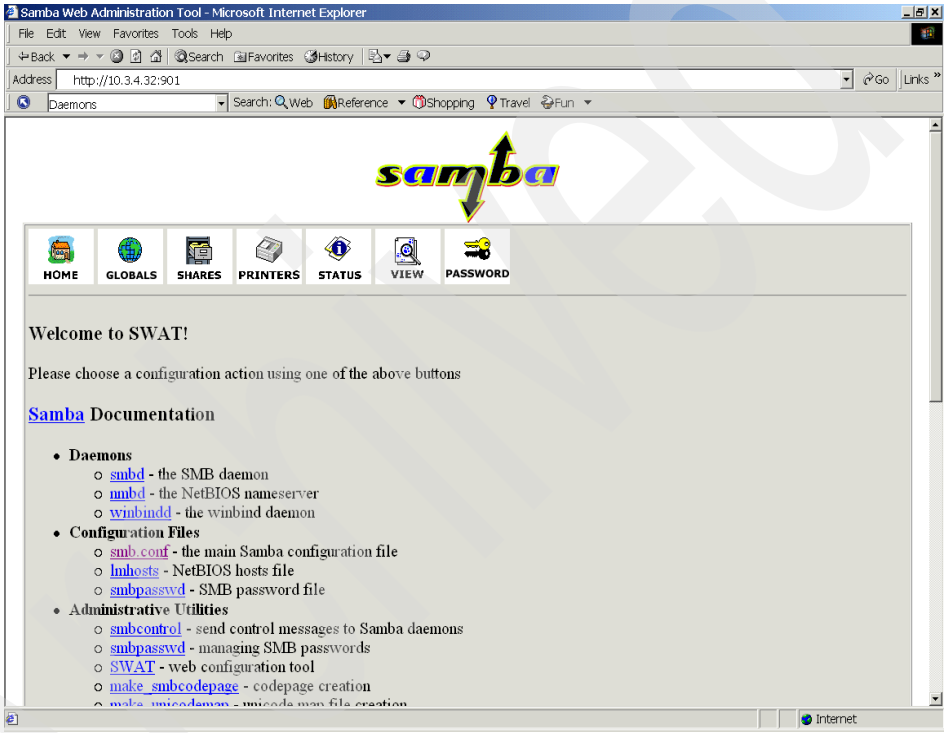

Figure 6-1 Starting page of SWAT

4. The explanations for the various options can be read in the documentation provided through the SWAT interface or you can refer to the redbook *Samba Installation, Configuration, and Sizing Guide,* SG24-6004 or to the Samba server section of the redbook *AIX 5L and Windows 2000: Solutions for Interoperability*, SG24-6225. The links are given at the start of this chapter.

5. Next, we configure the Shares. Each section in the configuration file smb.conf (except the [global] section) describes a shared resource and the parameters within the section define the share attributes. Go to the Shares page by clicking on **Shares**. You should see Figure 6-2 on page 127.

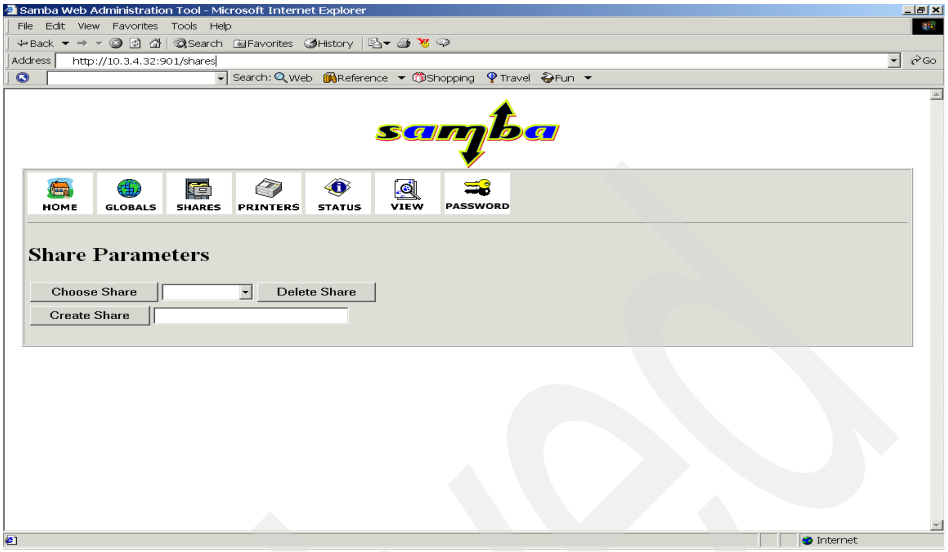

Figure 6-2 Creating new shares through SWAT

6. Type in the share name `homes` in the **Create Share** space and then click **Create Share**. You should see Figure 6-3 on page 128.

   The homes share makes it possible to access the home directories of the users on the Samba server, depending upon the user ID and appropriate password used while connecting to the Samba server. We describe how to access the home directories from the Samba client in step 2 on page 132.

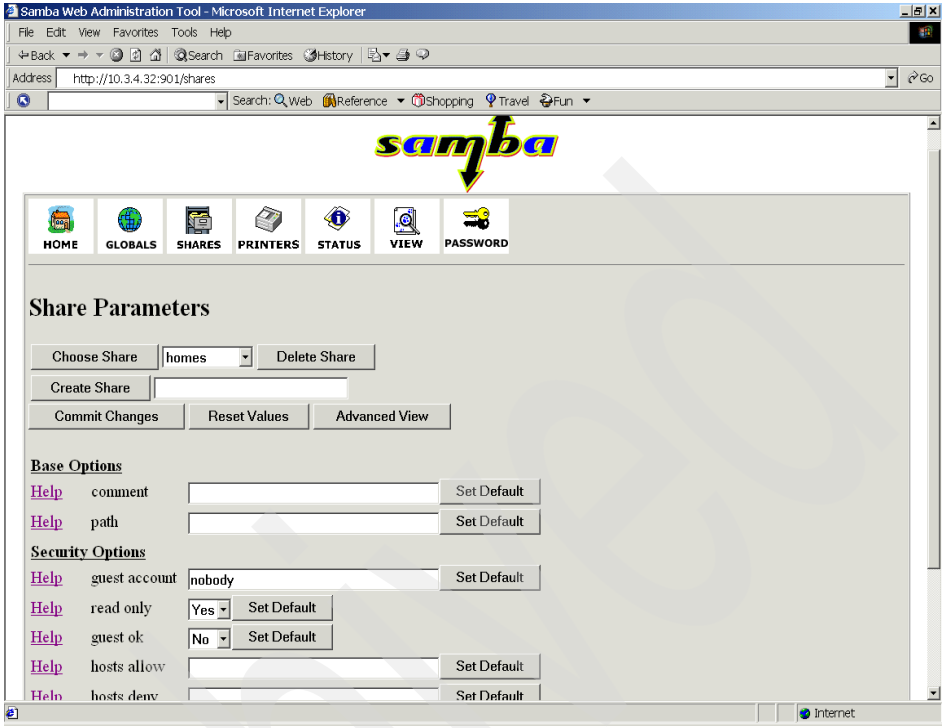

Figure 6-3 Creating homes share through SWAT

The default options are appropriate most of the time. You can find out more about the options from the documentation given with SWAT.

7. To set up the printer as a share, follow these steps:

   a. Go to the printers section by clicking the **Printers** icon on any of the SWAT interfaces. You will see Figure 6-4 on page 129.

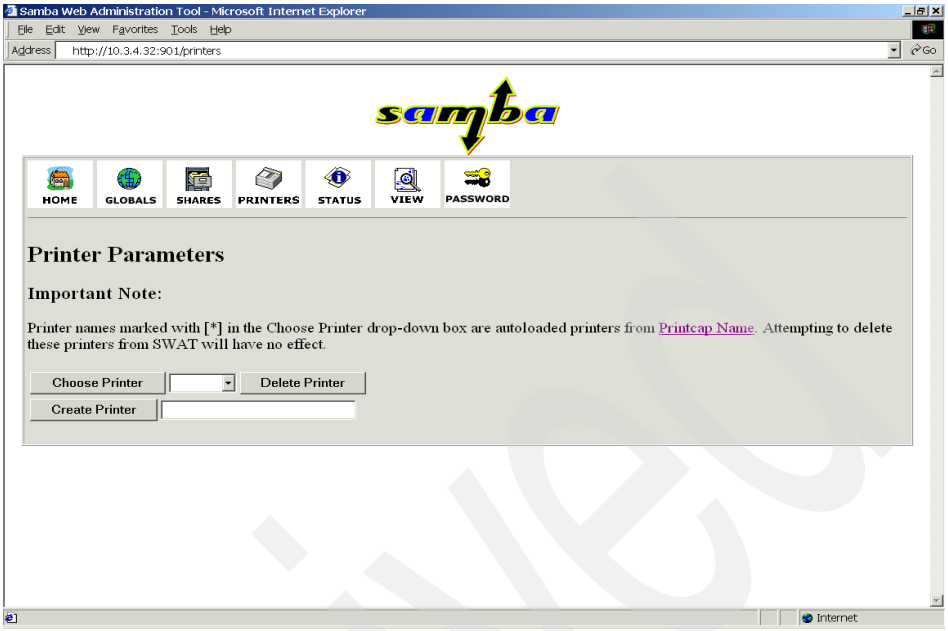

Figure 6-4 SWAT Printers section

b. Choose the appropriate printer from the **Choose Printer** drop-down menu. You should see all the print queues on the AIX system in the drop-down menu. This list is taken from the /etc/qconfig file. After selecting the printer and clicking on **Choose Printer**, you will see Figure 6-5 on page 130.

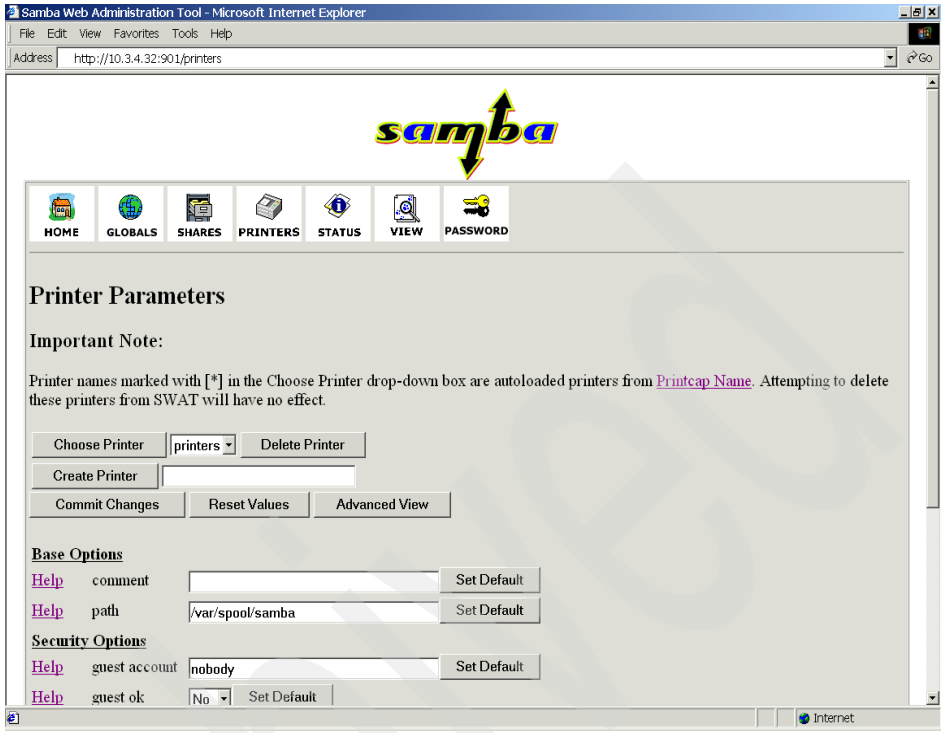

Figure 6-5 Configuring print shares on the Samba server

c. You can configure the printer settings now. When you are done, save the settings by clicking the **Commit Changes** button.

d. In the **Globals** -> **Advanced View** section of SWAT, check the **Printing Options** section. It should have entries relevant to the AIX Operating System, as shown in Figure 6-6 on page 131. Generally, all these entries are filled automatically by Samba when you make a print share.

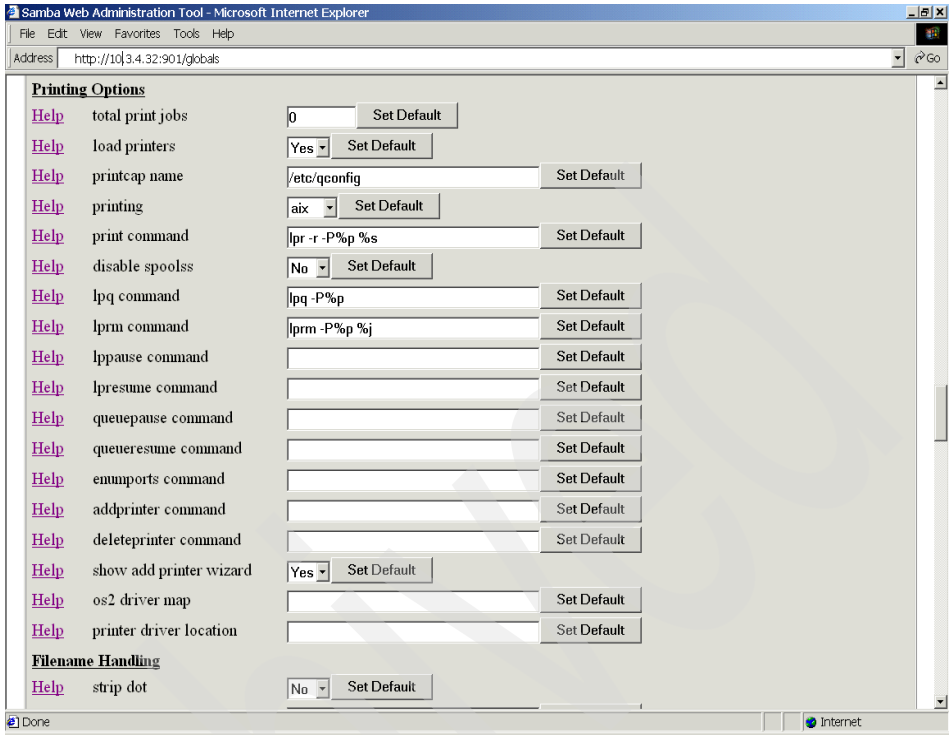

Figure 6-6 Printer settings for AIX on the SWAT Global printer options

This sets up the Samba file and print server on AIX.

6.3.2 Configuring a client on Linux

We describe the client configuration by continuing with the preceding example. In order to access the home shares we just created on AIX, we need to have smbclient and smbmount installed on Linux.

To access the Samba file server on AIX, follow these steps:

1. Verify the Samba connection between Linux and AIX by running the **`smbclient`** command.
   a. To know which shares are accessible on a host, run the following command:

      `/usr/bin/smbclient -L` *`host`*

      where host is the resolvable name or IP address of machine running the Samba server.

      We type in the **`smbclient`** command, as shown in Example 6-2.

Example 6-2 Verifying the shares available on AIX

```
# smbclient -L aix_host -U bob
added interface ip=10.3.5.23 bcast=10.3.5.255 nmask=255.255.254.0
Password:
Domain=[WORKGROUP] OS=[UNIX] Server=[Samba 2.2.3a]

        Sharename      Type      Comment
        ---------      ----      -------
        samba_test     Disk
        tmp            Disk
        homes          Disk
        IPC$           IPC       IPC Service (Samba 2.2.3a)
        ADMIN$         Disk      IPC Service (Samba 2.2.3a)

        Server               Comment
        ---------            -------
        SRVR5OF              Samba 2.2.3a

        Workgroup            Master
        ---------            -------
        WORKGROUP
```

This shows that samba_test, tmp and homes are available shares on the aix_host system.

smbclient can be used like the ftp protocol to transfer files between the client and server. Run **man smbclient** for more details.

In case you do not get this output, check the installation of the Samba server on AIX. Also, check the Samba log files and the online FAQ on the Samba Web site.

2. Accessing the homes share of users through **smbmount**.

   a. Type in the **smbmount** command as follows:

   ```
   # /usr/bin/smbmount //aix_host/bob /samba_mount -o username=bob
   Password:
   ```

   The password typed in is the password for that user on AIX. You can also type the numerical user ID and group ID of the user instead of the user name. For more details, run **man smbmount**.

   This command mounts the home directory /home/bob on the aix_host system onto the /samba_mount directory of the Linux system.

> **Note:** The **smbmount** command is generally executed by root. In order to allow other users to do **smbmount** and **smbumount**, they must be installed as suid root.

b. You can verify the smb mounted directory by using the **mount** command, as shown in Example 6-3.

The output shows, among other things, that /home/bob on system aix_host is mounted on /samba_mount on local system.

Example 6-3 Mount status after executing smbmount command

```
# mount
/dev/sda5 on / type ext3 (rw)
none on /proc type proc (rw)
usbdevfs on /proc/bus/usb type usbdevfs (rw)
/dev/sda1 on /boot type ext3 (rw)
none on /dev/pts type devpts (rw,gid=5,mode=620)
/dev/sda3 on /home type ext3 (rw)
none on /dev/shm type tmpfs (rw)
/dev/sda6 on /var type ext3 (rw)
//aix_host/bob on /samba_mount type smbfs (0)
```

c. Once you mount the directory on /samba_mount, then you can navigate that directory. If the share homes has the read only option set to `Yes`, then you cannot create new files or delete the files. For example:

```
# cd /samba_mount
# ls
file1 file2 testdir
# rm file1
rm: cannot remove `file1': Permission denied
```

If you change the read only option on AIX Samba server for homes share to `No`, restart the smbd daemon and try the same thing again; the output is then as follows:

```
# cd /samba_mount
# ls
file1 file2 testdir
# rm file1
# ls
file2 testdir
```

3. Printing from Linux to a printer attached to AIX using the SMB protocol.

a. To print from the Linux Samba client, we use the smbprint filter. Your Linux distribution may or may not include the smbprint filter. The smbprint filter source code is shown in Example 6-4. Make sure that smbprint is placed in /usr/bin/smbprint and has executable permission.

Example 6-4 smbprint filter

```
#!/bin/sh
# This script is an input filter for printcap printing on a unix machine. It
# uses the smbclient program to print the file to the specified smb-based
```

```
# server and service.
# For example you could have a printcap entry like this
#
# smb:lp=/dev/null:sd=/usr/spool/smb:sh:if=/usr/local/samba/smbprint
#
# which would create a unix printer called "smb" that will print via this
# script. You will need to create the spool directory /usr/spool/smb with
# appropriate permissions and ownerships for your system.

# Set these to the server and service you wish to print to
# In this example I have a WfWg PC called "lapland" that has a printer
# exported called "printer" with no password.

#
# Script further altered by hamiltom@ecnz.co.nz (Michael Hamilton)
# so that the server, service, and password can be read from
# a /usr/var/spool/lpd/PRINTNAME/.config file.
#
# In order for this to work the /etc/printcap entry must include an
# accounting file (af=...):
#
#   cdcolour:\
#       :cm=CD IBM Colorjet on 6th:\
#       :sd=/var/spool/lpd/cdcolour:\
#       :af=/var/spool/lpd/cdcolour/acct:\
#       :if=/usr/local/etc/smbprint:\
#       :mx=0:\
#       :lp=/dev/null:
#
# The /usr/var/spool/lpd/PRINTNAME/.config file should contain:
#   share=PC_SERVER
#   user="user"
#   password="password"
#
# Please, do not modify the order in the file.
# Example:
#   share=\\server\deskjet
#   user="fred"
#   password=""

#
# The last parameter to the filter is the accounting file name.
#   Extract the directory name from the file name.
#   Concat this with /.config to get the config file.
#
eval acct_file=\${$#}
spool_dir=`dirname $acct_file`
config_file=$spool_dir/.config
```

```
# Should read the following variables set in the config file:
#   share
#   hostip
#   user
#   password

eval `cat $config_file`

share=`echo $share | sed "s/[\]/\//g"`

if [ "$user" != "" ]; then
   usercmd="-U"
else
usercmd=""
fi

if [ "$workgroup" != "" ]; then
   workgroupcmd="-W"
else
   workgroupcmd=""
fi

if [ "$translate" = "yes" ]; then
 command="translate ; print -"
else
 command="print -"
fi
#echo $share $password $translate $x_command > /tmp/smbprint.log

cat | /usr/bin/smbclient "$share" "$password" -E ${hostip:+-I} \
     $hostip -N -P $usercmd "$user" $workgroupcmd "$workgroup" \
     -c "$command" 2>/dev/null
```

Note:

- The $# special variable may evaluate to a two-digit number. Therefore, it should be surrounded by the curly braces, such as:

  ```
  eval acct_file=\${$#}
  ```

- The .config file must follow the shell rules for variable assignment.

b. As described in the smbprint filter, you need to edit the /etc/printcap file on the Linux system. A typical entry in the /etc/printcap file would appear as:

```
lp|pclq|ibm4340:\
        :mx#0:sh:lp=/dev/null:sd=/var/spool/lpd/pclq:\
        :af=/etc/lpd/pclq/acct:if=/usr/bin/smbprint:
```

Where pclq is the printer share name, ibm4340 is the printer name, /var/spool/lpd/pclq is the spool directory on the local system, and /etc/lpd/pclq/acct is the account file on the local system.

You need to create the spool and account directories with the appropriate permissions.

c. After editing the printcap file, restart the lpd daemon from the /etc/rc.d/init.d or /etc/rc.d/ directory, depending upon your Linux distribution. For example, for Red Hat Linux, you can start lpd by running:

```
# /etc/rc.d/init.d/lpd start
```

d. Create a .config file in the account directory you mentioned in the /etc/printcap file. For example, we create the /etc/lpd/pclq/.config file with the following entries:

```
share=//srvr50f/pclq
hostip=10.3.4.32
user=
password=
```

Where srvr50f is the server name and pclq is the print share name.

e. You can now print from a Linux system to a printer attached to AIX using the **lpr** command:

```
# lpr /etc/printcap.local
```

You can check the print queue by running the **lpq** command. You should get a similar output to the following output:

```
# lpq
Printer: lp@localhost 'ibm4340'
 Queue: no printable jobs in queue
 Server: no server active
 Status: job 'root@localhost+20' saved at 16:39:53.333
 Rank   Owner/ID                  Class Job Files                 Size
Time
done   root@localhost+20            A    20 /etc/printcap.local    603
16:39:52
```

6.4 Samba file and print server on Linux and client on AIX

In this section, we discuss the scenario of a Linux machine hosting the Samba server and AIX acting as a Samba client and accessing the shares by using commands like **smbtar** and **smbprint**.

6.4.1 Configuring Samba server on Linux

As described earlier, the Samba server is configured using the smb.conf file. You can edit the smb.conf manually using your favorite editor or using the SWAT interface.

To set up the SWAT interface, after installing the SWAT package, you need to make changes in the inetd.conf file. The format of inetd.conf depends upon the Linux distribution you are using. For example, in Red Hat Linux Version 8.0, create /etc/xinetd.d/swat with the contents shown in Example 6-5.

Example 6-5 /etc/xinetd.d/swat entry in Red Hat

```
service swat
{
        disable = no
        socket_type             = stream
        wait                    = no
        user                    = root
        server                  = /usr/sbin/swat
        nice                    = 10
}
```

You need to restart the inetd daemon after making changes to /etc/inetd.conf. You can do this by going to the /etc/rc.d/init.d or /etc/rc.d/ directory, depending upon your Linux distribution, and restarting the inetd or xinetd daemon by running:

```
# /etc/rc.d/init.d/xinetd restart
```

The configuration of the Samba server is similar to that described for AIX in 6.3.1, “Configuring a Samba server on AIX” on page 125. You can configure smb.conf through SWAT or by manually editing it using your favorite editor. The location of smb.conf file depends upon the Linux distribution. For example, in Red Hat Linux, it is present in /etc/samba/smb.conf, and on SuSE Linux, it is present in /etc/smb.conf.

You can use the SWAT interface to configure the file share and print share, as described in 6.3.1, “Configuring a Samba server on AIX” on page 125 by using the SWAT interface.

Example 6-6 shows an smb.conf file.

Example 6-6 A typical smb.conf file

```
# Global parameters
[global]
        workgroup = MYGROUP
```

```
        server string = Samba Server
        max log size = 0
        socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
        dns proxy = No
        printing = lprng

[homes]
        comment = Home Directories
        writeable = Yes
        browseable = No

[printers]
        comment = All Printers
        path = /var/spool/samba
        printable = Yes
        browseable = No
```

Note: Whenever you change the smb.conf file, you should execute the **testparam** command to check for any errors and then restart the Samba server.

You can verify if the Samba server is running by running:

```
# ps -auwwwx | grep [s]mbd
root      1614  0.0  0.5  2908 1084 ?        S    11:09   0:00 smbd -D
```

If you get a similar output, then the Samba server on Linux is started.

6.4.2 Configuring a Samba client on AIX

A Samba client is installed by default on AIX. In this section, we describe examples of accessing SMB shares:

- Using SMBFS
- Using the **smbtar** command

Using SMBFS

You can use the new Server Message Block File System (SMBFS) feature, which is available on AIX 5L Version 5.2.0. SMBFS allows access to shares on SMB servers as a local file system on AIX. Furthermore, you can create, delete, read, write, and modify the access times of files and directories. The owner or access mode of files and directories cannot be changed.

Installing SMBFS

SMBFS can be installed from the base operating system CD by using the following command:

```
# installp -ac -d /dev/cd0 bos.cifs_fs
```

When installing the bos.cifs_fs fileset, the following components are installed:

- SMIT panels
- The /usr/lib/drivers/nsmbdd device driver
- The /usr/lib/methods/cfgnsmb configuration method
- The /sbin/helpers/mount_cifs mount helper
- The /etc/mkcifs_fs boot time script

Furthermore, the /dev/nsmb0 device is created and is always available. At boot time, this device is created by the /etc/mkcifs_fs script.

> **Note:** SMBFS in only supported on a 32-bit kernel, so an installation on a 64-bit kernel will fail.

Mounting a file system

To mount an SMBFS file system, as with any other file system, the **mount** command should be used. For the mount of an SMBFS file system, the following syntax is applicable:

```
mount -v cifs -n Node [-o Options ] Share Directory
```

The -v cifs option specifies the file system as defined by the VfsName parameter in the /etc/vfs file. Read the mount manual pages for more details.

For example, to mount the share export on the node linux_host and connect to the share with the user name dave and the password xyz123 under the mount point /mnt, the following command should be used:

```
# mount -v cifs -n linux_host/dave/xyz123 /export /mnt
```

> **Note:** The SMBFS cannot be automatically mounted with the /etc/filesystem stanza. This limitation occurs due to the need for passwords.

Using smbtar

We are using the **smbtar** command to make a backup of the tmp share stored on the Linux system to the tape drive on the AIX system. The various options of **smbtar** can be found in its the man pages. We describe a few of those options, which we use for the backup.

- -s server

  The SMB/CIFS server that the share resides on.

- -x service

The share name on the server. The default is backup.

- -d directory

  Changes the initial directory before restoring/backing up files.

- -p password

  The password to access the share.

- -u user

  The user ID to connect as; the default is the UNIX login name.

- -t tape

  The tape device. It can also be a regular file. The default is to use the environment variable TAPE; if it is not set, a file called tar.out is created.

- -r restore

  Files are restored to the share from the tar file.

The command to start the backup is:

```
# smbtar -v -s linux_server -x tmp -p password -d /var/tmp -t /dev/rmt0
```

The -v option is for verbose mode, which will help in the case of errors. The password typed in is a plain text password. You should get an output similar to Example 6-7.

Example 6-7 Output of smbtar command

```
server    is redhat
share     is tmp\/var/tmp
tar args  is
tape      is /dev/rmt0
blocksize is
added interface ip=10.3.4.32 bcast=10.3.4.255 nmask=255.255.254.0
tarmode is now full, system, hidden, noreset, verbose
Domain=[MYGROUP] OS=[UNIX] Server=[Samba 2.0.10]
     34816 (  971.4 kb/s) \WordDocument.doc
     12577 ( 1364.7 kb/s) \WordDocument.sxw
     41472 ( 1265.6 kb/s) \WordDocument1.doc
   2595888 (  552.3 kb/s) \samba-2.0.10-2.i386.rpm
    776045 (  954.5 kb/s) \samba-client-2.0.10-2.i386.rpm
    968670 (  928.3 kb/s) \samba-common-2.0.10-2.i386.rpm
   3273161 (  924.4 kb/s) \samba-swat-2.0.10-2.i386.rpm
tar: dumped 7 files and directories
Total bytes written: 7704576
```

This shows that the contents of the tmp share on Linux have been backed up on a tape of the AIX system.

To restore the share, type in the following command:

```
# smbtar -v -s linux_server -x tmp -p password -d /var/tmp -t /dev/rmt0 -r
```

You should get an output similar to Example 6-8.

Example 6-8 Restoring backed up share files through the smbtar command

```
server    is redhat
share     is tmp\/var/tmp
tar args  is
tape      is /dev/rmt0
blocksize is
added interface ip=10.3.4.32 bcast=10.3.5.255 nmask=255.255.254.0
tarmode is now full, system, hidden, noreset, verbose
Domain=[MYGROUP] OS=[UNIX] Server=[Samba 2.0.10]
restore tar file \WordDocument.doc of size 34816 bytes
restore tar file \WordDocumentWordDocument.sxw of size 12577 bytes
restore tar file \WordDocument1.doc of size 41472 bytes
restore tar file \samba-2.0.10-2.i386.rpm of size 2595888 bytes
restore tar file \samba-client-2.0.10-2.i386.rpm of size 776045 bytes
restore tar file \samba-common-2.0.10-2.i386.rpm of size 968670 bytes
restore tar file \samba-swat-2.0.10-2.i386.rpm of size 3273161 bytes
tar: restored 7 files and directories
```

The **smbtar** command can also be used to do a backup on the same system.

7

NFS

This chapter introduces the Sun Network File System (NFS) as one of the ways to share files between AIX and Linux systems. Using NFS, it is possible to mount a disk partition on a remote machine and access it as if it were on a local hard drive.

The topics covered in this chapter are:

- What NFS is
- Installing NFS on AIX
- Installing NFS on Linux
- Configuring an NFS server on AIX and an NFS client on Linux
  - Configuring an NFS server on AIX
  - Configuring an NFS client on Linux
- Configuring an NFS server on Linux and an NFS client on AIX
  - Configuring an NFS server on Linux
  - Configuring an NFS client on AIX
- Other NFS topics
  - NFS automount
  - User and group ID mappings
  - Access Control Lists
  - NFS locking

7.1 What NFS is

NFS was designed by Sun Microsystems. NFS is a method of accessing disk or file space on other systems transparently, for example, accessing remote objects as if they resided on a local hard disk. The NFS protocol is designed to be portable across different machines, operating systems, network architectures, and transport protocols. By using NFS, users and administrators can distribute data files across AIX and Linux systems by storing the data on either AIX or Linux.

NFS uses Remote Procedure Call (RPC) to provide a procedure-oriented interface to remote services. For detailed information on the NFS protocols and procedures, refer to RFC1094, available at:

http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc1094.html

Read the AIX Documentation on NFS for more details on NFS on AIX.

Linux NFS-HOWTO also provides very relevant and important information. The Linux NFS HOWTO is available at:

http://nfs.sourceforge.net/nfs-howto/index.html

These documents also discuss the security issues for NFS. We recommend you read these documents before installing NFS.

Recent NFS implementations support both the NFS version 3 and version 2. One of the major differences is that NFS Version 3 may use TCP as a transport protocol, thus better supporting WANs than NFS Version 2, which is limited to UDP. NFS Version 3 may transfer data in chunks larger than 8 KB, which is important for some new network technologies, such as gigabit Ethernet. NFS Version 2 cannot be used if files are larger than 2 GB (a 32-bit offset limitation). NFS Version 3 keeps file offsets as 64-bit values. Support for NFS Version 3 in the Linux kernel has only recently been included and NFS over TCP is still considered experimental. NFS Version 4 has been finalized as a protocol only recently, and no implementations are considered production ready.

7.2 Installing NFS on AIX

The NFS client is installed by default on AIX. The NFS server installation on AIX using SMIT is done as follows:

1. Start the SMIT interface and go to the **Install and Update Software** screen by choosing **Software Installation and Maintenance**. You should see a screen similar to Example 7-1 on page 145.

Example 7-1 The Install and Update Software screen

```
                        Install and Update Software

Move cursor to desired item and press Enter.

  Install Software
  Update Installed Software to Latest Level (Update All)
  Install Software Bundle
  Update Software by Fix (APAR)
  Install and Update from ALL Available Software


F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

2. Choose the **Install Software** option. After you type in the appropriate **Input Device / directory**, select the **F4** option to list all the software that can be installed. Refer to Example 7-2 and Example 7-3 on page 146.

Example 7-2 Installing software through SMIT

```
                            Install Software

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                      [Entry Fields]
* INPUT device / directory for software               /aix_source
* SOFTWARE to install                                [_all_latest]          +
  PREVIEW only? (install operation will NOT occur)    no                   +
  COMMIT software updates?                            yes                  +
  SAVE replaced files?                                no                   +
  AUTOMATICALLY install requisite software?           yes                  +
  EXTEND file systems if space needed?                yes                  +
  OVERWRITE same or newer versions?                   no                   +
  VERIFY install and check file sizes?                no                   +
  Include corresponding LANGUAGE filesets?            yes                  +
  DETAILED output?                                    no                   +
  Process multiple volumes?                           yes                  +
  ACCEPT new license agreements?                      no                   +
  Preview new LICENSE agreements?                     no                   +

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

3. Select the **Network File System Server** and press Enter.

Example 7-3 Installing NFS server from SMIT

```
                                Install Software

Ty+-------------------------------------------------------------------------+
Pr|                            SOFTWARE to install                          |
  |                                                                         |
  | Move cursor to desired item and press F7. Use arrow keys to scroll.     |
* |     ONE OR MORE items can be selected.                                  |
* | Press Enter AFTER making all selections.                                |
  |                                                                         |
  | [MORE...467]                                                            |
  |    + 5.2.0.0  IP Security WebSM                                         |
  |    + 5.2.0.0  IPv6 Mobility                                             |
  |    @ 5.2.0.0  Network Computing System 1.5.1                            |
  |    @ 5.2.0.0  Network File System Client                                |
  |    + 5.2.0.0  Network File System Development Toolkit                   |
  |    + 5.2.0.0  Network File System Server                                |
  |    @ 5.2.0.0  Network Information Service Client                        |
  |    + 5.2.0.0  Network Information Service Server                        |
  | [MORE...1103]                                                           |
  |                                                                         |
  | F1=Help                 F2=Refresh              F3=Cancel               |
F1| F7=Select               F8=Image                F10=Exit                |
F5| Enter=Do                /=Find                  n=Find Next             |
F9+-------------------------------------------------------------------------+
```

This installs the NFS server. AIX can act as either an NFS server or client or both as a server and as an NFS client for another NFS server at the same time.

7.3 Installing NFS on Linux

You can install NFS from the NFS Red Hat Package Manager (RPM) packages accompanying your Linux distribution or you can download the RPMs, depending upon the hardware of your Linux system, from the following Web site:

http://speakeasy.rpmfind.net/linux/rpm2html/search.php?query=NFS

After you download the appropriate RPM or after locating the RPM in your Linux distribution CD, install it using the command:

```
# rpm -i /var/tmp/nfs-utils-1.0.1-2.i386.rpm
```

Run **man rpm** for an explanation about RPM. The RPM is stored in the /var/tmp directory in this case. Change /var/tmp to the directory where you stored the NFS RPM.

This installs the NFS server as well as the NFS client.

You can check if NFS is installed on your system by running:

```
# rpm -qa | grep nfs
nfs-utils-1.0.1-2
```

If you get a similar output to the one above, then NFS is installed.

7.4 Configuring an NFS server on AIX and an NFS client on Linux

In this section, we describe a scenario where AIX has the NFS server and Linux has the NFS client.

7.4.1 Configuring an NFS server on AIX

You can configure the server by using AIX's SMIT utility and doing the following steps:

1. To go directly to the NFS server configuration section of SMIT, type the following SMIT fast path:

   ```
   # smit nfs
   ```

   You should see a screen similar to Example 7-4.

Example 7-4 Selecting NFS to configure on AIX

```
                                    NFS

Move cursor to desired item and press Enter.

  Configure TCP/IP (If Not Already Configured)
  Network File System (NFS)
  Network Information Service (NIS)
  Configure Secure NFS & NIS



F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

2. Select **Network File System (NFS)** to get the screen shown in Example 7-5 on page 148.

Example 7-5 NFS options from SMIT

```
                         Network File System (NFS)

Move cursor to desired item and press Enter.

  Configure NFS on This System
  Add a Directory to Exports List
  Change / Show Attributes of an Exported Directory
  Remove a Directory from Exports List
  Add a File System for Mounting
  Change / Show Attributes of an NFS File System
  Remove an NFS File System



F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

3. Then select **Configure NFS on This System** to get a screen similar to Example 7-6. This screen allows you to control the NFS services and configure them according to your requirements. Most of the options are self-explanatory.

Example 7-6 Configuring NFS on AIX

```
                       Configure NFS on This System

Move cursor to desired item and press Enter.

  Start NFS
  Stop NFS
  Change Number of nfsd, biod, lockd Daemons
  Start Automounter
  Stop Automounter



F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

4. You have the option of starting the NFS services at boot time, as shown in Figure 7-7 on page 149.

Example 7-7 Option of starting NFS at boot time

```
                                Start NFS

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                        [Entry Fields]
* START NFS now, on system restart or both             both                  +



F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

5. You can also change the number of NFS daemons with **Change Number of nfsd, biod, lockd Daemons** in Example 7-6. Depending on the workload imposed by NFS clients, the number of nfsd processes may be modified (the default number is 8). The biod processes are handling requests on NFS clients and they may also influence performance. The NFS locking is supported by the lockd servers, which are running on both clients and servers. In combination with the statd status monitoring process, the lockd servers support recovery in case of a host crash. For more information, see the nfsd(8), biod(8), lockd(8), and statd(8) man pages. The Auotmounter options can be used to start the automountd daemon with appropriate parameters. This is explained in 7.6.1, “NFS automount” on page 156.
6. In order to start exporting the system directories and files, execute the command:

   ```
   # smit mknfsexp
   ```

   You will see the screen shown in Example 7-8.

Example 7-8 Adding a directory to exports list on AIX

```
                  Add a Directory to Exports List

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                        [Entry Fields]
* PATHNAME of directory to export                  [/usr/home]
/
* MODE to export directory                           read-write            +
  HOSTS & NETGROUPS allowed client access            []
  Anonymous UID                                      [-2]
  HOSTS allowed root access                          []
  HOSTNAME list. If exported read-mostly             []
```

```
  Use SECURE option?                                  no                    +
  Public filesystem?                                  no                    +
* EXPORT directory now, system restart or both        both                  +
  PATHNAME of alternate Exports file                 []


F1=Help          F2=Refresh       F3=Cancel        F4=List
F5=Reset         F6=Command       F7=Edit          F8=Image
F9=Shell         F10=Exit          Enter=Do
```

You can edit the values in the field by pressing **F7**. The required entries are:

- PATHNAME of directory to export
  - In our example, we are setting it as /usr/home.
- MODE to export directory
  - The default mode is read-write, which is the same for our example. You can limit the access by using a client list.
- EXPORT directory now, system restart or both
  - Again the default **both** is applicable to our example.

  The field-names are self-explanatory. Fill in these parameters and press Enter. SMIT will create or update the /etc/exports file and then run the **exportfs -a** command. You should see a screen similar to Example 7-9.

Example 7-9 Starting the NFS server

```
                              COMMAND STATUS

Command: OK            stdout: yes           stderr: no

Before command completion, additional instructions may appear below.

/usr/home
Exported /usr/home


F1=Help          F2=Refresh       F3=Cancel        F6=Command
F8=Image         F9=Shell         F10=Exit          /=Find
n=Find Next
```

Note: You can also configure the server by executing the **src** commands. But before starting the NFS daemons, you need to create /etc/exports file. This can be done with the **touch /etc/exports** command. This would not export any file systems from AIX as of yet.

Using the **src** command, you can start all the nfs related daemons:

```
# startsrc -g nfs
```

You can also start the NFS related daemons separately by using the appropriate **startsrc** command for biod, rpc.lockd, rpc.statd, and rpc.mountd.

7.4.2 Configuring NFS client on Linux

To set up Linux as a NFS client, do the following steps:

1. Start the portmap, statd, and lockd daemons.

   Most of the time, these services are started at boot time. If you are unable to do that, then you need to manually start these services with the appropriate commands. Please refer to the Linux NFS HOWTO for further information, found at:

   http://nfs.sourceforge.net/nfs-howto/index.html

2. Mounting the remote file system.

   You should be able to mount the remote file system in a similar way to how you mount a local hard drive or floppy drive. For example:

   ```
   # mount aix_host:/usr/home /homes
   ```

   This mounts /usr/home from the aix_host machine onto the /homes directory of the local Linux machine.

   Note: In this example, the aix_host entry should be resolvable to the IP address. This can be done by simply making an appropriate entry in the /etc/hosts file. You can also use the IP address directly in the **mount** command. Refer to the manual pages for more information on the **mount** command.

   You can verify the mounted directories by executing the **mount** command.

   ```
   # mount
   ```

   Among other things, you should see a line similar to:

   ```
   aix_host:/usr/home on /homes type nfs (rw,addr=10.3.4.32)
   ```

This means that /usr/home from aix_host is mounted on /homes on the local system with a read-write attribute and the IP address of aix_host is 10.3.4.32.

3. We can test the NFS mounted directory now. If, for example, a user Bob with UID=512 both on Linux and AIX owns the directory /usr/home on AIX, then after exporting /usr/home, user Bob is also the owner of /homes on Linux and can modify or create more sub-directories and files under /homes on Linux, as shown in the following lines:

```
[bob@localhost bob]$ cd /homes
[bob@localhost homes]$ ls
temp
[bob@localhost homes]$ touch file1 file2
[bob@localhost homes]$ ls
file  file2  temp
[bob@localhost homes]$ rm -rf temp
[bob@localhost homes]$ ls
file  file2
```

These commands show creating files and removing directories on the /homes directory by user Bob, who has same user ID number on AIX and Linux.

Mounting the NFS file systems at boot time

In order to make the NFS mount changes permanent, you need to make the appropriate entry in the /etc/fstab file. The entry is in the form:

```
 device              mountpoint fs-type   options     dump fsckorder
aix_host:/usr/home  /homes      nfs       rw          0    0
```

This would mount /usr/home from aix_host on /homes after system restart.

7.5 Configuring an NFS server on Linux and an NFS client on AIX

In this section, we describe a scenario of an NFS server on Linux and Linux files and directories mounted on an AIX system through NFS.

7.5.1 Configuring an NFS server on Linux

As described in the Linux NFS HOWTO, we can configure the NFS server in two steps:

1. Setting up the configuration files
2. Starting the NFS services

Setting up the configuration files

To configure the NFS server, we need to edit just the /etc/exports file. Access to the portmapper server may be controlled using the tcpwrapper. NFS clients should be entered in the /etc/hosts.allow file; otherwise, access to the portmapper port should be prohibited. Some implementations of other NFS specific servers, such as statd, include support for the tcpwrapper. That is typically the case with the Linux NFS implementation.

The /etc/exports file contains entries detailing which directories or files can be shared and how they are shared. The format is as follows:

```
directory machine1(option11,option12) machine2(option21,option22)
```

For example:

```
/home/bob aix_host(rw,insecure)
```

This exports the directory and sub-directories under /home/bob to a system called aix_host. These options specify the read-write access (rw) and that NFS clients connecting from ports higher than 1023 are allowed access. Ports 1-1023 are considered secure because only the root user can bind to them.

> **Note:** The insecure option is required because Linux, by default, requires the NFS mount to use ports below 1024 and AIX by default uses ports above 1024. Instead of mentioning the insecure command option on NFS server, you can also solve this problem by executing the **nfso** command on AIX (NFS client):
>
> ```
> # nfso -o nfs_use_reserved_ports=1
> ```

You can find out about all the options by running **man 5 exports** on Linux and **man nfso** on AIX.

If you change the /etc/exports file later, then you need to export the new entries again. This can be done by running the **exportfs -ra** command to force NFS to reread the /etc/exports file.

> **Note:** The /etc/exports file is very sensitive to spaces. Be careful not to add any extra spaces.

Starting NFS services

After editing the /etc/exports file, the following NFS services can be started:

- portmap
- mountd
- nfsd

- lockd
- statd

The first service that should be started is portmap. All these services are generally started automatically once you start the main NFS daemon from the /etc/rc.d/init.d or /etc/rc.d/ directory, depending upon your Linux distribution. For example, on Red Hat Linux, you can start the NFS daemons by executing **/etc/rc.d/init.d/nfs start**.

Most of the services are part of the standard NFS utilities, but if your distribution does not include them, then you should ensure that they are configured and started in the above order.

Once you start all the services, you can verify that NFS is running by executing the **rpcinfo -p** command.

You should see a screen similar to Example 7-10. You should at least see the lines with portmapper, nfs, and mountd. If not, then repeat the configuration steps and verify the installation process. The Troubleshooting section of the Linux NFS HOWTO should also be referred to:

http://nfs.sourceforge.net/nfs-howto/index.html

Example 7-10 Verifying NFS is running

```
program vers proto   port
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
    100011    1   udp    878  rquotad
    100011    2   udp    878  rquotad
    100011    1   tcp    881  rquotad
    100011    2   tcp    881  rquotad
    100003    2   udp   2049  nfs
    100003    3   udp   2049  nfs
    100021    1   udp  32780  nlockmgr
    100021    3   udp  32780  nlockmgr
    100021    4   udp  32780  nlockmgr
    100005    1   udp  32781  mountd
    100005    1   tcp  32780  mountd
    100005    2   udp  32781  mountd
    100005    2   tcp  32780  mountd
    100005    3   udp  32781  mountd
    100005    3   tcp  32780  mountd
```

The Linux NFS server is now exporting the /home/bob directory. We need to configure the NFS client on AIX.

7.5.2 Configuring the NFS client on AIX

You can start making the mounts from the NFS server by doing the following:

1. Start SMIT by running:

   ```
   # smit mknfsmnt
   ```

2. The screen in Example 7-11 appears. The required entries are denoted by (*). Most of the default values should be valid for your configuration. In case you need more explanation, press **F1** for that particular parameter.

   For the **MOUNT now, add entry to /etc/filesystems or both** option, the default is `now`, which implies that the mount is in effect only until the system reboots or until the **`umount`** command for that particular directory is run. In order to make the change permanent and make an entry in /etc/filesystems, you can select the `both` option.

Example 7-11 Adding a File System for Mounting on AIX using SMIT

```
                        Add a File System for Mounting

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

[TOP]                                                   [Entry Fields]
* PATHNAME of mount point                              [/bob_home]
* PATHNAME of remote directory                         [/home/bob]
* HOST where remote directory resides                  [10.3.5.23]
  Mount type NAME                                      []
* Use SECURE mount option?                              no                      +
* MOUNT now, add entry to /etc/filesystems or both?    now                     +
* /etc/filesystems entry will mount the directory      no                      +
   on system RESTART.
* MODE for this NFS file system                        read-write              +
* ATTEMPT mount in foreground or background            background              +
  NUMBER of times to attempt mount                     []                       #
  Buffer SIZE for read                                 []                       #
  Buffer SIZE for writes                               []                       #
  NFS TIMEOUT. In tenths of a second                   []                       #
[MORE...25]

F1=Help          F2=Refresh       F3=Cancel        F4=List
F5=Reset         F6=Command       F7=Edit          F8=Image
F9=Shell         F10=Exit          Enter=Do
```

After making your entries, press Enter. You will see OK in the command status screen. Alternately, the operation performed in the SMIT menu can be done with the following one line command:

```
# mount 10.3.5.23:/home/bob /bob_home
```

3. To verify the mounted directory, we show you an example of user bob with the same user ID (numerical) on AIX and Linux.

   a. Execute the **mount** command. You should get an output similar to the following:

   ```
   # mount
   10.3.5.23 /home/bob /bob_home       nfs3   Nov 05 10:01 bg,hard,intr
   ```

   b. Log in as user bob and run as test as follows:

   ```
   $ cd /bob_home/
   $ ls
   test
   $ touch file1 file 2
   $ ls
   file1  file2  test
   $ rm -rf test
   $ ls
   file1  file2
   ```

7.6 Other NFS topics

The topics discussed in this section are applicable for any NFS client/server model, but are nevertheless important enough to be covered under the AIX and Linux interoperability.

- NFS Automount
- User and group ID mapping
- Access Control Lists
- NFS locking

7.6.1 NFS automount

The **automount** command is used to administer the AutoFS file system. It is used to mount NFS directories only when a user requests access to a file residing on the server.

Please refer to the *AIX 5L Version 5.2 Commands Reference, Volume 1* for more details on the automount daemon.

The steps to configure automount in a general NFS server-client environment are:

1. Configure the NFS server and client if they are not already configured.
2. Create the master map file on the NFS client.

The master map file is generally created as /etc/auto.master and has the following format:

```
Directorypath    Automountmapname
```

For example,

```
/home/home      /etc/auto.home
```

This indicates that you should use the /etc/auto.home map to access /home/home directory.

3. Create the map files on the NFS client.

   The map files are generally in the form auto.mapname, for example, /etc/auto.home or /etc/auto.direct. The format for the map files is:

   ```
   subdirectory   [-mount options] server:/directory
   ```

   Where subdirectory is under the mount directory. For example, if /etc/auto.home contains an entry similar to:

   ```
   borg  -rw,hard,intr   nfs_server:/home/borg
   ```

   it means that /home/borg from nfs_server is to be automounted on /home/home/borg.

4. If Network Information Service (NIS) is to be used for configuring the automount maps, integrate the maps with NIS.

   You have to set up NIS on AIX and Linux. The NIS server can be either on Linux or AIX. Note that NIS server or client is independent of the NFS server and client and is only used for maps.

   Refer to the “NIS Automount” section in the *AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide*, for details about NIS mappings. Refer the Linux NIS HOWTO for information on setting up NIS on Linux, found at:

   http://www.linux-nis.org/nis-howto/

5. Start automounter from the NFS client.

   On AIX, automount is started by running **/usr/sbin/automount** or from SMIT by running the **smit mkautomnt** fast path. Run **man automount** for more details.

> **Note:** On AIX 5L Version 5.2.0, automount uses the TCP protocol by default. The NFS server on Linux, on the other hand, is listening on a UDP port. Hence, when using the **automount** command on AIX 5L Version 5.2.0, the automount map files should have proto=udp in the mount options. For example, the /etc/auto.home file on AIX (NFS client) should be:
>
> ```
> borg -rw,hard,intr,proto=udp 10.3.5.23:/home/borg
> ```

On Linux, automount is started by running **/usr/sbin/automount** or by the /etc/init.d/autofs script. This may be different, depending on your Linux distribution. For example, in Red Hat Linux, execute **/etc/rc.d/init.d/autofs start**.

Run **man autofs** on AIX and **man automount** on Linux for more details.

6. After starting the automounter, you can test if automount is working by just changing the directory to the desired directory on the NFS client. For example:

   ```
   $ cd /home/home/borg
   ```

   You should now be able to access the /home/borg directory, which is the directory exported from the NFS server to the NFS client at mount point /home/home/borg. This mount is done without explicitly executing the **mount** command.

Here is an example scenario of an NFS server on AIX and mounting the AIX directory on a Linux system using automount:

1. Configure the NFS server on AIX and NFS client on Linux.
2. On the Linux system, create the /etc/auto.master file with following contents:

   ```
   /home/home /etc/auto.home
   ```

3. Create a new file or add the following contents to the /etc/auto.home file:

   ```
   borg -rw,hard,intr  aix_host:/home/borg
   ```

4. Ensure that the NFS server on AIX is exporting the /home/borg directory with the proper permissions.
5. Start the autofs script on Linux (depending upon your Linux distribution). For example, on Red Hat Linux, the autofs script is at /etc/init.d/autofs:

   ```
   # /etc/init.d/autofs start
   Starting automount:                                        [  OK  ]
   ```

6. When you issue the **mount** command, you should get the following line in the output, among other general mount information:

   ```
   # mount
   automount(pid27470) on /home/home type autofs
   (rw,fd=5,pgrp=27470,minproto=2,maxproto=3)
   ```

7. Change the working directory to /home/home/borg:

   ```
   # cd /home/home/borg
   ```

8. Issue the **mount** command again to check the output. It should contain lines similar to the following:

   ```
   automount(pid27470) on /mnt type autofs
   (rw,fd=5,pgrp=27470,minproto=2,maxproto=3)
   ```

```
aix_host:/home/borg on /home/home/borg type nfs
(rw,hard,intr,addr=10.3.4.32)
```

9. Now you can access the /home/home/borg directory on Linux as a local directory.

7.6.2 User and group ID mapping

The user ID and group ID mappings should be the same on both the NFS client and the NFS server. Every NFS request carries user credentials in the form of the UID and a list of GIDs. In case this user has a different UID on the NFS server, he/she may not be able to access his/her files. This problem may be solved either by sharing the user information from the /etc/passwd and /etc/group maps or by running the dynamic ugidd mapping daemon on NFS clients. The former solution is preferable.

For example, if a user named Martha has UID=510 on the Linux system and that Linux system (NFS client) has /usr/local/ mounted from an AIX system (NFS server), and Martha creates a new file, for example, new_file in /usr/local, this new file will be owned by the user with UID=510 on both AIX and Linux. So any user with UID=510 on AIX would be treated as the owner of the /usr/local/new_file file.

The group ID mapping is handled in a similar manner.

Under normal circumstances, the NFS server maps the UID 0 (root) to the UID of the nobody user before performing access checks for a client. This process prevents gaining root privileges on remote file systems. You can use the squash_root option (A Linux directive; the "anon=0" option can be used on AIX) in the NFS server's /etc/exports file to avoid this situation, but its use is not recommended for security reasons.

7.6.3 Access control lists

The NFS ACL support is typically limited to cases when both the client and the server run the same NFS implementation. The NFS Version 4 standardizes the use of ACLs, but there are still no usable implementations. The ACLs on NFS Version 2 and NFS Version 3 are supported with vendor specific extensions.

NFS no longer uses access control lists by default in AIX 5L Version 5.2 (though it is still the default on AIX Version 4.3). If you need to use ACLs with NFS, then you have to use the acl option with the **mount** command as follows:

```
# mount -o acl
```

An RPC program handles the ACLs between clients and servers. Using ACLs can cause unexpected behavior, as the NFS clients cannot see the ACLs. For example, a directory or file's permission bits may show write access to the client, but if the ACL associated with that file does not permit write, then the client would not be able to write to that file. The permissions on a server are set according to the ACLs, causing a client to get permissions error.

Even though AIX supports ACLs over NFS, the general ACL support on Linux is very recent and still considered experimental. When implemented partly or not well understood, use of ACL and NFS may result in a security risk. For more information on the Linux ACL development, refer to the following Web site:

http://acl.bestbits.at/

7.6.4 NFS locking

Locking is a mechanism by which the data integrity of a shared file is maintained. By applying locking, processes can safely modify and access shared files. Most of the time, the user is not required to know the details of locking and how is it implemented. But in an NFS environment, where more than one machine can access the same file, the problem of recovering from a server or client crash is handled in a different way.

The problems related to locking in a network context are:

- If the client crashes, the server must release the locks.
- If the server crashes, clients should be able to recover their locks.

NFS is a *stateless* protocol and does not deal with locking itself. The recovery from server or client crash is handled using two daemons:

- lockd
- statd

The lock daemon cooperates with the status daemon to ensure that it is notified of relevant machine crashes. Figure 7-1 on page 161 shows the locking mechanism used by NFS.

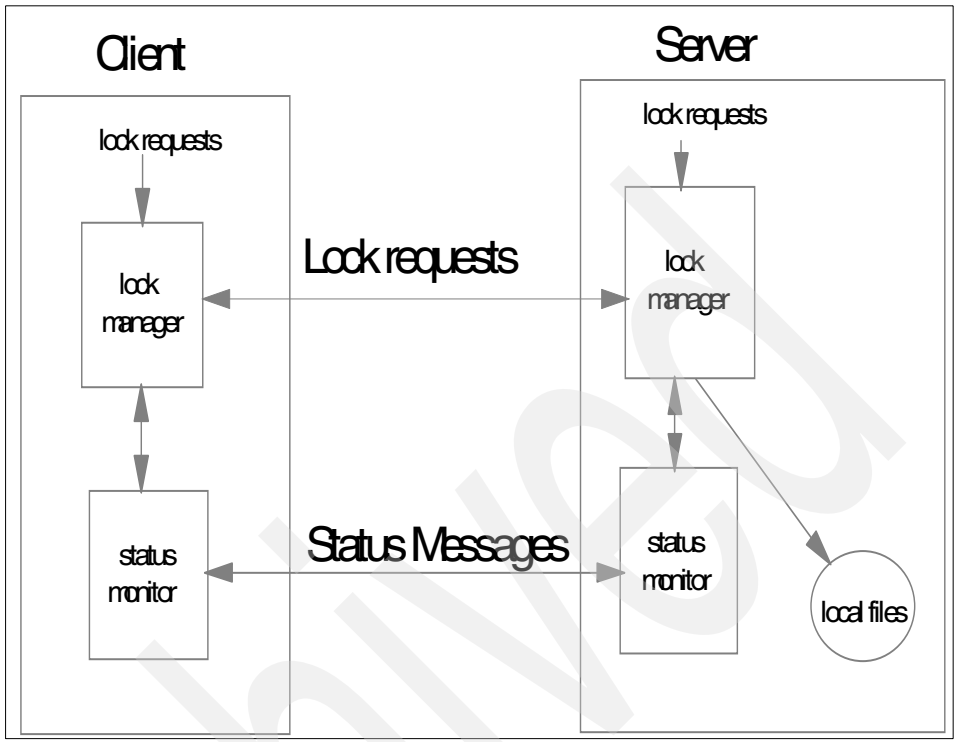

Figure 7-1 NFS locking architecture

8

File systems and data archiving

In this chapter, we discuss file systems and backup and archiving tools on AIX and Linux. In particular, we examine the compatibility of archivers, as this sometimes may be an issue for data transfers between different computer architectures.

Though the Linux journaling file systems market offers many different file systems, we concentrate on the IBM JFS, as it is available on both AIX and Linux. We also cover file Access Control Lists (ACL) in this context.

An important new feature delivered with AIX 5L Version 5.2 is the UDF file system support. AIX computers equipped with DVD RAM devices may use UDF for backups or data archiving.

8.1 Journaled File System (JFS)

The Journaled File System is a file system that provides improved structural consistency and recoverability and much faster restart times that non-journaled file systems, such as DOS FAT, traditional UNIX file systems, and ext2. Using database journaling techniques JFS can restore a file system to a consistent state in a matter of seconds, versus hours or days with non-journaled file systems when the amount of data is really big.

In this section, we will cover how to convert an existent ext3 file system based Linux system to one running entirely on IBM Journaled File System (JFS). For more information about JFS, please visit the JFS for Linux Web site at:

http://www-124.ibm.com/jfs

> **Note:** ext3 is a journaled file system, but the intention here is to show how you can convert a given file system to JFS.

8.1.1 Converting a existing ext3 file system to JFS

We are going to convert an ext3 partition to a JFS partition in a machine running Red Hat Linux Version 8.0. During the installation of Red Hat Linux, we cannot choose a JFS partition, so in this section, we will suppose that the partition was formatted, using ext3, for installation. There is no way to directly convert an ext3 partition to JFS. All that we can do is create a new partition and then copy the ext3 partition's contents to the JFS partition.

The Red Hat Linux Version 8.0 distribution is shipped with JFS, but that does not mean that we will not have to recompile the kernel to install JFS. He have to recompile it because in Red Hat, the JFS is as a *module* in the kernel and we have to change it to a *built-in* option. In other words, if you do not recompile the kernel, you will not be able to boot from a JFS partition.

To configure the kernel, use the **`make config`**, **`make menuconfig`**, or **`make xconfig`** command. In the section labeled File Systems, change the option **JFS filesystem support** from `M` (module) to * (built-in) and press the space bar. Configure the rest of kernel as needed for your system. After that, save the configuration and then build the kernel issuing the commands **`make dep`**, **`make clean`**, **`make bzImage`**, and, if necessary, **`make modules`** and **`make modules_install`**, as shown in Example 8-1 on page 165.

Example 8-1 Recompiling the kernel

```
# cd /usr/src
# ln -s linux-2.4.18-14 linux
# cd linux
# make menuconfig
# make dep
# make clean
# make bzImage
# cp arch/i386/boot/bzImage /boot/vmlinuz-2.4.18-14-jfs
```

Now we have to change the lilo.conf file and add a new entry. The lilo.conf file is showed in Example 8-2.

Example 8-2 lilo.conf file

```
prompt
timeout=50
default=jfs
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
message=/boot/message
linear

image=/boot/vmlinuz-2.4.18-14smp
       label=linux
       initrd=/boot/initrd-2.4.18-14smp.img
       read-only
       append=”root=LABEL=/”

image=/boot/vmlinuz-2.4.18-14
       label=linux-up
       initrd=/boot/initrd-2.4.18-14.img
       read-only
       append=”root=LABEL=/”

image=/boot/vmlinuz-2.4.18-14-jfs
       label=jfs
       read-only
       root=/dev/hda1
```

Run the **lilo** command to activate the new configuration:

```
# lilo
Added linux
Added linux-up
Added jfs *
```

Reboot the system. Now we have JFS support built-in into the kernel and we can create the JFS partition.

Let us begin checking the current file system that we are using, issuing the **mount** command, as showed in Example 8-3.

Example 8-3 Verifying the current file system

```
# mount
/dev/hda1 on / type ext3 (rw)
none on /proc type proc (rw)
usbdevfs on /proc/bus/usb type usbdevfs (rw)
none on /dev/pts type devpts (rw,gid=5,mode=620)
none on /dev/shm type tmpfs (rw)
```

Now we can create the JFS file system. This can be created in a separate disk or in the same disk; in our case, we will create this in the same disk. Make sure that you have a Linux partition (type 83) created on the destination partition and issue the **mkfs** command, as show in Example 8-4. In our case, we will be using the /dev/hda2 partition.

Example 8-4 Creating the JFS file system

```
# mkfs -t jfs /dev/hda2
mkfs.jfs version 1.0.17, 02-Apr-2002
Warning!  All data on device /dev/hda3 will be lost!

Continue? (Y/N) y
   /

Format completed successfully.

5927985 kilobytes total disk space.
```

Now that we have created a JFS partition, it is time to mount this partition and copy the root file system on to it. We have to create a mount point for the new JFS partition and mount it. The commands to mount the file system and copy the files is shown in Example 8-5.

Example 8-5 Copying the file system

```
# mkdir /jfs
# mount -t jfs /dev/hda2 /jfs
# cd /
# dump -0f - / | (cd /jfs; restore -rf -)
  DUMP: Date of this level 0 dump: Thu Nov 21 10:43:04 2002
  DUMP: Dumping /dev/hda1 (/) to standard output
  DUMP: Added inode 8 to exclude list (journal inode)
```

```
  DUMP: Added inode 7 to exclude list (resize inode)
  DUMP: Label: /
  DUMP: mapping (Pass I) [regular files]
  DUMP: mapping (Pass II) [directories]
  DUMP: estimated 803317 tape blocks.
  DUMP: Volume 1 started with block 1 at: Thu Nov 21 10:43:09 2002
  DUMP: dumping (Pass III) [directories]
  DUMP: dumping (Pass IV) [regular files]
./tmp/rstdir1037896984: (inode 213246) not found on tape
./tmp/rstmode1037896984: (inode 213340) not found on tape
  DUMP: 63.12% done at 1690 kB/s, finished in 0:02
  DUMP: Volume 1 completed at: Thu Nov 21 10:52:10 2002
  DUMP: Volume 1 866550 tape blocks (846.24MB)
  DUMP: Volume 1 took 0:09:01
  DUMP: Volume 1 transfer rate: 1601 kB/s
  DUMP: 866550 tape blocks (846.24MB)
  DUMP: finished in 541 seconds, throughput 1601 kBytes/sec
  DUMP: Date of this level 0 dump: Thu Nov 21 10:43:04 2002
  DUMP: Date this dump completed:  Thu Nov 21 10:52:10 2002
  DUMP: Average transfer rate: 1601 kB/s
  DUMP: DUMP IS
DONE
```

It will take a few minutes.

The next step is change the fstab and lilo.conf files. Remember that the fstab file that we want to use is under the /jfs directory, so the full path for this file will be /jfs/etc/fstab. In Example 8-6, we show the /jfs/etc/fstab before any change. In Example 8-7, we show the changes necessary.

Example 8-6 Original /jfs/etc/jfs file

```
LABEL=/          /                ext3    defaults        1 1
none             /dev/pts         devpts  gid=5,mode=620  0 0
none             /proc            proc    defaults        0 0
none             /dev/shm         tmpfs   defaults        0 0
/dev/hda2        swap             swap    defaults        0 0
/dev/cdrom       /mnt/cdrom       iso9660 noauto,owner,kudzu,ro 0 0
/dev/fd0         /mnt/floppy      auto    noauto,owner,kudzu 0 0
```

Example 8-7 Modified /jfs/etc/fstab file

```
/dev/hda3        /                jfs     defaults        1 1
none             /dev/pts         devpts  gid=5,mode=620  0 0
none             /proc            proc    defaults        0 0
none             /dev/shm         tmpfs   defaults        0 0
/dev/hda2        swap             swap    defaults        0 0
/dev/cdrom       /mnt/cdrom       iso9660 noauto,owner,kudzu,ro 0 0
```

```
/dev/fd0            /mnt/floppy             auto    noauto,owner,kudzu 0 0
```

At this moment, we can umount the /jfs directory by issuing the following command:

```
# umount /jfs
```

In Example 8-2 on page 165, we show /etc/lilo.conf, which must be modified to point to the new root partition. In Example 8-8, we show the modification that you should do in this file.

Example 8-8 Modified /etc/lilo.conf

```
prompt
timeout=50
default=jfs2
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
message=/boot/message
linear

image=/boot/vmlinuz-2.4.18-14smp
       label=linux
       initrd=/boot/initrd-2.4.18-14smp.img
       read-only
       append="root=LABEL=/"

image=/boot/vmlinuz-2.4.18-14
       label=linux-up
       initrd=/boot/initrd-2.4.18-14.img
       read-only
       append="root=LABEL=/"

image=/boot/vmlinuz-2.4.18-14-jfs
       label=jfs
       read-only
       root=/dev/hda1

image=/boot/vmlinuz-2.4.18-14-jfs
       label=jfs2
       read-only
       root=/dev/hda2
```

After modifying the lilo.conf file, run the **lilo** command:

```
# lilo
Added linux
Added linux-up
Added jfs
Added jfs2 *
```

After that, you can reboot you system. After reboot, you can verify if the system is now completely running on JFS by issuing the **mount** command, as shown in Example 8-9.

Example 8-9 Verifying the file system

```
# mount
/dev/hda2 on / type jfs (rw)
none on /proc type proc (rw)
usbdevfs on /proc/bus/usb type usbdevfs (rw)
none on /dev/pts type devpts (rw,gid=5,mode=620)
none on /dev/shm type tmpfs (rw)
```

If everything works well and you do not need to go back to your old system, the old root partition can be reformatted as JFS and turned into any directory that you want. If you used a different disk, you can now unplug the original disk and use only the disk where JFS was installed.

8.1.2 Using ACL on JFS

Sometimes, the access control provided by Linux that defines the access to a file or directory, based on the user, group, or others associated to this file or directory, is not enough. For example, if you have some users that have to access a given file and you want to give a different access permission to each one, you cannot do it in a trivial way, so you need to use ACLs.

To use ACLs we need to apply a patch to the kernel. Red Hat Linux Version 8.0 does not ship with ACL support on the kernel. In the document *Red Hat Linux 8.0 Release Notes*, available at:

http://www.redhat.com/docs/manuals/linux/RHL-8.0-Manual/pdf/rhl-relnotes-x86-en-80.pdf

You can find the following special note, which explains why ACLs are not shipped with the current version of Red Hat Linux:

“The ACL support added to the kernel in the first two public beta releases proved to be unstable and cause the kernel to regress in terms of standards compliance. Red Hat has therefore removed that ACL support from the kernel fro Red Hat linux 8.0. Kernel engineers will continue work on improving the ACL support,

which will be available in future release. The attr and acl packages needed to support ACLs are still included to make it easier for users and developers who wish to test ACLs. Red Hat may, at our discretion, provide ACL support for this release of Red Hat Linux by means of an upgrade, if future testing demonstrates that the ACL support has sufficiently improved in quality."

But that does not mean that you cannot install the ACL support to your kernel. The patches for the kernel are available at:

http://acl.bestbits.at

To support ACL on a JFS file system, you have to apply a specific patch, but before that, you should had applied the patch provided by the Web site above. You can find the kernel's patches for a JFS file system at:

http://oss.software.ibm.com/developerworks/patch/?group_id=35

We will show you some commands to set up and verify ACLs on Linux, but understand that at the time of the writing of this redbook, we were unable to set up ACLs on the Red Hat Linux Version 8.0 kernel.

Using the **setfacl** command, we will be able to set the ACL for a file. In Example 8-10, we set read and write permissions for the user jack over the passwd.txt file.

Example 8-10 Setting ACLs for a user

```
# setfacl -m u:jack:rw password.txt
```

On Example 8-11, we set read, write, and execution rights to the group admin in a file named report.sh.

Example 8-11 Setting ACLs for a group

```
# setfacl -m g:admin:rwx report.sh
```

You can check the ACL of a file by issuing the **getfacl** command. In Example 8-12, we check the ACL of the files used in Example 8-10 and Example 8-12.

Example 8-12 Checking the ACLs

```
# getfacl password.txt
# file: password.txt
# owner: john
# group: users
user::r--
user:jack:rw-
group::r--
```

```
mask:rw-
other:---

# getfacl report.sh
# file: report.sh
# owner: bob
# group: operators
user::r-x
group::r-x
group:admin:rwx
mask:rw-
other:---
```

As you can see, ACLs support more fine-grained permissions. If you want to know more about ACLs and how they work, refer to the following Web page:

http://acl.bestbits.at/about-acl.html

8.2 Universal Disk Format (UDF)

Universal Disk Format (UDF) is a newer CD-ROM file system type required for DVD-ROMs. It is a new feature that was introduced in AIX 5L 5.2.0 and is not supported in earlier versions of AIX. UDF is available for DVD-RAM media on AIX. Linux also supports the UDF file system. UDF is very useful for creating backups, as it uses less disk space for creating the backup because it writes the backup directly to the DVD.

The main advantage of UDF is that it allows you to manipulate files directly on the DVD-RAM media. This is very useful for the backup process, as you can directly change user-defined files like bosinst.data, keeping the system backup image intact on the DVD itself. If you wanted to do the same without UDF, then you will have to restore the backup image, add the files, and then back up the files again.

To learn more about the procedure of making an AIX system backup on DVD-RAM, refer to the “DVD-RAM and Universal Disk Format” section of the *AIX 5L Version 5.2 Installation Guide and Reference: Creating System Backups*.

To learn more about UDF support on Linux, refer to the following Web site:

http://www.trylinux.com/projects/udf/

By using the UDF file system you can use your favorite editor or other commands like **cp**, **touch**, **mv**, and so on, to manipulate the files *directly* on the DVD media. There are UDF related commands on AIX, such as **udfcreate**, **udflabel**, and

udfcheck. The command names are self-explanatory. Refer to the man pages of those commands for the details.

In the following example, after mounting the DVD-RAM, we create an UDF file system on a DVD-RAM. We can then take this DVD-RAM and place it on a Linux system and read the data created on AIX.

The steps to do so are as follows:

1. Check if the DVD-RAM drive is installed on your system. You can do this by using SMIT as follows:
   a. Start the SMIT interface and go to the **Devices** section. You will see a screen similar to Example 8-13.

Example 8-13 Installing the DVD-RAM drive using SMIT

```
                                    Devices

Move cursor to desired item and press Enter.

[TOP]
  Install/Configure Devices Added After IPL
  Printer/Plotter
  TTY
  Asynchronous Adapters
  PTY
  Console
  MPIO Management
  Fixed Disk
  Disk Array
  CD ROM Drive
  Read/Write Optical Drive
  Diskette Drive
  Tape Drive
  Communication
  Graphic Displays
[MORE...10]

F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

   b. Go to the **CD ROM Drive** section. You will see a screen similar to Example 8-14 on page 173.

Example 8-14 The CD-ROM section of SMIT

```
                                CD ROM Drive

Move cursor to desired item and press Enter.

  List All Defined CD ROM Drives
  List All Supported CD ROM Drives
  Add a CD ROM Drive
  Change / Show Characteristics of CD ROM Drive
  Remove a CD ROM Drive
  Configure a Defined CD ROM Drive
  Generate Error Report
  Trace a CD ROM Drive



F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

c. Select the **List All Defined CD ROM Drives** option. If you see a screen similar to Example 8-15, then the DVD-RAM drive is already installed. If not, go back to the screen in Example 8-14 and select the **Add a CD ROM drive** option.

Example 8-15 DVD-RAM drive installed

```
COMMAND STATUS

Command: OK            stdout: yes           stderr: no

Before command completion, additional instructions may appear below.

cd0 Available 10-60-00-2,0 SCSI DVD-RAM Drive


F1=Help             F2=Refresh          F3=Cancel           F6=Command
F8=Image            F9=Shell            F10=Exit            /=Find
n=Find Next
```

For just checking the location of the DVD-RAM drive, type the following command:

```
# lsdev -Cccdrom
```

2. After installing the DVD-RAM drive, create an UDF file system on the DVD media with the **udfcreate** command:

```
# udfcreate -d /dev/cd0
```

Where /dev/cd0 is the DVD-RAM drive. Run **man udfcreate** to see all the options.

3. You can also set a label on the DVD media using the **udflabel** command:

```
# udflabel -d /dev/cd0 -l test
```

Where test is the label. Run **man udflabel** to see all the options.

4. After creating the UDF file system, you can now mount the DVD-RAM using the **mount** command:

```
# mount -v udfs /dev/cd0 /mnt/dvdram
```

The /mnt/dvdram directory should already be present before executing the **mount** command.

Execute the **mount** command without any options to see the presently mounted file systems:

```
# mount
node       mounted        mounted over    vfs       date        options
-------- ---------------  ---------------  ------ ------------
---------------
         /dev/hd4         /               jfs2   Nov 05 13:46
rw,log=/dev/hd8
         /dev/hd2         /usr            jfs2   Nov 05 13:46
rw,log=/dev/hd8
         /dev/hd9var      /var            jfs2   Nov 05 13:46
rw,log=/dev/hd8
         /dev/hd3         /tmp            jfs2   Nov 05 13:46
rw,log=/dev/hd8
         /dev/hd1         /home           jfs2   Nov 05 13:49
rw,log=/dev/hd8
         /proc            /proc           procfs Nov 05 13:49 rw
         /dev/hd10opt     /opt            jfs2   Nov 05 13:49
rw,log=/dev/hd8
         /dev/janlv       /janfs          jfs    Nov 05 13:49
rw,log=/dev/loglv00
         /dev/cd0         /mnt/dvd        udfs   Nov 22 16:12
```

The last line shows that /dev/cd0 has been mounted on the /mnt/dvdram directory.

5. Go to the /mnt/dvdram directory and copy, create, or move files and directories:

```
# cd /mnt/dvdram/
# touch file1 file 2
# ls
file1 file2
# mkdir test_dir
# ls
file1 file2 test_dir
```

```
# cp /etc/qconfig .
# ls
file1 file2 test_dir qconfig
```

This shows that by using the UDF file system, we can manipulate files directly on the DVD media.

6. Take this DVD and place it on the DVD Drive of a Linux system. Linux kernel Versions 2.3.17 and higher already have the UDF driver installed. If your kernel does not support the UDF file system, you can refer to the following Web site and download and install UDF module for your kernel:

   http://www.trylinux.com/projects/udf/

7. Mount the DVD drive on a local directory of the Linux system with the file system type set as udf:

   ```
   # mount -t udf /dev/cd0 /mnt/dvd
   ```

8. If you execute the **mount** command as follows, you should get an output similar to the following:

   ```
   # mount
   /dev/cdrom on /mnt/cdrom type udf (rw,sync)
   ```

9. Now you can read the files on the DVD media as follows:

   ```
   # cd /mnt/cdrom
   # ls
   file1 file2 test_dir qconfig
   ```

   We can see the AIX files which we stored on the DVD media.

UDF offers one more way of sharing data between AIX and Linux. You can also mount the DVD drive from the Linux system to the AIX system over NFS. For example, add the following line in the /etc/exports file of the Linux system:

```
/mnt/cdrom aix_sys(rw)
```

and execute **exportfs -a**.

Now from the AIX system, mount the exported directory on /mnt/dvd of the local system as follows:

```
# mount -v nfs 10.3.4.30:/mnt/cdrom /mnt/dvd
# mount
10.3.4.30 /mnt/cdrom       /mnt/dvd         nfs3   Nov 22 14:51
```

Here, 10.3.4.30 is the IP address of the Linux system acting as the NFS server.

Since the export rules allow the write operation, you can treat the /mnt/dvd directory on AIX as a DVD read-write media on local system:

```
# cd /mnt/dvd
# ls
file1 file2 test_dir qconfig
# touch goliath david
# mkdir casesar
# ls
casesar david file1 file2 goliath qconfig test_dir
```

8.3 Backup and file systems

Traditional low level UNIX backup utilities are dump and restore. These programs are best suited for the system level backups. They operate on a file system inode level and therefore have to know the underlying file system technology. A direct consequence is that they can normally operate on only one file system type. For example, the Linux dump utility is capable of backing up only ext2 and ext3 file systems. Similarly, the restore utility can only understand backup images produced by the corresponding dump utility.

AIX and Linux share only the JFS file system implementation, and we discuss the Linux JFS port in the previous section. Linux supports a myriad of file systems. The most popular journaling file systems are ext3, Reiser FS, SGI XFS, and IBM JFS. Backup of the ext3 file system is supported by the dump and restore programs. SGI XFS has its own pair of backup utilities: xfsdump and xfsrestore. The Reiser FS and IBM JFS do not have backup programs and the only option is to back them up using the GNU tar program. Note that the JFS Linux port is coming from IBM OS/2®. The same version is available on AIX 5L.

On AIX, the backup functionality is provided by the backup and restore programs. The backup utility has two modes of operation: Using file system inodes and using file names. The former is used to back up whole file systems and the latter to archive individual files.

The GNU tar supports file system level backups. It does not operate on the inode level. Hence, it is slower than the dump utility, but it does not care about the underlying file system type. It supports incremental backups and backup of special devices, which is not common for tar implementations. Though a seasoned system administrator would not approve, the GNU tar program is regularly used on Linux to back up file systems. Also, the GNU tar was available for this purpose before the first stable release of the dump and restore utilities and it probably prevails even today as a system backup utility.

Both AIX and Linux have programs capable of doing remote backups through the rmt program. Note that this kind of remote backup is possible only if trust between the involved hosts is on the root user level, which is seldom the case. Still, backup operations may be done through other communication channels which are more appropriate, such as the ssh channel:

```
# dump -0f - -b 10 /usr | ssh backup@tapehost "dd of=/dev/rmt0 bs=10k"
```

It is preferable not to allow remote connection for the root user. Note that it may be necessary to preserve the blocking factor when using the network, as we do in the example.

8.4 Data archiving

User data backup or data transfers are typically done by data archiving. The most popular tools for this purpose are tar and cpio.

The tar (tape archiver) program is probably the oldest archiving programs on UNIX systems. It is still probably the most used archiving tool. The format of tar archives is standard and tar programs are commonly able to read each other's archives. This format was designed for character devices such as magnetic tape, so you should pay attention to the blocking factor. In particular, if the best portability is desired, archives should be created with the blocking factor of 20. This is the default blocking factor on most tar implementations.

AIX and Linux have different tar implementations. The Linux default tar option is the GNU tar program. This tar version is available for AIX as one part of the Linux toolbox project. We recommend you install the GNU tar version on AIX hosts. However, make sure that the AIX tar version is, by default, accessible to system programs, because some option letters are different.

The GNU tar was based on the draft of the ustar standard (POSIX.1). However, current implementations have many extensions that are not POSIX compliant. If the POSIX compliance is desired, one should set the POSIXLY_CORRECT environment variable or use the --posix option when creating an archive.

Some useful but seldom used tar options are significantly different between the GNU and AIX tar programs. One typical example is the "list of files" parameter. The fact that different option letters are used is somehow expected: AIX tar makes use of -L while the GNU tar option is -T. AIX tar cannot read the file list from the standard input. In other words, specifying -L - produces a `File not found` error. Finally, unlike GNU tar, the AIX tar program will not archive directories recursively if they are specified in the input list file. One should be very careful when deploying shell scripts that use tar and are supposed to run on both platforms.

Another popular archiver is cpio. It is well suited for archiving sets of unrelated files. While Linux is equipped with only the GNU cpio version, AIX 5L Version 5.2 may have up to three cpio programs installed: the standard AIX cpio in /usr/bin/cpio, the System V compatibility version in /usr/sysv/bin/cpio, and the GNU version from the Linux toolbox in /opt/freeware/bin/cpio. For the most part, the different versions will cooperate nicely.

However, the AIX System V cpio version does not correctly treat archives created on Linux. It unpacks files without a hitch, but every 16-bit word has its bytes swapped. The problem obviously stems from the fact that two architectures (Intel and PowerPC®) have different endianness.

The pax program is yet another archiver available on AIX and Linux. It combines functionality of tar and cpio. The pax specification and user interface are defined by the POSIX.2 standard (pax stands for portable archive exchange). The Linux pax implementation is a port of the OpenBSD pax program. It is worth noting that the AIX pax program supports files larger than 2 GB, unlike cpio and tar. If you want to archive files of that size, pax or backup are the only options on AIX.

We recommend keeping archives as simple as possible in order to improve portability. If you expect to have the archive distributed to a different platform, try to limit it with the following rules:

- Avoid deeply nested directory trees.
- Limit file names and paths length. Most tar implementations have a 100 character limit for file names, but some old UNIXes have a 14 character limit. Note that the limit is imposed by the file system as well.
- Limit file names to the ASCII standard and use only digits, letters, slash, period, dash, and underscore.
- Avoid large values. Some systems do not support big UID numbers. For example, Linux kernel Version 2.2 supports only 16-bit user and group IDs. Also, do not try to use time stamps that refer to times before January 1, 1970 or which are beyond the year 2038.
- Avoid large files. If your files are really huge (bigger than 2 GB), perhaps you should be using a system backup utility such as dump.
- Dereference soft links. Soft links may cause trouble if extracting on a foreign platform.
- Try to follow standards. Some may argue that extensions to GNU tar are great, but that does not help if the archive is to be unpacked on a system that does not understand them. Standards such as POSIX should be used, because they usually are the most common denominator.

9

Security

One of the main intentions of this redbook is to show you how to make systems running AIX and Linux interoperable. In this chapter, we will show you how to interoperate these two system in a secure way. We will show a program that could be used to protect a network as well as tools that could be used to protect your application.

This chapter discusses the following topics:

- IPSec
- Security Tools

9.1 IPSec

In this topic, we are going to cover how to establish a secure tunnel between two networks. One network will have an AIX gateway and the other one will have a Linux gateway. We will show you how to accomplish this task using IPSec. First, we will cover some concepts regarding IPSec.

IPSec was developed by the Internet Engineering TasK Force (IETF) to be an open security standard for security. IPSec provides cryptography-based protection of all data at the IP layer of the communications stack. No changes are needed for existing applications. IPSec is the industry-standard network-security framework chosen by the IETF for both the IP Version 4 and 6 environments.

IPSec protects your data using the following cryptographic techniques:

- Authentication

  Process by which the identity of a host or end point is verified

- Data Integrity

  Process of ensuring that no modifications were made to the data while in transit across the network

- Encryption

  Process of ensuring privacy by “hiding” data and private IP addresses while in transit across the network

Authentication algorithms prove the identity of the sender and data integrity by using a cryptographic hash function to process a packet of data (with the fixed IP header fields included) using a secret key to produce a unique digest. On the receiver side, the data is processed using the same function and key. If either the data has been altered or the sender key is not valid, the datagram is discarded.

Encryption uses a cryptographic algorithm to modify and randomize the data using a certain algorithm and key to produce encrypted data known as *ciphertext*. Encryption makes the data unreadable while in transit. After it is received, the data is recovered using the same algorithm and key (with symmetric encryption algorithms). Encryption must occur with authentication to verify the data integrity of the encrypted data.

These basic services are implemented in IPSec by the use of the Encapsulating Security Payload (ESP) and the Authentication Header (AH). ESP provides confidentiality by encrypting the original IP packet, building an ESP header, and putting the ciphertext in the ESP payload (see Figure 9-1 on page 181).

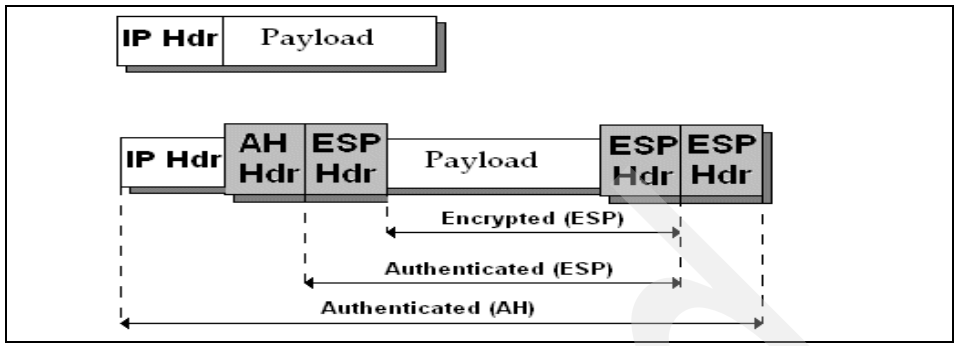

Figure 9-1 IPSec transport mode

The AH can be used alone for authentication and integrity-checking if confidentiality is not an issue. With AH, the static fields of the IP header and the data have a hash algorithm applied to compute a keyed digest. The receiver uses its key to compute and compare the digest to make sure the packet is unaltered and the sender's identity is authenticated.

9.1.1 Security associations (SA)

The building block on which secure communications is built is a concept known as a *security association* (SA). Security associations relate a specific set of security parameters to a type of traffic. With data protected by IPSec, a separate security association exists for each direction and for each header type, AH or ESP. The information contained in the SA includes:

- Security Parameter Indices (SPI)
- IP address (endpoints of tunnels)
- Security protocol (AH or ESP)
- Cryptography algorithm
- Keys used for cryptography
- Modes of encapsulation
- Life time of keys.

9.1.2 Tunnels and key management

To set up a secure communication between two hosts, a security association must be negotiated and managed during the use of the tunnel. The following types of tunnels are supported, each using a different key management technique:

- IKE tunnels
- Manual tunnels

IKE tunnel support

IKE tunnels are based on the ISAKMP/OAkley (Internet Security Association and Key Management Protocol) standards developed by the IETF. With this protocol, security parameters are negotiated and refreshed, and keys are exchanged securely. The following types of authentication are supported: Preshared key and X.509 Version 3 digital certificates' signatures.

The negotiation uses a two-phase approach. The first phase authenticates the communicating parties, and specifies the algorithm to be used for securely communicating in phase 2. During phase 2, the IP Security parameters to be used during data transfer are negotiated, and the security association and keys are created and exchanged.

In AIX 5L Version 5.2, the AIX Internet key exchange (IKE) components now use the system wide pseudo-random number generator (PRNG) as the random number source.

Another enhancement of AIX 5L Version 5.2 AIX IKE is the support for Diffie-Hellman (GH) group 5. Group 5 specifies the 1536-bit Diffie-Hellman group. Prior releases of the AIX only supported DH group 1 and 2. Group 1 specifies the 768-bit Diffie-Hellman group and Group 2 specifies the 1024-bit Diffie-Hellman group. The group 5 option works exactly like group 1 and group 2; however, group 5 provides a higher level of security and requires more process time than group 1 and 2. Diffie-Hellman key exchange is a public key cryptosystem where public values are exchanged to arrive at a symmetric key among the end entities.

The DTD (Document Type Definition) of the IKE database configuration file has been extended to support DH group 5.

Manual tunnel support

Manual tunnels provide backward compatibility, and they interoperate with machines that do not support IKE key management protocols. The disavantage of manual tunnel is that the key values are static. The encryption and

authentication keys are the same for the life of the tunnel and must be manually updated.

Native filtering capability

Filtering is a basic function in which incoming and outgoing packets can be accepted or denied based on a variety of characteristics. This allows a user or system administrator to configure the host to control the traffic between this host and other hosts. Filtering is done on a variety of packet properties, such as source and destination addresses, IP version (4 or 6), subnet masks, protocol, port, routing characteristics, fragmentation, interface, and tunnel definition.

Rules, known as *filter rules*, are used to associate certain kinds of traffic with a particular tunnel. In a a basic configuration for manual tunnels, when a user defines a host-to-host tunnel, filter rules are auto generated to direct all traffic from that host through the secure tunnel. If more specific types of traffic are desired (for example, subnet to subnet), the filter can be edited or replaced to allow precise control of the traffic using a particular tunnel.

For IKE tunnels, the filter rules are also automatically generated and inserted in the filter table once the tunnel is activated.

Similarly, when the tunnel is modified or deleted, the filter rules for that tunnel are automatically deleted, which simplifies IP Security configuration and helps reduce human error. Tunnel definitions can be propagated and shared among machines and firewalls using import and export utilities, which is helpful in the administration of a large number of machines.

After the filter rules are generated, they are stored in a table and loaded into the kernel. When packets are ready to be sent or received from the network, the filter rules are checked in the list from top to bottom to determine whether the packet should be permitted, denied, or sent through a tunnel. The criteria of the rule is compared to the packet characteristics until a match is found or the default rule is reached.

The IP security function also implements filtering of non-secure based on very granular, user-defined criteria, which allows the control of IP traffic between networks and machines that do not require the authentication or encryption properties of IP Security.

Digital certificate support

IPSec supports the use of X.509 Version 3 digital certificates. The Key Manager tool manages certificates requests, maintains the key database, and performs other administrative functions.

9.1.3 Installing IPSec on Linux

The IPSec Implementation for Linux that is most widely used is provided by FreeS/WAN (http://www.freeswan.org).

We are going to install FreeS/WAN in a machine running Red Hat Linux Version 8.0. At the time of writing this redbook, the current FreeS/WAN Version is 1.98b.

Installing FreeS/WAN

The steps necessary to install FreeS/WAN on Linux are as follows:

1. Be sure that your kernel sources are installed:

   ```
   # rpm -q kernel-source-2.4.18-14
   ```

2. If the kernel sources are not installed, install it from the Red Hat installation CD:

   ```
   # rpm -ivh /mnt/cdrom/RedHat/RPMS/kernel-source-2.4.18-14.i386.rpm
   ```

3. Get the current version of FreeS/WAN:

   ```
   # wget ftp://ftp.xs4all.nl/pub/crypto/freeswan/freeswan-*
   ```

4. After that, you can check the integrity of the file using the following command:

   ```
   # md5sum freeswan-1.98b.tar.gz
   ```

5. Compare the result with the freeswan-1.98b.tar.gz.md5; it must be the same. The next step is to create a symbolic link from /usr/src/linux to /usr/src/linux-2.4.18-14:

   ```
   # ln -s /usr/src/linux2.4.18-14 /usr/src/linux
   ```

6. Before we recompile the kernel, we need to edit the /usr/src/linux/Makefile and change the forth line from:

   ```
   EXTRAVERSION = -14custom
   ```

   to

   ```
   EXTRAVERSION = -14
   ```

7. Issue the command:

   ```
   # make mrproper
   ```

8. Now, copy the file that corresponds to your system from the directory /usr/src/linux/configs to /usr/src/linux/.config. In our case:

   ```
   # cp /usr/src/linux/configs/kernel-2.4.18-i686.config
   /usr/src/linux/.config
   ```

9. In the /usr/src/linux directory, issue the following command:

   ```
   # make menuconfig
   ```

10. Do not make any changes; just select `Exit` and then `Yes` to save the current kernel configuration. After that, issue the following commands:

```
# make dep
# make modules
```

11. Those three last steps will recompile your kernel with you current configuration. Now we can install the FreeS/WAN package. Uncompact the file freeswan-1.98b.tar.gz into the directory /usr/local/src:

```
# tar -xvzf freeswan-1.98b.tar.gz -C /usr/local/src
```

12. Before compiling the FreeS/WAN, a patch must be applied. You can find this patch in the freeS/WAN mailing list archives. See the following Web site for more information about this patch.

    http://lists.freeswan.org/pipermail/users/2002-October/014948.html

13. Once the patch is applied, change to the /usr/local/src/freeswan/freeswan-1.98b directory and then issue the following command:

```
# make oldmod
# mkdir /lib/modules/2.4.18-14/kernel/net/ipsec
# cp /usr/src/linux-2.4.18-14/net/ipsec/ipsec.o
/lib/modules/2.4.18-14/kernel/net/ipsec
```

After that, the installation is done. It could be verified. Example 9-1 shows the commands needed to verify the FreeS/Wan installation.

Example 9-1 IPSec verify

```
# 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 forward key for brazuca.itsc.austin.ibm.com     [OK]
Looking for KEY in reverse map: 20.5.10.10.in-addr.arpa     [OK]
Does the machine have at least one non-private address      [OK]
```

Configuring FreeS/WAN

FreeS/WAN is configured through the ipsec.conf and ipsec.secrets files.

The ipsec.conf file contains most of information about your system and your IPSec peers. It is located in the /etc directory.

The ipsec.secrets file contains sensitive authentication data. Its permission should be changed to `600`.

In Figure 9-2, we suggest a test scenario, where we will be configuring a IPSec channel between Network A and Network B. The machines on Network A will be able to access the machines in Network B and vice-versa in a secure way.

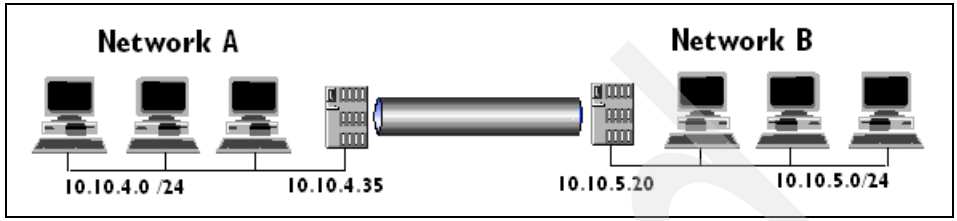

Figure 9-2 Ipsec.conf scenario

In Example 9-2 and, we suggest a ipsec.conf and ipsec.secrets configuration for this scenario. In this configuration, a secure tunnel could be established between two gateway machines. In this manner, we will be connecting the network behind this gateways in a secure way.

Example 9-2 ipsec.conf example

```
config setup
        interfaces=%defaultroute
        klipsdebug=all
        plutodebug=all
        plutoload=%search
        plutostart=%search
        uniqueids=yes

conn %default
        keyingtries=1
        type=tunnel
        auth=esp
        authby=secret
        pfs=no
        keyfile=1800s
        rekeymargin=900s
        rekeyfuzz=90%
        auto=add
        left=%defaultroute

conn aix
        left=10.10.5.20
        leftsubnet=10.10.5.0/24
        right=10.10.4.35
        rightsubnet=10.10.4.0/24
```

Example 9-3 Ipsec.secrets example

```
10.10.5.20          10.10.4.35        : "secret"
```

The ipsec.conf file is a text-based file formed by one or more sections. The first section contains a section that deals specifically with FreeS/WAN settings, like interfaces and connections to be negotiated when IPSec is enabled. The line that defines the default gateway as the interface for all the VPN tunnels is:

```
interfaces=%defaultroute
```

Some parameter values are preceded by a “%”, which indicates a system variable that is loaded by FreeS/WAN when the tunnel is brought up. To turn on the KLIPS and Pluto debugging, we use the following lines:

```
klipsdebug=all
plutodebug=all
```

To tell Pluto to search (%search) for connections to negotiate, we use the following lines:

```
plutoload=%search
plutostart=%search
```

Pluto scans the ipsec.conf file for connections that will be loaded (con sections) and negotiates access for the VPN tunnel when necessary. To set the default settings for all parameters for all tunnels that follow, we use the following line:

```
conn %default
```

To tell IKE to attempt to re-key a failed connection, we use the following line:

```
keyingtries=1
```

Finally to set the authentication method for FreeS/WAN to be a public shared lkey(PSK), we use the following line:

```
authby=secret
```

The public shared key is generally a secret alphanumeric string shared by both parts. This secret is defined in the ipsec.secrets file.

You can find more about Frees/WAN advanced configuration at:

http://www.freeswan.org/freeswan_trees/freeswan-1.98b/doc/adv_config.html

9.1.4 Installing IPSec on AIX

The IPSec feature in AIX is separately installable and loadable. The filesets that need to be installed are as follows:

- bos.net.ipsec.rte (The run-time environment for the kernel IPSec environment commands)
- bos.msg.*LANG*.net.ipsec (where *LANG* is the desired language, such as en_US)
- bos.net.ipsec.keymgt
- bos.net.ipsec.websm
- bos.crypto-priv (The file set for DES and triple DES encryption)

The bos.crypto-priv file set is located on the Expansion Pack. For IKE digital signature support, you must also install the gskit.rte fileset (AIX Version 4) or gskkm.rte (AIX 5L Version 5.1 or later) from the Expansion Pack.

After it is installed, IP security can be separately loaded for Version 4 and IP Version 6, either by using the recommended procedure provided in Loading IP Security or by using the **mkdev** command.

Loading IPSec

> **Attention:** Loading IPSec enables the filtering function. Before loading, it is important to ensure the correct filter rules are created. Otherwise, all communication might be blocked.

Use SMIT or the Web-based System Manager to automatically load the IP security modules when IP Security is started. Also, SMIT and the Web-based System Manager ensure that the kernel extensions and IKE daemons are loaded in the correct order.

If the loading completes successfully, the **lsdev** command shows the IP Security devices as `Avaiable`. See Example 9-4.

Example 9-4 lsdev command output

```
# lsdev -C -c ipsec
ipsec_v4 Defined  IP Version 4 Security Extension
ipsec_v6 Defined  IP Version 6 Security Extension
```

Before starting the IPSec, it is necessary to create a Key Database. A key database is used by the Certificate Proxy Server Daemon (CPSD).

To create a key database using the **`certmgr`** command, use the following procedure:

1. Start the Key Manager tool by typing:

   ```
   # certmgr
   ```

2. Select **New** from the Key Database File pull down menu, as shown in Figure 9-3.

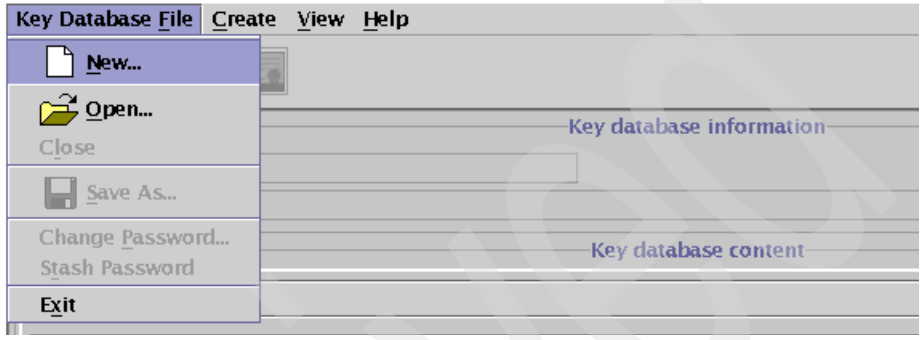

Figure 9-3 Creating a new key database

3. Accept the default value, `CMS key databse file`, for the **Key database type** field.
4. Enter the following file name in the File Name field:

   ```
   ikekey.kdb
   ```

5. Enter the following location of the database in the Location field:

   ```
   /etc/security
   ```

   See Figure 9-4.

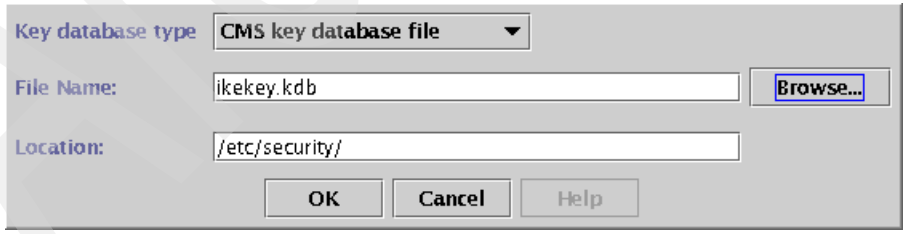

Figure 9-4 Name and location of the new key database

Note: The key database must be named ikekey.kdb and it must be placed in the /etc/security directory; otherwise, IPSec cannot function correctly.

6. Click **OK**. The **Password Prompt** window is displayed.
7. Enter a password in the **Password** field, and enter it again in the **Confirm Password** field.
8. If you want to change the number of days until the password expires, enter the desired number of days in the **Set expiration time?** field. The default value for this field is 60 days. If you do not want the password to expire, clear the **Set expiration time?** field.
9. To save an encrypted version of the password in a stash file, select the **Stash the password to a file?** field and enter YES. See Figure 9-5.

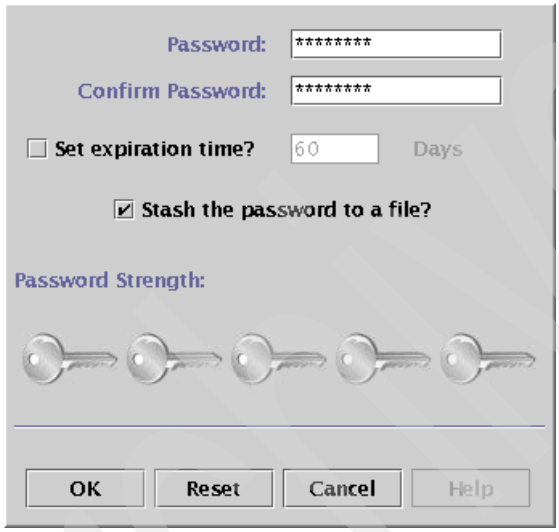

Figure 9-5 Setting password for the new key database

Note: You must stash the password to enable the use of digital certificates with IPSec.

10. Click **OK**. A confirmation window displays, verifying that you have created a key database.

You have just created a key database located in /etc/security/ikekey.kdb. You can close the certificate manager program now and continue with the IPSec configuration.

Now you can start the IPSec service on AIX. We can do this by using the command line or by using the Web-based System Manager.

Starting IPSec using the command line

To start the IPSec on AIX, issue the **startsrc -g ike** command, as shown in Example 9-5.

Example 9-5 Starting IPSec on AIX

```
# startsrc -g ike
0513-059 The cpsd Subsystem has been started. Subsystem PID is 348188.
0513-059 The tmd Subsystem has been started. Subsystem PID is 13216.
0513-059 The isakmpd Subsystem has been started. Subsystem PID is 303328.
```

Starting IPSec using the Web-based System Manager

To start the IPSec using the Web-base System Manager, issue the **wsm** command. A window similar to Figure 9-6 will appear.

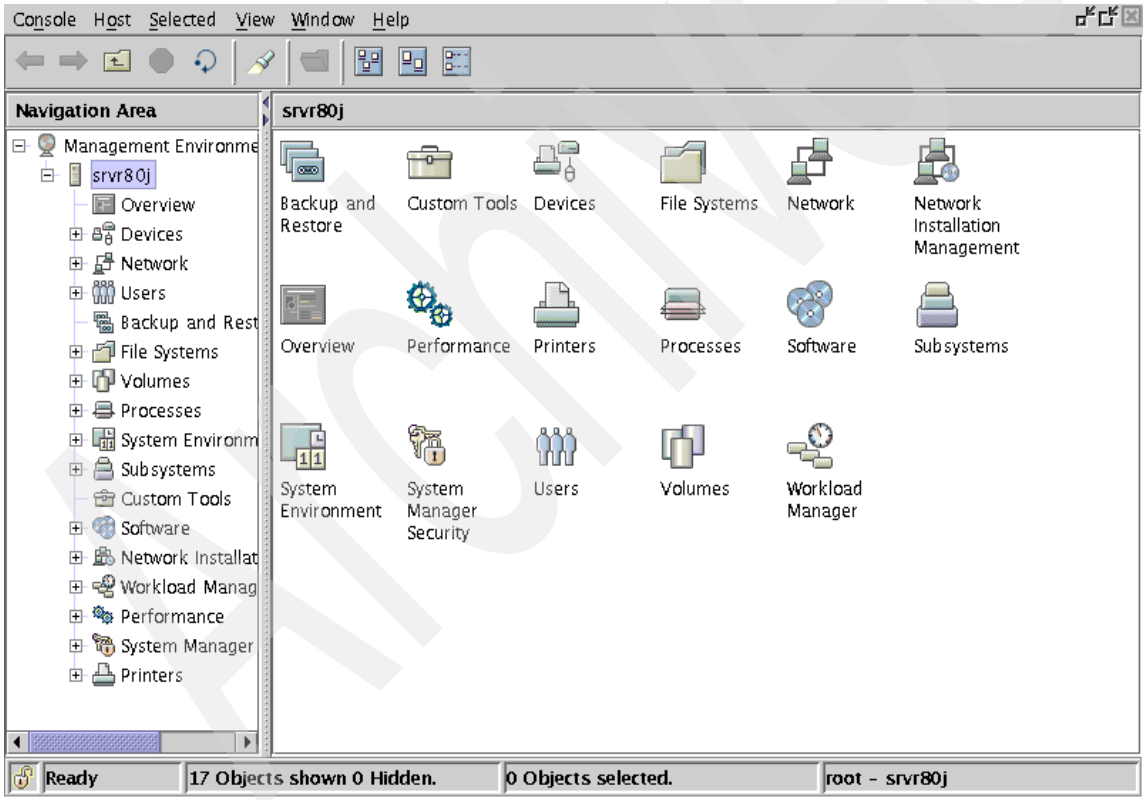

Figure 9-6 Web-based System Manager interface

Expand the **Network** icon, on the left side. Now, expand the **Virtual Private Network (IP Security)** icon. Click on the **Overview and Tasks** icon. You will see a window similar to Figure 9-7.

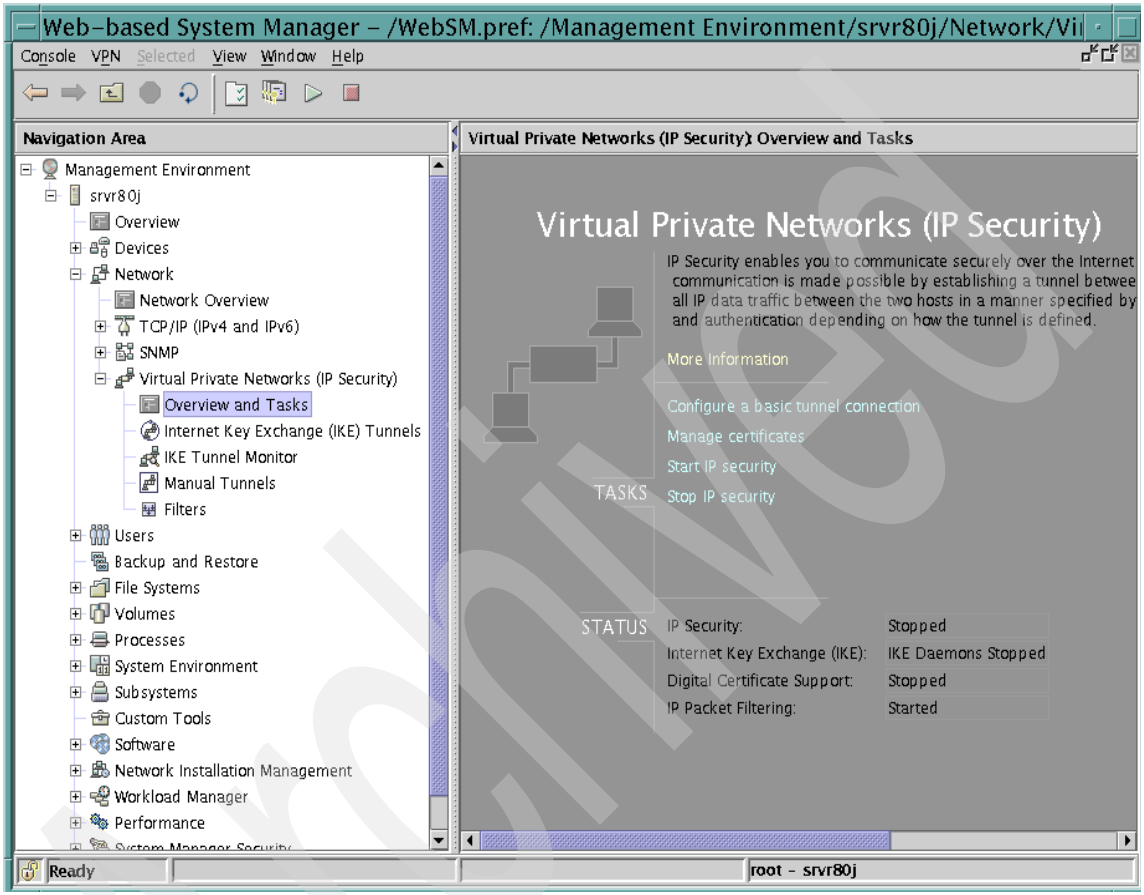

Figure 9-7 IPSec on Web-based System Manager

To start IPSec, click on the **PLAY** button on the top or click on the **Start IP Security** task.

Logging facilities on AIX

This section describes the configuration and format of the system logs related to IPSec on AIX. As hosts communicate with each other, the transferred packets may be logged to the system log daemon, syslogd. Other important messages about IPSec will appear as well. We could use these logs for traffic analysis or to troubleshoot.

The following steps set up the logging facilities:

1. Edit the /etc/syslog.conf file to add the following entry:

   ```
   local4.debug var/adm/ipsec.log
   ```

   Use the local4 facility to record traffic and IP Security events. Standard AIX priority levels apply. You should set the priority level of debug until traffic through IP Security tunnels and filters show stability and proper movement.

   > **Note:** The logging of filter events can create significant activity at the IPSec host and can consume large amounts of storage.

   Save /etc/syslog.conf.

2. Go to the directory you specified for the log file and create an empty file with the same name. In the case above, you would change to the /var/adm directory and issue the command:

   ```
   # touch ipsec.log
   ```

3. Issue a **refresh** command to the syslogd subsystem:

   ```
   # refresh -s syslogd
   ```

Now all messages from IPSec could be easily checked in /var/adm/ipsec.log. On Linux, the IPSec messages are redirected to /var/log/messages.

Interoperating AIX and Linux

Our intention is to establish a secure tunnel between an AIX system and a Linux system. We are able to accomplish this task at this point.

In 9.1.3, “Installing IPSec on Linux” on page 184, we explain how to install and configure IPSec on a Linux system using the FreeS/WAN program. Now we will cover how to import that data from Linux to AIX and then establish a connection.

AIX provides a conversion facility through the **ikedb** command to facilitate interoperability with Linux. The -c option of the **ikedb** command provides an easy way for the user to configure a tunnel between a Linux and AIX machine. To use the -c option, define the tunnel between Linux and AIX on the Linux system. Two files were created on the Linux machine, ipsec.conf and ipsec.secrets, which are located in /etc directory.

The ipsec.conf file contains information about the tunnel while ipsec.secrets holds the keys. A sample ipsec.conf was provided in Example 9-2 on page 186 and a sample ipsec.secrets was provided in Example 9-3 on page 187.

Once these files are created on Linux. move the files to the AIX machine by ftp, scp, or any other method. Use the **ikedb -c** command to convert the Linux

configuration to the corresponding AIX configuration and load the information into the AIX database. In the example that follows, it is assumed that the user already has the ipsec.conf and ipsec.secrets files in the /tmp directory, as shown in Example 9-6.

Example 9-6 ikedb command output

```
# cd /tmp
# ikedb -c
Warning: This installation of ikedb uses version 2.0 of DTD.
Your XML file corresponds to version 1.0 of the DTD.
ikedb can read the older format, but this usage is deprecated.
You should update your XML file to use the latest DTD format.
```

Before the tunnel can be activated, the ike daemons should be started. The commands for starting (restarting) the daemons are as follows:

- On Linux:

  ```
  # ipsec setup start
  ```

- ON AIX:

  ```
  # stopsrc -g ike
  # startsrc -g ike
  ```

The tunnel can be activated from either Linux or AIX. The command to activate the tunnel, as per the example that we used, is as follows:

- On Linux:

  ```
  # ipsec auto --up aix
  ```

- On AIX:

  ```
  # ike cmd=activate
  ```

These commands start the secure tunnel between the AIX and Linux machines. We can check the status of IPSec on Linux and AIX, as shown in Example 9-7 and Example 9-8 on page 195.

Example 9-7 Verifying the IPSec status on AIX

```
# ike cmd=list
Phase  Tun Id    Status       Local Id                  Remote Id
1      1         Negotiating  N/A                       10.10.5.20
2      1         Initial      10.10.4.0/255.255.255.0   10.10.5.0/255.255.255.0

# ike cmd=list
Phase  Tun Id    Status       Local Id                  Remote Id
1      1         Active       10.10.4.35                10.10.5.20
2      1         Active       10.10.4.0/255.255.255.0   10.10.5.0/255.255.255.0
```

Example 9-8 Verifying IPsec status on Linux

```
# ipsec look
brazuca.itsc.austin.ibm.com Wed Oct 30 10:39:04 CST 2002
10.10.5.0/24          -> 10.10.4.0/24       => tun0x1006@10.10.4.35
esp0xf4e0e378@10.10.4.35     (12)
ipsec0->eth0 mtu=16260(1443)->1500
esp0xa7102a45@10.10.4.35 ESP_3DES_HMAC_SHA1: dir=out src=10.10.5.20
iv_bits=64bits iv=0x89ab8f9a49358299 ooowin=64 alen=160 aklen=160 eklen=192
life(c,s,h)=addtime(1203,0,0)
esp0xb479e1bf@10.10.5.20 ESP_3DES_HMAC_SHA1: dir=in src=10.10.4.35
iv_bits=64bits iv=0xac3e8075022c4cc7 ooowin=64 alen=160 aklen=160 eklen=192
life(c,s,h)=addtime(1207,0,0)
esp0xb479e1c0@10.10.5.20 ESP_3DES_HMAC_SHA1: dir=in src=10.10.4.35
iv_bits=64bits iv=0x83aefe0650016fe9 ooowin=64 alen=160 aklen=160 eklen=192
life(c,s,h)=addtime(1203,0,0)
esp0xb479e1c1@10.10.5.20 ESP_3DES_HMAC_SHA1: dir=int src=10.10.4.35
iv_bits=64bits iv=0xb9473f48a2f8d48e ooowin=64 alen=160 aklen=160 eklen=192
life(c,s,h)=addtime(798,0,0)
esp0xbed80e62@10.10.4.35 ESP_3DES_HMAC_SHA1: dir=out src=10.10.5.20
iv_bits=64bits iv=0x538db87912fe2b3b ooowin=64 alen=160 aklen=160 eklen=192
life(c,s,h)=addtime(1205,0,0)
esp0xf4e0e378@10.10.4.35 ESP_3DES_HMAC_SHA1: dir=out src=10.10.5.20
iv_bits=64bits iv=0x1ece8b507accff08 ooowin=64 seq=6 alen=160 aklen=160
eklen=192
life(c,s,h)=bytes(656,0,0)addtime(798,0,0)usetime(157,0,0)packets(6,0,0)
idle=128
tun0x1001@10.10.5.20 IPIP: dir=in src=10.10.4.35 life(c,s,h)=addtime(1207,0,0)
tun0x1002@10.10.4.35 IPIP: dir=out src=10.10.5.20 life(c,s,h)=addtime(1205,0,0)
tun0x1003@10.10.5.20 IPIP: dir=in src=10.10.4.35 life(c,s,h)=addtime(1203,0,0)
tun0x1004@10.10.4.35 IPIP: dir=out src=10.10.5.20 life(c,s,h)=addtime(1203,0,0)
tun0x1005@10.10.5.20 IPIP: dir=in src=10.10.4.35 life(c,s,h)=addtime(798,0,0)
tun0x1006@10.10.4.35 IPIP: dir=out src=10.10.5.20
life(c,s,h)=bytes(448,0,0)addtime(797,0,0)usetime(157,0,0)packets(6,0,0)
idle=128
Destination     Gateway         Genmask         Flags     MSS Window   Irrt Iface
0.0.0.0         10.10.4.41      0.0.0.0         UG         40 0           0 eth0
10.10.4.0       0.0.0.0         255.255.255.0   U          40 0           0 ipsec0
10.10.4.0       10.10.4.35      255.255.255.0   UG         40 0           0 ipsec0
```

In AIX, we realize that the IPSec tunnel is activated when the Status changes to Activate. On Linux, we realize that the tunnel is established when the output of the **ipsec look** command presents information about the tunnels and encryption been used by the tunnel; otherwise, just the information regarding the routes and interfaces will be shown. In Example 9-8, if the IPSec tunnel is not establish, just the first two and the last four lines would be showed in the output of the **ipsec look** command.

We can also check the logs files. In AIX, we will check the file that we configured on syslogd, in our case, /var/adm/ipsec.log. On Linux, we have to check the default log file, /var/log/messages.

On AIX, you have a graphical interface (Web-based System Manager) that works with IPSec. We can check the status of IKE tunnels, as shown in Figure 9-8.

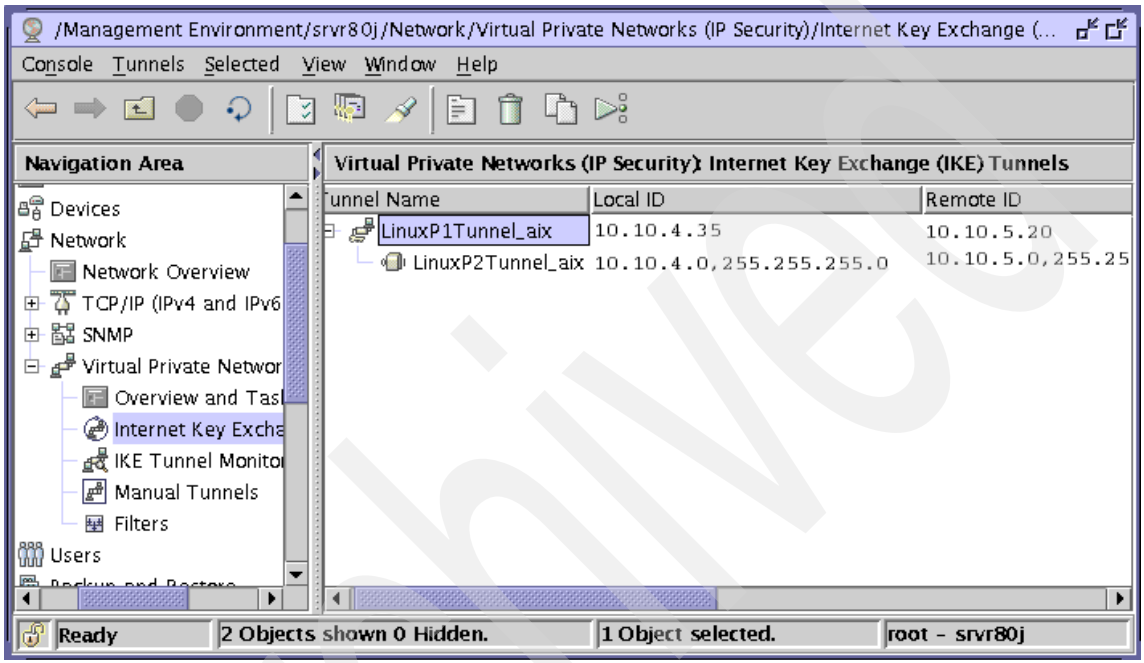

Figure 9-8 Web-based System Manager IKE Tunnels status

The LinuxP1Tunnel_aix tunnel is the tunnel created by the system when we used the **ikedb -c** command to import the linux configuration files on AIX. Another option avaiable on Web-based System Manager for IPSec is IKE Tunnel Monitor, as shown in Figure 9-9 on page 197.

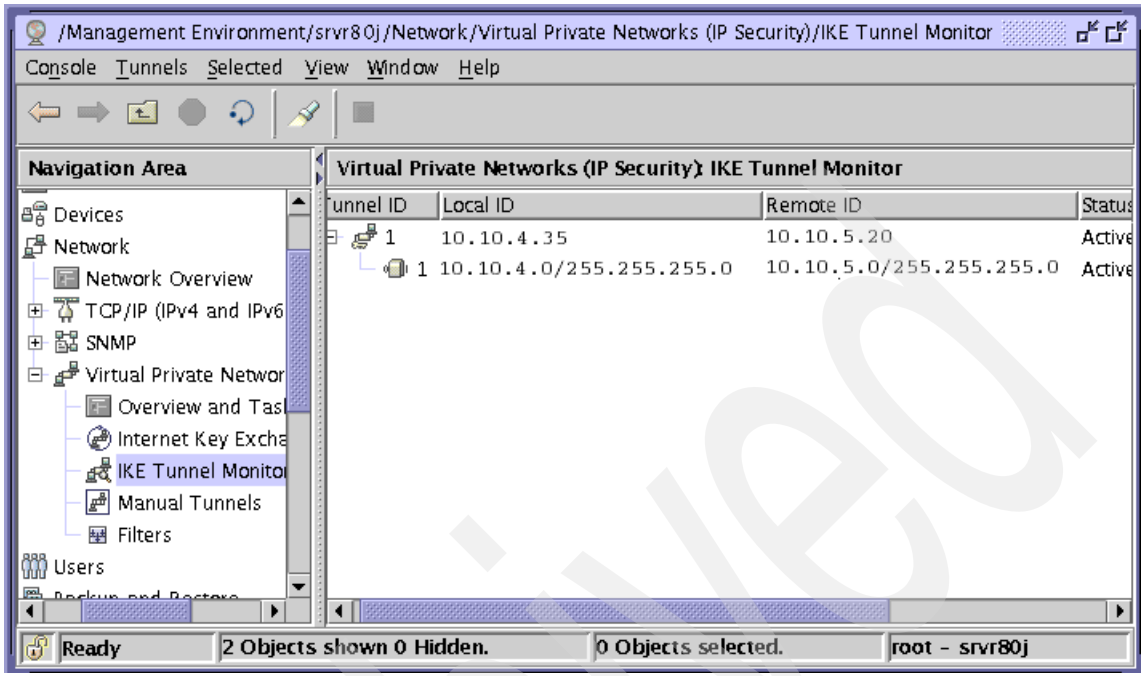

Figure 9-9 Web-based System Manager IKE Tunnel monitor

The IKE Tunnel monitor can supply statistics and information about the activated tunnels. In our case, the tunnel ID number 1 represents the tunnel between the Linux and AIX machine.

At this point, the installation and configuration of IPSec is completed and you established a secure tunnel between your Linux and AIX machine. This tunnel will ensure that all communication between Network A and Network B will be secure.

To verify if the traffic is really encrypted between Network A and B, you could do a simple test: Put a sniffer between the gateways A and B, now start a ftp session from any machine on Network A to any machine on Network B. You will realize that the packets captured by the sniffer are encrypted, which means you cannot visualize any data inside the packets. IPSec is working.

9.2 Security tools

In this chapter, we will cover some tools avaiable on Linux and AIX that could be used to enhance the security of those systems.

9.2.1 OpenSSL

Many server software that ask for an user's authentication before allowing services transmit, by default, the authentication information (user and password) in plain text. Using encryption mechanisms like SSL ensures safe and secure transactions. Using SSL, data transmitted over the network is point-to-point protected.

The SSL protocol runs above TCP/IP and below higher-level protocols, such as HTTP or IMAP. It uses TCP/IP on behalf of the higher-level protocols, and in the process allows an SSL-enabled server to authenticate itself to an SSL-enabled client, allows the client to authenticate itself to the server, and allows both machines to establish an encrypted connection.

The primary goal of cryptography is to secure important data as it passes through a medium that may not be secure itself. Its medium could be the Internet, for example.

There are many different cryptographic algorithms, each of which can provide one or more of the following services to applications:

- Confidentiality

  This is the guarantee that only the parties involved in a communication will be able to access the exchange information, even if the data travels through an insecure medium. In practice, when a message is encrypted, the plain text is transformed by an algorithm into enciphered text that hides the meaning of the message and can be sent via any public medium. This process involves a secret key that is used to encrypt and later decrypt the data. Without the secret key, the encrypted data has no meaning.

- Integrity

  There is a way for the recipient of data to determine if any modifications were made in the data. A cryptographic checksum, called Message Authentication Code (MAC), can be calculated over a user supplied text to protect the integrity of data. After that, the sender encrypts the text. Now he sends the encrypted text and the MAC to the receiver, which can verify the integrity of the message decrypting the receive message, extracting the MAC of the message and comparing with the MAC received.

- Authentication

  Cryptography can be used to protect personal identification.

- Non-repudiation

  Cryptography can enable a receiver to prove that a message he received from a given sender actually came from the given sender.

Now that you know a little more about cryptography and SSL, it is time to look specifically at the OpenSSL library. OpenSSL is derived from SSLeay. SSLeay was originally written by Eric A. Young and Tim J. Hudson in the beginning of 1995. In December 1998, development of SSLeay ceased, and the first version of OpenSSL was released as 0.9.1c, using SSLeay 0.9.1b as its starting point.

OpenSSL is essentially two tools in one: A cryptography library and a SSL toolkit.

The SSL library provides an implementation of all versions of the SSL protocol, including TLSv1. The cryptography library provides the most popular algorithms for symmetric key and public key cryptography, hash algorithms, and message digests. It also support a pseudorandom number generator, and support for manipulating common certificate formats and managing key material. There are also general-purpose helper libraries for buffer manipulation and manipulation of arbitrary precision numbers.

OpenSSL is the only free, full-featured SSL implementation currently available for use with the C and C++ programming languages. It works across every major platform, including all UNIX OSs and all common version of Microsoft Windows.

OpenSSL is avaiable for download from:

http://www.openssl.org

Installing OpenSSL on AIX

The OpenSSL binaries is available on Linux and AIX.

You can download the OpenSSL packages for AIX from the AIX Linux Toolbox site. You must first register in order to download these packages. You can register at the following Web site:

http://www.ibm.com/servers/aix/products/aixos/linux/rpmgroups.html

When you reach this page, click on the AIX Toolbox Cryptographic Content link. You should then see a registration page. After you have registered, you can download the OpenSSL packages.

To install the package execute the following command:

```
# rpm -ivh openssl-0.9.6e-2.aix4.3.ppc.rpm
```

```
openssl                    ##########################################
```

The OpenSSL binary is located at /usr/linux/bin directory. You could invoke it by issuing the following command:

```
# /usr/linux/bin/openssl
OpenSSL>
```

Later, we will cover how to use OpenSSL by using the command line.

Installing OpenSSL on Linux

In Red Hat Linux Version 8.0, OpenSSL is located on installation disk number 1.

To install the package, issue the following command:

```
# rpm -ivh openssl-0.9.6b-29.i686.rpm
```

The OpenSSL binary is installed by default in /usr/bin.

Configurations

OpenSSL includes a default configuration file that is used unless an alternate one is specified. The default file is openssl.cnf. The configuration file's default settings are all quite reasonable, but it can be useful to replace them with settings that are better tailored to your needs. The location of configuration file is different on Linux and AIX. You can check the location of the configuration file in any operation system by issuing the command:

```
# openssl ca
```

Any errors that are issued due the lack of options may be ignored. By default, on Linux, the configuration file is /usr/share/ssl/openssl.cnf, and on AIX, the configuration file is /var/ssl/openssl.cnf.

Just three of the many commands supported by the command line tool make use of this configuration file. The commands that use it are **ca**, **req**, and **x509** (we will talk about these commands later).

The configuration file for OpenSSL program is where you can configure the expiration date of your keys, the name of your organization, the address, the country, and so on. The parameters you may change will be in the [CA_default] and especially the [req_distinguised_name] sections. Example 9-9 shows an OpenSSL configuration file.

Example 9-9 OpenSSL configuration file

```
#
# OpenSSL example configuration file.
# This is mostly being used for generation of certificate requests.
```

```
#
# This definition stops the following lines choking if HOME isn't
# defined.
HOME                    = .
RANDFILE                = $ENV::HOME/.rnd

# Extra OBJECT IDENTIFIER info:
#oid_file               = $ENV::HOME/.oid
oid_section             = new_oids

# To use this configuration file with the "-extfile" option of the
# "openssl x509" utility, name here the section containing the
# X.509v3 extensions to use:
# extensions            =
# (Alternatively, use a configuration file that has only
# X.509v3 extensions in its main [= default] section.)

[ new_oids ]

# We can add new OIDs in here for use by 'ca' and 'req'.
# Add a simple OID like this:
# testoid1=1.2.3.4
# Or use config file substitution like this:
# testoid2=${testoid1}.5.6

####################################################################
[ ca ]
default_ca      = CA_default            # The default ca section

####################################################################
[ CA_default ]

dir             = /var/ssl              # Where everything is kept
certs           = $dir/certs            # Where the issued certs are kept
crl_dir         = $dir/crl              # Where the issued crl are kept
database        = $dir/index.txt        # database index file.
new_certs_dir   = $dir/newcerts         # default place for new certs.


certificate     = $dir/cacert.pem       # The CA certificate
serial          = $dir/serial           # The current serial number
crl             = $dir/crl.pem          # The current CRL
private_key     = $dir/private/cakey.pem# The private key
RANDFILE        = $dir/private/.rand    # private random number file

x509_extensions = usr_cert              # The extentions to add to the cert

# Extensions to add to a CRL. Note: Netscape communicator chokes on Version 2
CRLs
```

```
# so this is commented out by default to leave a V1 CRL.
# crl_extensions        = crl_ext

default_days    = 365                   # how long to certify for
default_crl_days= 30                    # how long before next CRL
default_md      = md5                   # which md to use.
preserve        = no                    # keep passed DN ordering

# A few difference way of specifying how similar the request should look
# For type CA, the listed attributes must be the same, and the optional
# and supplied fields are just that :-)
policy          = policy_match

# For the CA policy
[ policy_match ]
countryName             = match
stateOrProvinceName     = match
organizationName        = match
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

# For the 'anything' policy
# At this point in time, you must list all acceptable 'object'
# types.
[ policy_anything ]
countryName             = optional
stateOrProvinceName     = optional
localityName            = optional
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

####################################################################
[ req ]
default_bits            = 1024
default_keyfile         = privkey.pem
distinguished_name      = req_distinguished_name
attributes              = req_attributes
x509_extensions = v3_ca # The extentions to add to the self signed cert

# Passwords for private keys if not present they will be prompted for
# input_password = secret
# output_password = secret

# This sets a mask for permitted string types. There are several options.
# default: PrintableString, T61String, BMPString.
# pkix   : PrintableString, BMPString.
```

```
# utf8only: only UTF8Strings.
# nombstr : PrintableString, T61String (no BMPStrings or UTF8Strings).
# MASK:XXXX a literal mask value.
# WARNING: current versions of Netscape crash on BMPStrings or UTF8Strings
# so use this option with caution!
string_mask = nombstr

# req_extensions = v3_req # The extensions to add to a certificate request

[ req_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_default             = US
countryName_min                 = 2
countryName_max                 = 2

stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default     = Texas

localityName                    = Locality Name (eg, city)
localityName_default            = Austin

O.organizationName              = Organization Name (eg, company)
O.organizationName_default      = IBM
# we can do this but it is not needed normally :-)
#1.organizationName             = Second Organization Name (eg, company)
#1.organizationName_default     = World Wide Web Pty Ltd

organizationalUnitName          = Organizational Unit Name (eg, section)
organizationalUnitName_default  = International Technical Support Organization

commonName                      = srvr80j.itsc.austin.ibm.com
commonName_max                  = 64

emailAddress                    = admin@srvr80j.itsc.austin.ibm.com
emailAddress_max                = 40

# SET-ex3                       = SET extension number 3

[ req_attributes ]
challengePassword               = A challenge password
challengePassword_min           = 4
challengePassword_max           = 20

unstructuredName                = An optional company name

[ usr_cert ]

# These extensions are added when 'ca' signs a request.
```

```
# This goes against PKIX guidelines but some CAs do it and some software
# requires this to avoid interpreting an end user certificate as a CA.

basicConstraints=CA:FALSE

# Here are some examples of the usage of nsCertType. If it is omitted
# the certificate can be used for anything *except* object signing.

# This is OK for an SSL server.
# nsCertType                    = server

# For an object signing certificate this would be used.
# nsCertType = objsign

# For normal client use this is typical
# nsCertType = client, email

# and for everything including object signing:
# nsCertType = client, email, objsign

# This is typical in keyUsage for a client certificate.
# keyUsage = nonRepudiation, digitalSignature, keyEncipherment

# This will be displayed in Netscape's comment listbox.
nsComment                       = "OpenSSL Generated Certificate"

# PKIX recommendations harmless if included in all certificates.
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always

# This stuff is for subjectAltName and issuerAltname.
# Import the e-mail address.
# subjectAltName=email:copy

# Copy subject details
# issuerAltName=issuer:copy

#nsCaRevocationUrl              = http://www.domain.dom/ca-crl.pem
#nsBaseUrl
#nsRevocationUrl
#nsRenewalUrl
#nsCaPolicyUrl
#nsSslServerName

[ v3_req ]

# Extensions to add to a certificate request

basicConstraints = CA:FALSE
```

```
keyUsage = nonRepudiation, digitalSignature, keyEncipherment

[ v3_ca ]


# Extensions for a typical CA

# Extensions for a typical CA


# PKIX recommendation.

subjectKeyIdentifier=hash

authorityKeyIdentifier=keyid:always,issuer:always

# This is what PKIX recommends but some broken software chokes on critical
# extensions.
#basicConstraints = critical,CA:true
# So we do this instead.
basicConstraints = CA:true

# Key usage: this is typical for a CA certificate. However since it will
# prevent it being used as an test self-signed certificate it is best
# left out by default.
# keyUsage = cRLSign, keyCertSign

# Some might want this also
# nsCertType = sslCA, emailCA

# Include e-mail address in subject alt name: another PKIX recommendation
# subjectAltName=email:copy
# Copy issuer details
# issuerAltName=issuer:copy

# DER hex encoding of an extension: beware experts only!
# obj=DER:02:03
# Where 'obj' is a standard or added object
# You can even override a supported extension:
# basicConstraints= critical, DER:30:03:01:01:FF

[ crl_ext ]

# CRL extensions.
# Only issuerAltName and authorityKeyIdentifier make any sense in a CRL.

# issuerAltName=issuer:copy
authorityKeyIdentifier=keyid:always,issuer:always
```

Using OpenSSL

Now we will cover some uses for OpenSSL.

Message Digest Algorithms

Message Digest Algorithms are cryptographic hash functions, also known as Message Digest Algorithms, which can be used for computing a block of data. OpenSSL supports MD2, MD4, MD5, MDC2, SHA1, and RIPEMD-160. SHA1 and RIPEMD-160 produce 160-bit hashes, and the others all produce 128-bit hashes.

You can use the most supported algorithms by using the command line. The **dgst** command is the main command for accessing message digests, but you can access the others algorithms using a command line of the same name as the algorithm. There is an exception for RIPEMD-160 that is named rmd160.

Computing a hash for a block of data is the default operation performed by the message digest commands.

The message hashes commands can also be used for signing and verifying signatures. When signing, a signature is generated for the hash of the file to be signed. A private key is required to sign, and RSA and DSA keys can be used. When you used a DSA private key, you must use the DSS1 message digest. With the RSA key, you can use any other algorithm. Verifying a signature is simply the reverse of signing.

The basic usage for the message digest command is simple:

```
# openssl dgst -<digest> file
```

where the <digest> is md5, md4, md2, sha1, sha, mdc2, ripemd160, and dss1. Example 9-10 shows the hash of a file called redbook.txt.

Example 9-10 OpenSSL digest modes

```
bash-2.05a# openssl dgst -md5 redbook.txt
MD5(redbook.txt)= 7969c5b2225628c612908bf85cdb74eb
bash-2.05a# openssl dgst -md4 redbook.txt
MD4(redbook.txt)= ff93512641346e7a8e3e7527ebd2a80e
bash-2.05a# openssl dgst -md2 redbook.txt
MD2(redbook.txt)= 71836005e1bb637328b92ea9327534f0
bash-2.05a# openssl dgst -sha1 redbook.txt
SHA1(redbook.txt)= 6c4b875a5fc9d2b7b21e031c88e74739225e33a7
bash-2.05a# openssl dgst -sha redbook.txt
SHA(redbook.txt)= f4e09a291e756ae46b2d47102ed8b39343265608
bash-2.05a# openssl dgst -mdc2 redbook.txt
MDC2(redbook.txt)= 49ad048d8ab46ef70f744f60ed50535e
bash-2.05a# openssl dgst -ripemd160 redbook.txt
```

Symmetric key cipher

OpenSSL supports a wide variety of symmetric ciphers. Symmetric key encryption means that there is one password or passphrase used to encrypt or decrypt with symmetric key encryption. The basic ciphers supported are Blowfish, CAST5, DES, 3DES, IDEA, RC2, RC4, and RC5. Most of the supported symmetric ciphers support a variety of different modes, including CBC, CFB, ECB, and OFB:

- ECB (Electronic Code Block) mode is the basic mode of operation. In this mode, the cipher takes a single block of data and produces a single block of cipher-data. This mode is padded to accommodate messages that are not a multiple of the cipher's block size length.The mode is susceptible to dictionary attacks. We strongly recommend that you not use it unless you really know what you are doing. The advantage of ECB over the other cipher is that messages can be encrypted in parallel.
- CBC (Cipher Block Chaining) mode solves the dictionary problem with the ECB cipher. It does it by XORing the cipherdata of one block with the data of the next block. Parallelization is not possible in this mode. CBC is still a block-based mode, so the padding is generally used. In this mode, a dictionary attack is possible if the data streams have common beginning sequences. For this reason, it is possible to set a initialization vector (IV), which is a block of data that is XOR'd with the first block of data before encrypting. The value of the IV need to be secret, but it should be random. The receiver must know the IV value to be able to decrypt the cipherdata.
- CFB (Cipher Feedback) mode is one way of turning a block cipher into a stream cipher. A complete block of data must be received before the encryption begins. As with CBC, CFB mode can use a initialization vector.
- OFB (Output Feedback) mode is another way of turning a block cipher into a stream cipher.

The **`openssl enc`** command uses both block and stream ciphers. The block cipher encrypts and decrypts 64-bit blocks of data at a time, whereas the stream cipher can use blocks of variable lengths. Most symmetric key algorithms today are block ciphers.

RC4, when implemented with only 40-bit keys, can be broken by brute force. Another cipher that has been broken is 56-bit single DES. In small keys (less than 128-bit), CAST, IDEA, RC2, and RC5 have all shown some type of weakness or have been broken.

To encrypt a file using a symmetric key, you can use Example 9-11, Example 9-13, and Example 9-15 on page 209.

In Example 9-11, we show how to encrypt the contents of the redbook.txt file using DES3 in CBC mode and place the ciphertext into redbook.bin. Note that no password was specified, so a prompt for a password will be presented.

Example 9-11 OpenSSL encryption using des3 in CBC mode

```
# openssl enc -des3 -salt -in redbook.txt -out redbook.bin
enter des-ede3-cbc encryption password:
Verifying password - enter des-ede3-cbc encryption password:
```

Now we have a binary file named redbook.bin. If you look at the contents, you will get a lot of random characters, but binary formats are not readable by humans.

In Example 9-12, we decrypt the contents of the redbook.bin file created in Example 9-11. The password “brazil” will be used to decrypt the file.

Example 9-12 OpenSSL decryption using des3 in CBC mode

```
# openssl enc -des3 -d -in redbook.bin -out redbook.txt -pass pass:brazil
```

In Example 9-13, we encrypt the contents of the redbook.txt file using the Blowfish cipher in CFB mode and place the resulting ciphertext into redbook.bin. The contents of the environment variable PASSWORD will be used to generate the key.

Example 9-13 OpenSSL encryption using Blowfish in CFB mode

```
# export PASSWORD=brazil
# openssl enc -bf-cfb  -salt -in redbook.txt -out redbook.bin -pass
env:PASSWORD
```

In Example 9-14, we show how to decrypt the redbook.bin file generated by Example 9-13.

Example 9-14 OpenSSL decryption using Blowfish in CFB mode

```
# openssl enc -bf-cfb  -salt -d -in redbook.bin -out redbook.txt -pass
env:PASSWORD
```

In Example 9-15 on page 209, we encode the contents of the redbook.bin file in base64 and write the result to the base64.txt file.

Example 9-15 OpenSSL encryption using base64

```
# openssl enc -base64 -in redbook.bin -out base64.txt
```

In Example 9-16, we encrypt the contents of the redbook.txt file using the RC2 cipher in CBC mode and place the resulting ciphertext into redbook.bin. The salt, key, and initialization vector were specified and will be used to encrypt the plaintext. The keys are in hexadecimal.

Example 9-16 OpenSSL encryption using RC2

```
openssl enc -rc2 -in redbook.txt -out redbook.bin -S E453A1D55B3C95AA -iv
A61B7C8D837B7A03 -K 5A058C8D1A8C9A7E839A239B61C8A4D7
```

Public key

Using symmetric key encryption, we have to distribute only one key to everybody involved in the cryptography process and it is not an easy task. In a simplistic way, the public key cryptographic takes care of the need to distribute keys in a secret manner.

Depending on the algorithm used, public key cryptography is useful for key agreement, digital signing, and encryption. Three commonly used public key algorithms are supported by OpenSSL: Diffie-Hellman (DH), DSA (Digital Signature Algorithm), and RSA. These algorithms are not interchangeable. Diffie-Hellman is useful for key agreement, but cannot be used for digital signature or encryption. DSA is useful for digital signatures, but is incapable of providing key agreement or encryption services. RSA can be used for key agreement, digital signing, and encryption.

Public keys are more complex than secret keys. Their strength are n the size of their keys, which are usually very large numbers. As a result, operations involving public key cryptography are slow. Most often, it is used in combination with other cryptographic algorithms, such as message digests and symmetric cipher.

Diffie-Hellman

Diffie-Hellman is used for key agreement. Key agreement is the exchange of information over an insecure medium that allows each of the two parties involved in a conversation to compute a value that is typically used as key for a symmetric cipher. Diffie-Hellman cannot be used for authentication or encryption by itself, because it just provides secrecy. Some means of authentication for parties should be used.

In Example 9-17, we generate a set of Diffie-Hellman parameters using a generator of 2 and a random 1024-bit prime, and write the parameters in PEM format to the file dhparam.pem.

Example 9-17 Generating new set of Diffie-Hellman parameters

```
# openssl dhparam -out dhparam.pem -2 1024
warning, not much extra random data, consider using the -rand option
Generating DH parameters, 1024 bit long safe prime, generator 2
This is going to take a long time
.......................+.........................+..........++*++*++*
```

In Example 9-18, we read a set of Diffie-Hellman parameters from the dhparam.pem file and write a C code representation of the parameters to stdout.

Example 9-18 Reading Diffie-Hellman parameters and writing to stdout in C

```
# openssl dhparam -in dhparam.pem -noout -C
#ifndef HEADER_DH_H
#include <openssl/dh.h>
#endif
DH *get_dh1024()
        {
        static unsigned char dh1024_p[]={
                0xAF,0x2C,0xA1,0x7E,0x6C,0x3D,0x71,0x7F,0x6A,0x61,0x0C,0xEC,
                0xDE,0xD8,0xAA,0x74,0x7B,0xF7,0x96,0xCC,0x30,0xF4,0x97,0x5B,
                0xC2,0x91,0x79,0x7A,0xCB,0x22,0x1D,0x75,0xA4,0xEE,0xDA,0x89,
                0xEA,0x62,0x6D,0x9A,0x16,0x53,0x05,0xA4,0x04,0xFA,0x40,0x13,
                0xE4,0x31,0x76,0xF3,0x5A,0xEF,0x0B,0x7B,0xF0,0xDB,0x87,0x92,
                0x26,0xB7,0xC7,0xC7,0x2F,0xBF,0xA6,0xC4,0x12,0xA1,0x91,0xD3,
                0xE8,0xC7,0x16,0x84,0x23,0x56,0xC6,0x98,0xDE,0xAB,0xB7,0xDD,
                0x54,0x32,0x92,0xD3,0x60,0x0F,0x71,0xBD,0xB1,0xB4,0x12,0xA0,
                0x4E,0x86,0xB1,0xB4,0x88,0x7A,0x34,0x39,0x73,0xC1,0x62,0x2F,
                0x9A,0xCC,0xDA,0xCE,0xA8,0xED,0x32,0x6B,0x4B,0xD6,0x7D,0x69,
                0x57,0xAE,0x0D,0x22,0x2C,0x8A,0x81,0xCB,
                };
        static unsigned char dh1024_g[]={
                0x02,
                };
        DH *dh;

        if ((dh=DH_new()) == NULL) return(NULL);
        dh->p=BN_bin2bn(dh1024_p,sizeof(dh1024_p),NULL);
        dh->g=BN_bin2bn(dh1024_g,sizeof(dh1024_g),NULL);
        if ((dh->p == NULL) || (dh->g == NULL))
                { DH_free(dh); return(NULL); }
        return(dh);
        }
```

Digital signature

The digital signature algorithm (DSA) is used for creating and verifying digital signatures. Using a private key, the user can compute a signature for a given piece of data. Anyone possessing the user's public key that corresponds to the private key used to compute a signature can then verify the signature. The hash of data to be signed is computed, and the hash is actually signed, rather than the data itself. The public key corresponding to the private key used to compute a digital signature can then be used to obtain the hash of the data from the signature. This hash is then compared with the hash computed by the party that is verifying the signature. If they match, the data is considered authentic.

DSA is usually used in combination with Diffie-Hellman. Two parties in a conversation can exchange DSA public keys before the conversation begins and use the DSA keys to authenticate the communication of Diffie-Hellman parameters and keys. Diffie-Hellman with DSA provides authentication and secrecy, and when using the shared secret resulting from the Diffie-Hellman exchange as a key, a symmetric cipher can then be used for encryption.

In Example 9-19, we generate a new set of DSA parameters and write them to the dsaparam.pem file. The length of the prime and generator parameters in this example will be 1024 bits.

Example 9-19 Generating new DSA parameters

```
openssl dsaparam -out dsaparam.pem 1024
Generating DSA parameters, 1024 bit long prime
This could take some time
....+.........+.................+.+.................+.......+...........+++++
+++++++++++++++++++++++++++++++++++++++++++++*
...+.+.................+.+.........++++++++++++++++++++++++++++++++++++++++++++
+++++++++*
```

In Example 9-20, we generate a new DSA private key using the parameters from the dsaparam.pem file, encrypt the newly generated private key with the 3DES cipher, and write the result out to the dsaprivatekey.pem file.

Example 9-20 Generating new DSA parameters from a file

```
openssl gendsa -out dsaprivatekey.pem -des3 dsaparam.pem
warning, not much extra random data, consider using the -rand option
Generating DSA key, 1024 bits
Enter PEM pass phrase:
Verifying password - Enter PEM pass phrase:
```

In Example 9-21 we compute the public key that correspond to the private key create in Example 9-20 on page 211 and write the public key into the file dsapublickey.pem

Example 9-21 Generating the public key of a generated private key

```
# openssl dsa -in dsaprivatekey.pem -pubout -out dsapublickey.pem
read DSA key
Enter PEM pass phrase:
writing DSA key
# cat dsapublickey.pem
-----BEGIN PUBLIC KEY-----
MIIBtjCCASsGByqGSM44BAEwggEeAoGBAJj/uIO8xdAXJoMWZfoW7CV5WRmb4cCY
RO2xww2pmp9Ag/kPGq/OFbLiPObWRufGWR32rVMf/2oNPQ3HO1DFiry3wDHRAUPW
kvmqwvhOCZrZLDoLrGStXoS4/qCzSOzt24n2ZYA/RFJdmPuNDp37IxPHHSElPIXo
ms1DfYJsAmODAhUA12LwpBGL7jiClAMIyZzR7AQf6KOCgYAP5OBnGpocGpEadl+R
fRVUrEuhp6K6vkXJzNi6HzBXaUOLh9q4OciAaj3ZZFPuEYSlJVVOtN3Y5egHH3MU
h7DGwcpPOAO4+RFKEXecwH6awObHuYVAyNhG7GsvzQSzIrPFIOZ5KyU11TRZyx3c
FpqT/8PMOdrcZg/pXI38RBrVTwOBhAACgYAzfDvI199ne5aS47h2WRms43G9MyHK
2KmAw368AHdvgwO9DJklMFn+VuNhfg63M8En9JoEsQwroFcJbAQOOEqGr2oMDGwr
/PBKgQV+6TUDv6X6gN5hHRLip6zJtnC2dY1im4LPv6pp6kUYJVdJIMvLZhcz9CcM
dfehKcHCeeddJQ==
-----END PUBLIC KEY-----
```

In Example 9-22, we change the password of the private key storage in the dsaprivatekey.pem file. The old password was chile and the new one is brazil.

Example 9-22 Changing the password of a private key

```
openssl dsa -in dsaprivatekey.pem -out dsaprivatekey.pem -des3 -passin
pass:chile -passout pass:brazil
read DSA key
writing DSA key
```

RSA

RSA is the most popular public key algorithm in use. It is named after its creators, Ron Rivest, Adi Shamir, and Leonard Adleman. One of the reasons RSA is so popular is that it provides authentication, secrecy and encryption all in one package.

RSA, unlike Diffie-Hellman and DAS, does not require parameters to be generated before keys can be generated, which reduces the amount of work that is necessary to generate keys, and authenticate and encrypt communications.

In Example 9-23 on page 213, we generate a 2048-bit RSA private key, encrypt it using 3DES and a password of `brazil`, and write the output to the rsaprivatekey.pem file.

Example 9-23 Generating a RSA private key

```
# openssl genrsa -out rsaprivatekey.pem -passout pass:brazil -des3 2048
warning, not much extra random data, consider using the -rand option
Generating RSA private key, 2048 bit long modulus
.....................................................+++
.....................................................................
.........+++
e is 65537 (0x10001)
# cat rsaprivatekey.pem
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,AA623BEC2B00B630

DTrZqGrk85aQgO5iIy/VvCtPdPExAUA+5c4yh8Laz+XI+XbunRS4Lq58hfgOxp5m
UdeROOFiWJH+7X2mSM9/mOPHpmzLx1A9bhUJO286GnzyxnOPf9hU8LQVnpRpOF/L
tIFu1fOMb97TeuFpZMSndqQ8JZj765+Dh/r8HL8ayB/ZwGVSVZCbiPzOJoTcpOOA
3HIcMSHnkN8Hf1RLrp8vELjU4w2A1tL3/oOucWeFlxxtOwtW4K3TSIFSU8uTNske
VJwupW9ObQGHBbls1366lFc3TIRoDuOPBOcTvODoxmuYyjdv/CsOMwV4ot8FEqul
TpkPsF+Ex/Y/D+ZI9EVy9vkquBQIB94vtgHWeEOuTiTzD75UsJPPkjUlGroAHZHk
yXJGJFgyeCFVkv+SZDtn4EVnrxj/sPBqlqRktfPwZYm2Cv81yHWJVQqkmf4wTG1k
KUXykjECSnmFX3EqPXi9lK4e5lrGiSclk8aRBIhuIXvtUw/dwj3+cDhOeGAcG9Hs
Dxs4UeINEdfjkIhnFnYZMnWdwthmia3P/P9XCf3G/LTSQ2lK9xbr9fOO1MUI5eyl
E3cu5i9XM5rEZW1s7TYQRGBg728vD3OUK8j+UiUO5lwqGp2loKwatKeFUrhsNzOt
rrNTd3rx/MIK6IatWHfhu4SGVxtSRTd8krGzz9x3/ivuzFD8TE3w5hEUMWglOuMZ
lCMej4vY95kcQnFwDntekpkqzkkOIZXACfBrhEHVOAJ6Fge827YpdZpp6mz3zXip
JarowVa2xLOkDOoVIuLBpzszPCc9GCh6iiQ7FdHJW2TmlNo+q+m/+mXnk8OdZFIE
3N1vbE5Kq5yhUoMlP8WXKeYvc+y2I92RMQuIROnoA+bz2EXnLqxRte7MyEgVJOvh
kHIRhVP8Yw7+/OS4N7clkYls2tSR4H36RuRNU6TVLNTrhtB+R/kowSNgOWGuyCo2
eAyzlz7l66OOPkGs/6Sw4qXOaI6LcehOneWW4EO2GZfgHnBHoKIXUWLw4RlZdDve
r6HBQfXKTk75hU2IawZl8Ljj5dPuQYG5vj/BT4jhgbsmb5uOw3Wk2dUPPeU5yDgv
SNA49H4MwekTmLNXXY6ouQgOvJueRV+FXwJ4HFUGTFK7PYWg4M2EpPnqtlOOtD/p
cCMpOO3D1NrCfYp3T8BTS/iYfLJ/1TJk+Csa/KaP/9oqQOWZopeC+7ClsILADT55
xx9LuQaSpIakTzwlgM76IZ46TPm4IePYY9UxZAyt+iebyKBMcKGChY23AalMC4Ms
Ym6Mv8CtO+CldWrJuPY5PMqptljI2/ZaN2v8a98eDRxlaMp7b5hOuKcvPxRe3WUC
XoaJbOmRcLZoghaAU7Q/AjJwKu3nU/dtEzG/OwO58zKIg3n7RRJ3zXsiH2+nexWE
uxagRcTVGOdJkpc/OhCduYrWm9tX+cZsH1TLabk3rpV1fbOuewZLeH2ZAYWvA9ps
6TyxMpNbzXBI5Wf2hlFwuwxsnheDZzOHZcoHF3dCTqIY5r9FkbOD7O/Hba5+ROZp
wwSSOmu5/r+aUyvwA2oBNgDI2L/eT2hjlfrkdK1iLdd9pzLdEI1Y6Xz/Ir41SRi4
-----END RSA PRIVATE KEY-----
```

In Example 9-24 on page 214, we read the file generated in the Example 9-23, decrypt it using the password brazil, and write the corresponding public key to the file rsapublickey.pem.

Example 9-24 Generating a RSA public key from a private key

```
# openssl rsa -in rsaprivatekey.pem -passin pass:brazil -pubout -out
rsapublickey.pem
read RSA key
writing RSA key
```

In Example 9-25, we encrypt a file named redbook.txt using the RSA public key, generated in Example 9-24, and write the result to the redbookcipher.txt file.

Example 9-25 Encrypting a file using a given public key

```
# openssl rsautl -encrypt -pubin -inkey rsapublickey.pem -in redbook.txt -out
redbookcipher.txt
```

Example 9-26 shows how to decrypt the redbookcipher.txt file created in Example 9-25.

Example 9-26 Decrypting a file using a given public key

```
# openssl rsautl -decrypt -inkey rsaprivatekey.pem -in redbookcipher.txt -out
redbookplain.txt
```

Certification authority

OpenSSL provides all the functionality required to set up a minimal CA that can be used in a small organization.

In this section, we will cover how to configure a CA. The steps are the same for Linux and AIX.

We must choose a location for all of our CA's files, for example, we will use /opt/redbookca as the root directory for our CA. Besides the directory, we need to create three other files. The first file will keep the last serial number that was used to issue a certificate; this file is named serial. The second file is a database of sorts that keeps track of the certificates that have been issued by the CA; this file is named index.txt (see Example 9-27). The last file is the configuration file is openssl.cnf; we suggest a configuration file for this demo CA in Example 9-28 on page 215.

Example 9-27 Creating a CA configuration directory and files

```
# mkdir /opt/redbookca
# cd /opt/redbookca
# mkdir certs private
# chmod g-rwx,o-rwx private
# echo '01' > serial
```

```
# touch index.txt
```

Example 9-28 RedbookCA configuration file

```
[ ca ]
default_ca = redbookca

[ exampleca ]
dir                     = /opt/redbookca
certificate             = $dir/cacert.pem
database                = $dir/index.txt
new_certs_dir           = $dir/certs
private_key             = $dir/private/cakey.pem
serial                  = $dir/serial

default_crl_days        = 7
default_days            = 365
default_md              = md5

policy                  = redbookca_policy
x509_extensions         = certificate_extensions

[ redbookca_policy ]
commonName              = supplied
stateOrProvinceName     = supplied
countryName             = supplied
emailAddress            = supplied
organizationName        = supplied
organizationalUnitName  = optional

[ certificate_extensions ]
basicConstraints        = CA:false

[ req ]
default_bits            = 2048
default_keyfile         = /opt/redbookca/private/cakey.pem
default_md              = md5

prompt                  = no
distinguished_name      = root_ca_distinguised_name

x509_extensions         = root_ca_extensions

[ root_ca_distinguised_name ]
commonName              = redbook CA
stateOrProvinceName     = Texas
countryName             = US
emailAddress            = ca@austin.ibm.com
organizationName        = IBM
```

```
organizationalUnitName  = International Technical Support Organization

[ root_ca_extensions ]
basicConstraints        = CA:true
```

We have create a new configuration file, but we need to tell OpenSSL where to find it. To do that, we can issue the following commands:

```
# OPENSSL_CONF=/opt/redbookca/openssl.cnf
# export OPENSSL_CONF
```

Now we can generate our self-sign root certificate and generate a new key pair to go along with it. In Example 9-29, we show you how to accomplish that task.

Example 9-29 Generating a self-signed root certificate

```
#openssl req -x509 -newkey rsa -out cacert.pem -outform PEM
Using configuration from /opt/redbookca/openssl.cnf
Generating a 2048 bit RSa private key
.........................+++
.......................+++
writing new private key to '/opt/redbookca/private/cakey.pem'
Enter PEM pas phrase:
Verifying password - Enter PEM pass phrase:
-----

# openssl x509 -in cacert.pem -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 0 (0x0)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: CN=redbook CA, ST=Texas, C=US/Email=ca@austin.ibm.com, O=IBM,
OU=International Technical Support Organization
        Validity
            Not Before: Nov 12 19:08:00 2002 GMT
            Not After : Dec 12 19:08:00 2002 GMT
        Subject: CN=redbook CA, ST=Texas, C=US/Email=ca@austin.ibm.com, O=IBM,
OU=International Technical Support Organization
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (2048 bit)
                Modulus (2048 bit):
                    00:d1:37:19:08:07:84:ed:25:1d:3c:04:48:7e:d0:
                    c9:d3:4e:56:6e:c2:16:20:7e:8c:a5:bd:f8:e9:69:
                    32:00:ba:16:60:99:85:1d:06:cd:fd:e0:e0:64:f1:
                    fd:97:c6:fa:55:5d:2f:50:4b:c1:3a:9f:30:05:0f:
                    fb:70:0a:f5:a2:27:37:cd:14:66:57:ac:a9:f4:0b:
                    6f:9a:1c:5c:9e:8f:82:6c:3d:1e:0a:71:16:78:27:
```

```
                    fc:71:ea:dd:ec:21:50:f7:5f:10:18:fb:bf:81:ed:
                    31:87:3a:e2:78:ac:ec:a1:ce:36:56:02:6f:60:81:
                    9d:b6:76:a7:f1:42:1d:86:df:c3:23:9c:d6:52:4a:
                    bc:f2:53:77:4e:a6:80:9e:97:ac:4e:ba:fe:ef:9d:
                    86:b8:20:62:ff:3a:f8:cc:76:6f:2e:b8:2a:61:7e:
                    1c:cf:93:09:e8:6e:49:51:01:c1:7d:8e:4f:19:e4:
                    f6:83:2b:81:bd:b0:aa:34:14:20:fc:dc:7c:f7:aa:
                    a2:2e:8f:f9:f3:7a:77:d7:61:71:70:58:67:7c:3c:
                    32:35:ee:db:95:d8:9f:cf:d0:f7:be:4f:09:0a:05:
                    fe:18:63:0f:43:66:f5:17:5e:28:9d:dc:d3:95:f8:
                    6b:d8:c1:78:4f:08:30:1e:4b:58:4d:02:46:6f:a0:
                    8c:67
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:TRUE
    Signature Algorithm: md5WithRSAEncryption
        7f:29:ec:9c:ca:fd:36:57:a3:15:b9:33:4d:4d:21:22:49:5f:
        b3:11:79:70:8b:75:05:b2:42:cc:7d:e0:78:0b:4a:16:cb:ea:
        91:5e:4c:a7:a4:12:22:09:65:9b:e8:7a:af:4a:e4:61:47:9c:
        a6:77:15:56:e6:25:47:56:a7:9d:8c:67:e3:25:60:c8:c9:8d:
        9f:0d:67:70:f6:7e:02:33:35:56:18:54:62:f5:4b:5f:59:1c:
        45:a4:ca:87:18:9a:a6:a6:1f:2f:99:bf:3f:d1:c4:07:be:d5:
        0a:1b:30:04:3d:3d:45:cf:4e:0f:c0:21:3b:e3:59:28:3c:74:
        a2:23:5f:7c:7b:4b:14:e9:d9:73:c4:9b:1b:04:21:52:9f:59:
        12:f5:3e:74:f5:90:45:48:3e:37:8b:a2:68:0d:88:27:3d:5e:
        c2:1c:c0:cd:ac:c9:a7:db:3c:62:c8:f9:7a:99:c3:e4:d2:8a:
        87:c4:7a:7e:c0:34:17:c8:5a:dd:55:95:47:55:a0:e1:f9:94:
        33:28:73:ef:26:99:0c:69:ce:78:a8:ef:c3:88:1a:e1:dc:a3:
        d6:f9:a6:a8:95:d5:66:ec:19:38:ed:59:0a:ba:b0:c5:4a:d0:
        de:de:ae:e5:69:5c:b7:c8:3b:b3:47:ec:ee:3c:41:6c:73:91
d7:60:53:fe
```

Now that our CA is configured, we can start issuing some certificates. To create a certificate request, start with a clean shell without the OPENSSL_CONF environment variable set so the default configuration file will be used. In Example 9-30, we show how to request a certificate.

Example 9-30 Generating a certificate request

```
# openssl req -newkey rsa:1024 -keyout testkey.pem -keyform PEM -out
testreq.pem -outform PEM
Using configuration from /var/ssl/openssl.cnf
Generating a 1024 bit RSA private key
...................++++++
.....................++++++
writing new private key to 'testkey.pem'
Enter PEM pass phrase:
```

```
Verifying password - Enter PEM pass phrase:
Verify failure
Enter PEM pass phrase:
Verifying password - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [US]:
State or Province Name (full name) [Texas]:
Locality Name (eg, city) [Austin]:
Organization Name (eg, company) [IBM]:
Organizational Unit Name (eg, section) [International Technical Support
Organization]:
srvr80j.itsc.austin.ibm.com []:
admin@srvr80j.itsc.austin.ibm.com []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:moon river lone star
An optional company name []:redbook-R-US
```

After the execution of these commands we create two files: testreq.pem and teskey.pem. The testreq.pem file, contains the certificate request and the testkey.pem file contains the private key that matches the public key embedded in the certificate request.

Now that we have a certificate request, we can use our CA to issue a certificate. First of all, set the environment variable OPENSSL_CONF and issue the commands shown in Example 9-31.

Example 9-31 Issuing a certificate from a certificate request

```
# export OPENSSL_CONF=/opt/redbookca/openssl.cnf
# openssl ca -in testreq.pem
Using configuration from /opt/redbookca/openssl.cnf
Enter PEM pass phrase:
Check that the request matches the signature
Signature ok
The Subjects Distinguished Name is as follows
countryName           :PRINTABLE:'US'
stateOrProvinceName   :PRINTABLE:'Texas'
localityName          :PRINTABLE:'Austin'
organizationName      :PRINTABLE:'IBM'
```

```
organizationalUnitName:PRINTABLE:'International Technical Support Organization'
commonName            :PRINTABLE:'srvr80j.itsc.austin.ibm.com'
emailAddress          :IA5STRING:'admin@srvr80j.itsc.austin.ibm.com'
Certificate is to be certified until Nov 12 19:31:38 2003 GMT (365 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1 (0x1)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: CN=redbook CA, ST=Texas, C=US/Email=ca@austin.ibm.com, O=IBM,
OU=International Technical Support Organization
        Validity
            Not Before: Nov 12 19:31:38 2002 GMT
            Not After : Nov 12 19:31:38 2003 GMT
        Subject: CN=srvr80j.itsc.austin.ibm.com, ST=Texas,
C=US/Email=admin@srvr80j.itsc.austin.ibm.com, O=IBM, OU=International Technical
Support Organization
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:d7:03:f9:f9:e5:e7:a3:48:05:c6:0c:d4:e5:59:
                    b8:ff:26:8f:6c:f8:da:80:c8:d8:10:91:0e:6b:bd:
                    66:98:51:c4:60:ad:59:7e:8b:d8:ba:8e:b0:66:00:
                    2a:39:0c:01:59:aa:7a:72:4e:37:7b:b1:9b:0f:b2:
                    f8:a8:43:58:86:d6:db:44:dd:4b:18:6a:ba:f2:db:
                    04:68:83:5e:6a:48:7e:df:42:ba:e1:4a:d9:76:42:
                    f0:af:5d:51:c9:a9:f7:d8:30:e7:f7:93:b7:23:dd:
                    5a:bd:89:26:d9:cd:32:32:d0:e1:6b:7e:f7:e6:c5:
                    12:78:53:a7:b0:9e:0d:fb:3f
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
    Signature Algorithm: md5WithRSAEncryption
        c6:20:01:8e:c7:f4:e1:21:ab:51:57:64:f6:b0:d2:2d:65:b6:
        c1:63:44:5f:96:df:61:9c:2c:ab:75:2b:bf:7d:1f:09:5e:ab:
        b5:4f:3c:ec:5b:9e:42:7a:a9:bc:11:c9:18:e4:e3:b7:0c:8f:
        cf:86:91:41:fc:0a:d8:a1:82:d4:51:db:65:e1:ac:3d:9b:c3:
        65:54:40:66:e5:81:40:ef:64:ef:66:b2:71:ec:00:3f:f4:5d:
        56:0a:21:26:ea:60:ff:71:a8:18:18:35:0a:9b:68:95:64:b8:
        47:d1:80:35:ff:27:9e:0e:62:39:a1:d2:63:c2:18:b3:73:a5:
        bd:67:45:00:5c:cf:3e:12:6f:59:20:2f:86:55:2e:e1:e3:d7:
        41:3c:5e:82:2e:c3:15:fb:e2:3c:d7:d4:89:10:cd:ad:77:a6:
```

```
        ea:c2:f7:e0:67:50:34:5d:ab:a9:89:83:09:a6:17:47:94:af:
        cd:38:4e:b7:59:db:80:ee:15:d3:82:27:ce:49:b1:a8:10:d2:
        1b:01:d8:f3:bf:e8:0e:b3:f5:03:d9:f4:06:b5:85:c0:5d:fe:
        30:1e:50:ff:4a:17:82:56:fc:ae:73:8d:59:8c:de:8b:1d:a5:
        cc:00:4a:81:fc:a4:d7:17:7b:30:42:ba:d9:82:bf:f8:84:5d:
        2e:29:3c:68
-----BEGIN CERTIFICATE-----
MIIDWTCCAkGgAwIBAgIBATANBgkqhkiG9w0BAQQFADCBmTETMBEGA1UEAxMKUmVk
Ym9vayBDQTEOMAwGA1UECBMFVGV4YXMxCzAJBgNVBAYTAlVTMSAwHgYJKoZIhvcN
AQkBFhFjYUBhdXN0aW4uaWJtLmNvbTEMMAoGA1UEChMDSUJNMTUwMwYDVQQLEyxJ
bnRlcm5hdGlvbmFsIFRlY2huaWNhbCBTdXBwb3J0IE9yZ2FuaXphdGlvbjAeFw0w
MjExMTIxOTMxMzhaFw0wMzExMTIxOTMxMzhaMIG6MSQwIgYDVQQDExtzcnZyODBq
LmlOc2MuYXVzdGluLmlibS5jb2OxDjAMBgNVBAgTBVRleGFzMQswCQYDVQQGEwJV
UzEwMC4GCSqGSIb3DQEJARYhYWRtaW5Ac3J2cjgwai5pdHNjLmF1c3Rpbi5pYmOu
Y29tMQwwCgYDVQQKEwNJQkOxNTAzBgNVBAsTLEludGVybmFOaW9uYWwgVGVjaG5p
Y2FsIFN1cHBvcnQgT3JnYW5pemFOaW9uMIGfMAOGCSqGSIb3DQEBAQUAA4GNADCB
iQKBgQDXA/n55eejSAXGDNTlWbj/Jo9s+NqAyNgQkQ5rvWaYUcRgrVl+i9i6jrBm
ACo5DAFZqnpyTjd7sZsPsvioQ1iG1ttE3UsYarry2wRog15qSH7fQrrhStl2QvCv
XVHJqffYMOf3k7cj3Vq9iSbZzTIyOOFrfvfmxRJ4U6ewng37PwIDAQABowOwCzAJ
BgNVHRMEAjAAMAOGCSqGSIb3DQEBBAUAA4IBAQDGIAGOx/ThIatRV2T2sNItZbbB
YORflt9hnCyrdSu/fR8JXqu1TzzsW55Ceqm8EckY5OO3DI/PhpFB/ArYoYLUUdtl
4aw9m8NlVEBm5YFA72TvZrJx7AA/9F1WCiEm6mD/cagYGDUKm2iVZLhHOYA1/yee
DmI5odJjwhizc6W9ZOUAXM8+Em9ZIC+GVS7h49dBPF6CLsMV++I819SJEM2td6bq
wvfgZ1AOXaupiYMJphdHlK/NOE63WduA7hXTgifOSbGoENIbAdjzv+gOs/UD2fQG
tYXAXf4wHlD/SheCVvyuc41ZjN6LHaXMAEqB/KTXF3swQrrZgr/4hFOuKTxo
-----END CERTIFICATE-----
Data Base Updated
```

The passphrase that OpenSSL asks for is the passphrase for the CA's private key. The private key will be used to sign the new certificate. After that, OpenSSL prompts you for confirmation to sign the certificate. The final prompt is to confirm whether the certificates should be committed to the CA's database. So the new certificate is written to stdout. The resulting certificate is also written to the directory that we specified in our configuration file with the new_certs_dir key.

Using OpenSSL, we can revoke the certificate created in Example 9-31 on page 218. To revoke a certificate, all that we need is a copy of the certificate to be revoked. In our redbook CA, a copy of each certificate issue is keep in the /opt/redbookca/certs directory. In Example 9-32, we show the steps needed to revoke a certificate.

Example 9-32 Revoking a certificate

```
# cd /opt/redbookca/
# cp certs/01.pem testcert.pem
# openssl ca -revoke testcert.pem
Using configuration from /opt/redbookca/openssl.cnf
Enter PEM pass phrase:
```

```
Revoking Certificate 01.
Data Base Updated
```

Now that we have a certificate revoked, we can generate the revocation list. In Example 9-33, we show how to generate it.

Example 9-33 Generating a revocation list

```
# openssl ca -gencrl -out redbookca.crl
Using configuration from /opt/redbookca/openssl.cnf
Enter PEM pass phrase:
# openssl crl -in redbookca.crl -text -noout
Certificate Revocation List (CRL):
        Version 1 (0x0)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: /CN=redbook
CA/ST=Texas/C=US/Email=ca@austin.ibm.com/O=IBM/OU=International Technical
Support Organization
        Last Update: Nov 12 19:59:35 2002 GMT
        Next Update: Nov 19 19:59:35 2002 GMT
Revoked Certificates:
    Serial Number: 01
        Revocation Date: Nov 12 19:50:14 2002 GMT
    Signature Algorithm: md5WithRSAEncryption
        bd:31:53:b4:a3:8b:33:c2:05:83:8d:b2:58:21:e1:24:0b:d7:
        e2:3c:de:7c:3c:12:06:f7:5f:ea:88:a3:34:41:ac:da:75:e3:
        3f:43:27:d2:d4:45:e5:73:04:fe:f6:f8:eb:6a:70:ff:a8:2c:
        59:18:f0:e0:44:94:03:ee:26:d2:88:87:f3:a9:e1:8e:8c:90:
        0e:b9:26:bb:84:db:f3:7a:6f:42:ce:e1:9b:f4:68:95:11:98:
        a8:9d:4f:0c:57:98:e6:a3:de:5d:31:55:b3:c9:30:ee:1c:c2:
        eb:7c:78:56:7c:ec:47:b6:a0:1d:cf:9b:61:79:2c:f4:5d:12:
        02:c7:de:90:3e:db:5c:b0:7d:98:ee:84:35:89:54:24:df:3f:
        c8:14:54:84:a2:b0:c8:57:18:52:a8:78:7a:6e:a9:32:61:c3:
        87:12:64:3e:67:f3:0f:99:97:65:04:1c:68:c2:82:63:0c:77:
        1e:e5:be:8d:20:3a:7d:57:bd:aa:77:48:3d:1b:38:e8:e0:4a:
        52:06:5d:d3:3f:08:71:41:82:e7:3a:77:62:b1:f9:a5:10:bc:
        fb:77:43:2e:73:f9:19:15:67:8a:1f:61:9d:78:72:33:54:51:
        a9:7b:ba:d0:db:26:63:be:a8:ea:c8:05:0b:35:11:b5:bb:8e:
        bc:40:53:1a
# openssl crl -in redbookca.crl -noout -CAfile cacert.pem
verify OK
```

SSL and TLS

You can use OpenSSL to test connections over SSL/TLS. If you want to test a client, use the SSL server. To use the server, use the s_server command. To test the types of protocols that you client use, issue the command:

```
# openssl s_server -accept <port> [-ssl2|-ssl3|-tls1|-notls1|nossl3]
```

To test a connection to a SSL server, you can run the SSL client. We can use the command:

```
# openssl s_client -connect <host:portnumber>
[-ssl2|-ssl3|tls1|-no_ssl2|-no_ssl3|-no_tls1]
```

9.2.2 OpenSSH

OpenSSH is an security tool used to establish a security connection between hosts. It is an open implementation of SSH that supports SSH1 and SSH2 protocols.

OpenSSH is used as a secure and encrypted replacement for the vulnerable telnet and r commands, such **`rlogin`**, **`rsh`**, **`rcp`**, and **`rdist`**. OpenSSH can authenticate user using RSA and DSA authentication protocols, which are based on a pair of complementary numerical keys.

You can find more details about OpenSSH in Chapter 4, “Secure network connections to AIX” in *Managing AIX Server Farms*, SG24-6606.

9.2.3 tcp_wrapper

tcp_wrapper is one of the first tools installed by system administrators. The tcp_wrapper, written by Wietse Venema, is a utility that can by easily installed; it will be able to intercept, filter, and enhance the logs details for the services controlled by the inetd daemon.

This tool allows tight control over TCP connections requested in a system. The activation logic is simple: When the inetd receives a request for a service, it executes a control program (tcpd) instead of the requested service. The tcpd will check the connection’s origin against the rules defined and then take action. This action could be a double check of the origin IP:

```
IP -> name
name -> IP
```

The tcpd program also compares the request’s origin address against an access control list. It uses the identd daemon to identify the request origin. It can also registry the results by using syslogd.

Installing tcp_wrapper on Linux

On Red Hat Linux Version 8.0, the tcp_wrapper package is located on installation disc 1. You can install it by issuing the following command:

```
# rpm -ivh tcp_wrappers-7.6-23.i386.rpm
warning: tcp_wrappers-7.6-23.i386.rpm: V3 DSA signature: NOKEY, key ID db42a60e
Preparing...                ########################################### [100%]
```

```
1:tcp_wrappers          ######################################### [100%]
```

To learn how to install tcp_wrapper on AIX and how to configure it, please refer to Chapter 4, “Secure network connections to AIX” in *Managing AIX Server Farms*, SG24-6606.

10

System administration applications

In this chapter, we provide a general overview of the system administration applications available for AIX and Linux system environments. We also provide information on the different features of each application, how to install it, and its usage.

This chapter contains the following:

- Web-based System Manager
- Web-based System Manager Client for Linux
- Webmin on AIX and Linux

10.1 Web-based System Manager

Web-based System Manager is a system management application for administering computers. It is installed by default on graphical systems, and has been substantially enhanced for AIX 5L. It is a client-server application that gives the user a powerful interface to manage UNIX systems.

Web-based System Manager features a system management console for administering multiple hosts. It uses its graphical interface to enable the user to access and manage multiple remote machines. A plug-in architecture makes it easier to extend the suite. In addition, Web-based System Manager supports dynamic monitoring and administrator notification of system events.

The user interface, as shown in Figure 10-1,has improved noticeably and the console provides a convenient interface for managing multiple AIX hosts. Web-Based System Manager is intended to significantly increase the object-orientation of system management of AIX.

For more information on the Web-based System Manager for AIX 5L, refer to *AIX 5L Version 5.1 Web-based System Manager Administration Guide.*

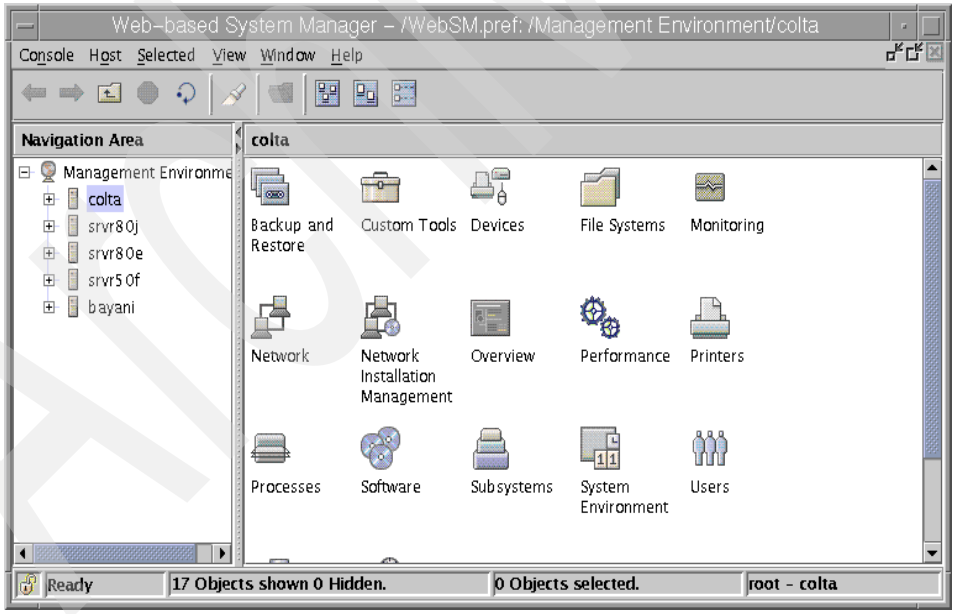

Figure 10-1 Web-based System Manager user interface

10.1.1 Web-based System Manager architecture

The Web-based System Manager enables a system administrator to manage AIX machines either locally from a graphics terminal or remotely from a PC client, a Linux client, or an AIX client. Information is entered through the Graphical User Interface (GUI) components on the client side. The information is then sent over the network to the Web-based System Manager server, which runs the necessary commands to perform the required action.

The Web-based System Manager is implemented using the Java programming language. The implementation of Web-based System Manager in Java provides:

- Cross-platform portability: Any client platform with a Java 1.3-enabled Web browser is able to run a Web-based System Manager client object.
- Distributed processing: A Web-based System Manager client is able to issue commands to AIX machines remotely through the network.
- Multiple launch points: The Web-based System Manager can be launched either in a Java Application Mode locally within the machine to manage both a local and remote system, in a Java Applet mode through a system with a Web browser with Java 1.3 installed, and in Linux/Windows PC Client mode, where client code is downloaded from an AIX host.

10.1.2 Installing Web-based System Manager

Using Web-based System Manager effectively requires that the client computer have at least the following characteristics:

- Graphics display
- 50 MB free disk space
- 256 MB of memory
- 30 MHz CPU

To use a PC to run Web-based System Manager in applet mode, it should have a processor speed of at least 500 MHz.

To use Web-based System Manager, it must be installed on the client used to run it and on any managed machines. For machines with the AIX 5L operating system, Web-based System Manager is already installed by default.

To verify this, enter the following command:

```
# lslpp -h sysmgt.websm.framework
```

If Web-based System Manager is not installed, you will see a message that is similar to the following:

```
lslpp: Fileset sysmgt.websm.framework not installed
```

If Web-based System Manager is installed, you will see a message that is similar to Example 10-1.

Example 10-1 lslpp -h output

```
Fileset                      Level  State      Description
  ----------------------------------------------------------------------------
Path: /usr/lib/objrepos
  sysmgt.websm.framework     5.2.0.0  COMMITTED  Web-based System Manager
                                                 Client/Server Support

Path: /etc/objrepos
  sysmgt.websm.framework     5.2.0.0  COMMITTED  Web-based System Manager
                                                 Client/Server Support
```

If you do not have the sysmgt.websm.framework fileset installed, use the operating system installation CDs. To access the installation tools, use **smit installp** command.

Choose **Install and Update from ALL Available Software**. Then fill in the appropriate fields, as shown in Example 10-2.

Example 10-2 smit installp screen

```
                 Install and Update from ALL Available Software

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                        [Entry Fields]
* INPUT device / directory for software                 /dev/cd0
* SOFTWARE to install                                  [sysmgt.websm.framewor> +
  PREVIEW only? (install operation will NOT occur)      no                      +
  COMMIT software updates?                              yes                     +
  SAVE replaced files?                                  no                      +
  AUTOMATICALLY install requisite software?             yes                     +
  EXTEND file systems if space needed?                  yes                     +
  OVERWRITE same or newer versions?                     no                      +
  VERIFY install and check file sizes?                  no                      +
  DETAILED output?                                      no                      +
  Process multiple volumes?                             yes                     +
  ACCEPT new license agreements?                        no                      +
  Preview new LICENSE agreements?                       no                      +
```

```
F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Alternatively, you can use the following command to install the fileset:

```
# /usr/lib/instl/sm_inst installp_cmd -a -d /dev/cd0 -f \
sysmgt.websm.framework -c -N -g -X
```

10.1.3 Modes of operation

Like in previous releases, the Web-based System Manager can be launched from a variety of launch points. It can be configured to run in a variety of operating modes. The modes of operation are as follows:

- Stand-alone application mode
- Client-Server mode
- Applet mode
- PC Client mode

Stand-alone application mode

All AIX systems with a Graphical User Interface (GUI) can use this mode to perform local tasks. This mode is enabled by default. No configuration is necessary to run Web-based System Manager in the stand-alone application mode. It can be launched in two ways:

- Running the Java application mode by running the **wsm** command on the AIX command line on the system being managed.
- Through the Management Console icon on the CDE desktop Application Manager.

To start Web-based System Manager from the Common Desktop Environment (CDE), do the following:

1. Select the **Application Manager** icon in the CDE front panel.
2. Select the **System_Admin** icon.
3. Select the **System Manager** icon. Refer to Figure 10-2 on page 230.

The launch pad icon in the CDE application manager loads the Web-based System Manager environment with launch icons for all of the Web-based System Manager applications. Multiple applications can be started without needing to restart the Web-based System Manager environment each time.

Figure 10-2 Web-based System Manager icon on CDE user interface

Alternatively, to run Web-based System Manager from the command line, you can type the following command:

```
# wsm
```

The command line allows the Web-based System Manager to be in XWindows from an aixterm window or in the CDE desktop from a dtterm window.

Client-Server mode

When multiple hosts are set up to be managed from a single console, the Web-based System Manager operates in a client-server mode. In this mode, you can manage your local machine and also other AIX remote machines that have been configured for remote management. You can specify the machines you want to manage by adding them to the management environment.

The first machine contacted by the client acts as the managing host while the other hosts in the navigation area are managed hosts. You can also select a different host than your local machine to act as the managing machine. To do this, use the following command:

```
# wsm -<managing machine host>
```

The host you specify as managing machine host displays under the Navigation Area as the first host name under the list of hosts that can be managed. This host is also used to load the Web-based System Manager user preference profile ($HOME/WebSM.pref). Using the -host argument displays the console of the machine you are using, but uses the preferences file of the remote host you specify.

Note: Any AIX target host to be managed by Web-based System Manager must have the Web-based System Manager server installed and configured.

Applet mode

In applet or browser mode, the administrator can manage one or more AIX hosts remotely from the client platform's Web-browsers with Java 1.3. To access the console in this manner, an AIX host need only be configured with a Web-server (for example, IBM HTTP-Server, which is provided on the AIX Expansion Pack CDs). Once the Web-server is installed and configured, the host can serve the console to the client. The administrator simply enters the URL in the browser:

```
http://<hostname>/wsm.html
```

where hostname is the host name of the managing machine.

A Web page is then served to the browser that prompts the user for a user name and password. Once authenticated to the server, the console launches into a separate panel frame. In Web-based System Manager applet mode, the browser is used only for logging in and launching the console. Once running, the console is relatively independent of the browser.

> **Note:** In applet mode, it is only possible to manage a set of machines that have the same version of Web-based System Manger installed.

PC Client mode

The PC Client code is downloaded from an AIX host, then installed permanently on the PC. Because all the Java code is native on the PC, the startup time and performance are exceptionally good compared to the applet mode. However, the PC Client differs from the client-server mode such that the Windows system running PC Client is the managing machine but is not included in the Management Environment area.

10.2 Web-based System Manager Client for Linux

Support has been added to the Web-based System Manager Client for the Linux platform. Since the Web-based System Manager is a platform independent Java application, the Linux client is identical to the Web-based System Manager PC Client for Windows. It is supported on Red Hat Linux Version 7.2 or later. It allows you to remotely manage AIX systems and the Hardware Management Console (HMC) for the Regatta systems.

To install the Web-based System Manager Client for Linux over the network, an AIX 5L system (the managing machine) needs to be configured with a Web server (In this case, we installed the IBM HTTP Server).

To install Web-based System Manager Client for Linux, do the following steps:

1. On the AIX system, verify that sysmgt.websm.webaccess is installed. You can do this by running the command:

   ```
   # lslpp -L sysmgt.websm.webaccess
   ```

   If it is installed, the fileset will be listed. If it is not installed, refer to 10.1.2, “Installing Web-based System Manager” on page 227 for installation instructions.

2. Then, install IBM HTTP-Server (IHS) from the Expansion Pack CD using the command line:

   ```
   # installp -acY -d /dev/cd0 http_server.base
   ```

   Alternatively, you can also use **`smit installp`** for this task.

3. Start the Configuration Assistant wizard by running the command:

   ```
   # /usr/bin/configassist
   ```

   When the wizard starts, click the **Next** button.

4. From the task list, select **Configure a Web server to run Web-based System Manager in a browser** (see Figure 10-3) and click **Next**.

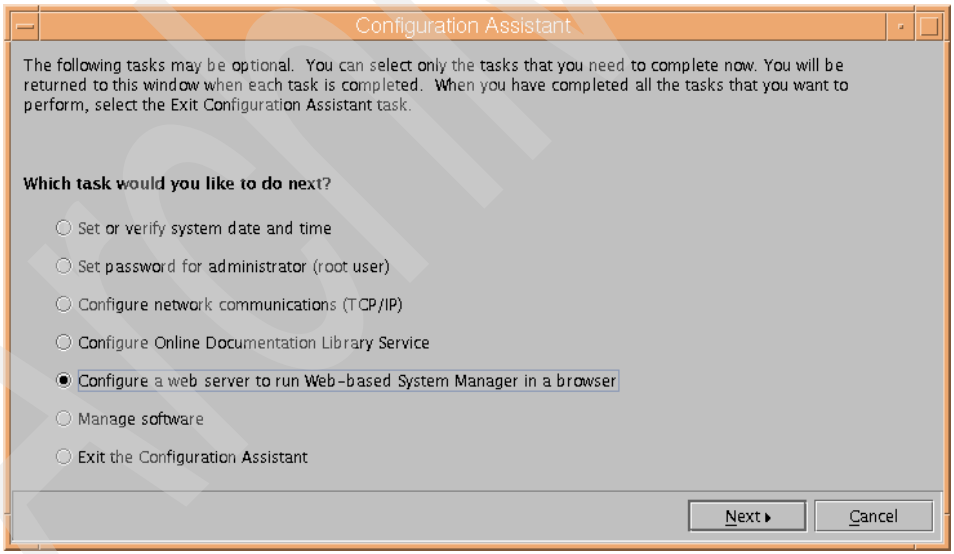

Figure 10-3 Configuration Assistant Wizard window

5. To accept the default values on the next dialog, click **Next** again.

6. On the Red Hat Linux system, launch a Web browser and connect to the previously configured Web server by specifying the fully qualified domain name or the IP address of the server in the following URL:

   `http://<managing machine>/remote_client.html`

7. On the Web page, click on the **Linux** link and save the wsmlinuxclient.exe file to a directory of your choice, for example, /tmp.

8. On the Linux command line, run the following commands:

```
# chmod +x /tmp/wsmlinuxclient.exe
# /tmp/wsmlinuxclient.exe
```

9. The InstallShield Wizard will start the setup of your Web-based system Manager Client, as shown in Figure 10-4.

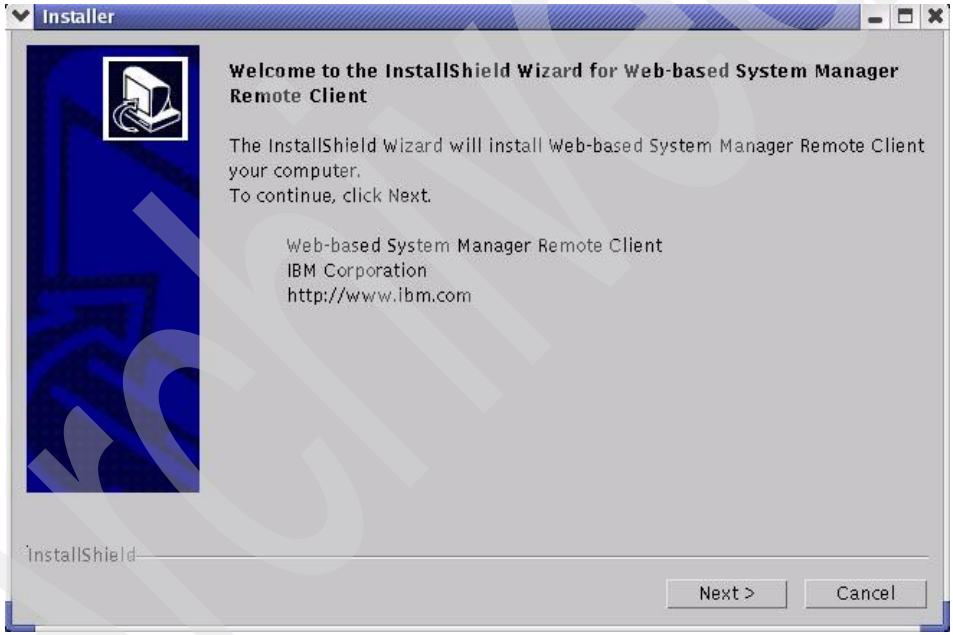

Figure 10-4 InstallShield Wizard for Web-based System Manager Remote Client

10. The Installation Wizard will lead you through the whole installation process. Fill out all the necessary fields, as shown in Figure 10-5 on page 234.

Figure 10-5 Installation of Web-based System Manager Remote Client

11. When the installation is finished, you can launch the Web-based System Manager Client by issuing the command:

```
# /opt/websm/bin/wsm
```

12. After running the **wsm** command, you will receive a login window, as shown in Figure 10-6.

Figure 10-6 Log on window for Web-based System Manager Client

13. Once you are logged in, Web-based System Manager will run and you are able to manage your AIX operating system from your Linux system, as shown in Figure 10-7.

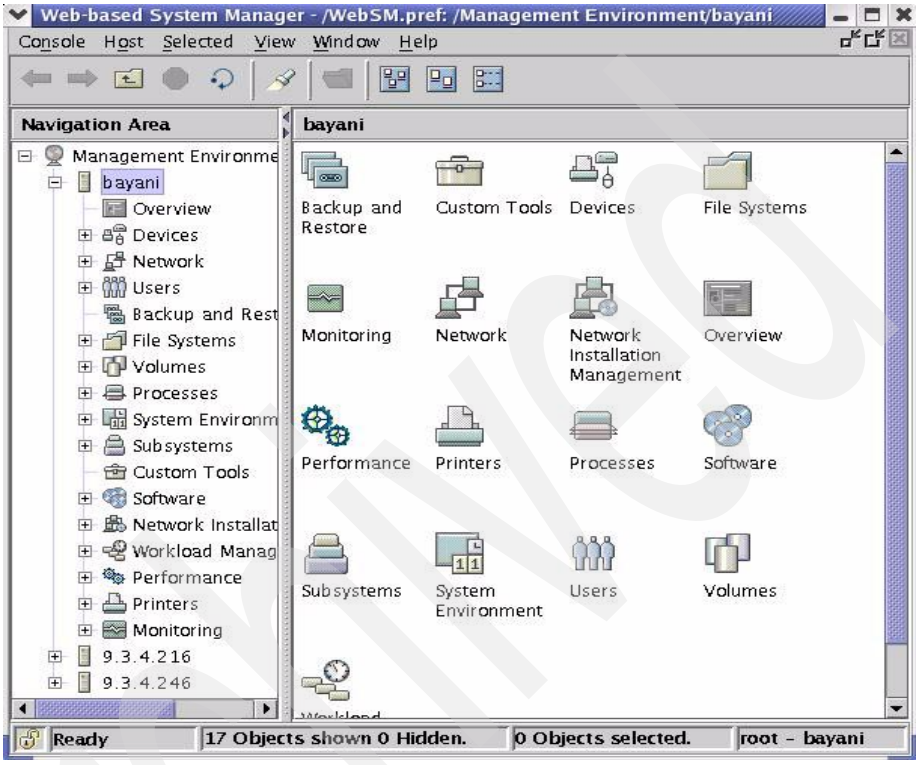

Figure 10-7 Web-based System Manager Client for Linux

10.2.1 Implementing a secure connection in server-client mode

In the Web-based System Manager secure operation, the managed machines are servers, and the managing users are the clients. Each Web-based System Manager server has its private key and a certificate of its public key signed by a Certificate Authority (CA) that is trusted by the Web-based System Manager clients. The private key and the server certificate are stored in the server's private key ring file. The Web-based System Manager client has a public key ring file that contains the certificates of the CAs that it trusts.

To further illustrate how the Web-based System Manager Security works, let us set up an environment that will show this feature.

We have a network that consists of the following setup:

- An AIX system, with host name colta, which serves as the Web-based System Manager Certificate Authority and the server.
- A Linux system, with host name clientA, which serves as the client.

In this scenario, we will use a single machine, the AIX server, to define an internal Certificate Authority (CA) and produce ready-to-go key ring files. This task will generate a public key ring file that you must copy to the Linux client as well as a unique private key ring file for the Web-based System Manager server.

The following steps describe how to set up this environment:

1. In colta, check if the Web-based System Manager Security fileset, sysmgt.websm.security, is installed. This fileset is not installed by default. You can use the AIX 5L Expansion Pack CD and install this fileset using the **`smit installp`** command.
2. Enable Web-based System Manager Server.

   In colta, execute the command:

   ```
   # /usr/websm/bin/wsmserver -enable
   ```

3. Define the internal Web-based System Manager Certificate Authority.

   In colta, log in locally as the root user and start Web-based System Manager:

   ```
   # wsm &
   ```

   > **Note:** The security configuration applications of Web-based System Manager are not accessible if you are not logged in as the root user or if you are running Web-based System Manager in remote application or applet mode.

   Select **Management Environment** -> **colta** -> **System Manager Security** -> **Certificate Authority**.

   From the task list for **Certificate Authority**, select **Configure this system as a Web-based System Manager Certificate Authority**. When the wizard opens, click **Next** and fill out the following information as follows:

   a. **Certificate Authority distinguished name:** `WebSysMgr CA-1 colta`

      This should be a descriptive name that helps you identify the CA machine and the instance of the CA, for example, the machine's host name plus a sequence number. Blanks are permitted in the name. The default CA name is `WebSysMgr CA-1 <hostname>`.

b. **Organization name:** `redbook`

   This should be a descriptive name that identifies your company or your organization.

c. **ISO country code:** `US` (United States)

   This is your two-character ISO country code. You can select from the list given.

d. Click **Next**.

e. **Expiration date:** `November 5 2004`

   You can change this date or accept the default value. After the certificate expires, reconfigure Web-based System Manager security by redefining the CA and generating new private key ring files for all of your servers.

f. Click **Next**.

g. **Public key ring directory:** `/var/websm/security/tmp`

   The public key ring (SM.pubkr) containing the CA's certificate is written to this directory. The default directory is /var/websm/security/tmp.

h. Click **Next**.

i. **Password:** `passw0rd`

   The CA's private key ring file is encrypted with this password. You will need to enter this password each time you perform a task on this CA.

j. Click **Next**, and then click the **Finish** button on the next window.

From the task list for **Certificate Authority**, the status of the Certificate Authority for Web-based System Manager should now be set to “Configured”, with the CA’s distinguished name listed.

You can also define an internal CA from the command line with:

```
# /usr/websm/bin/smdefca <CA name> -o <organization> -c \
<country code> -d <public key ring directory> [ -e <mm/dd/yyyy>]
```

4. Generate a Private Key Ring File for the Web-based System Manager.

   From the task list for **Certificate Authority** in colta, select **Generate Server’s Private Key Ring Files**. This task will prompt you for the CA password generated from step 3. Enter the CA password and fill out the following information as follows:

   a. **List of Servers:** `colta.itsc.austin.ibm.com`

      This should be a fully qualified TCP/IP name of your Web-based System Manager Server. If your machine is under a working DNS environment (with the /etc/resolv.conf file), the wizard will detect the host name automatically.

b. **Organization name:** `redbook`

c. **ISO country code:** `US` (United States)

d. **Location for Private key ring directory:** `/var/websm/security/tmp`

The private key ring file (colta.itsc.austin.ibm.com.privkr) for the server is written to this directory. The default directory is /var/websm/security/tmp.

e. **Length in bits of server keys:** `512`

This is the key length. The default value is 512.

f. **Expiration date:** `November 5 2004`

You can change this date or accept the default value. After the certificate expires, reconfigure Web-based System Manager security by generating new private key ring files for your server.

g. **Encrypt the server private key ring files**

If you select this option, you are prompted for a password, which you will need when you install the private key ring on the server.

h. Click **OK**.

i. **Password:** `passw0rd`

This will be private key ring file's password.

j. Click **OK**.

Alternatively, you can also generate a private key ring file from the command line with:

```
# /usr/websm/bin/smgenprivkr <CA name> -o <organization> -c \
<country code> -d <private key ring directory) [ -e <mm/dd/yyyy>]
```

5. Distribute the Public Key Ring File (SM.pubkr) to the Web-based System Manager Client.

Copy colta's CA public key ring file from the /var/websm/security/tmp directory, which we specified in step 3, to clientA's /opt/websm/codebase directory.

> **Note:** The content of this file is not secret. However, placing it on a client machine specifies which CA the client trusts. Thus, access to this file on the client machine should be limited.

6. Install the Private Key Ring File in the Web-based System Manager server.

   Log on to colta and start Web-based System Manager. Select **Management Environment** -> **colta** -> **System Manager Security** -> **Server Security** and do the following steps:

   a. From the task list, select **Install the private key ring file for this server**.

   b. Select the source for the server private key ring files, /var/websm/security/tmp, and then click **OK**.

   c. Since the key ring file is encrypted, you are asked for the password. Enter `passwOrd` and then click **OK**.

   d. The private key is then installed in the /var/websm/security directory.

   Alternatively, you can also install the private key ring file from the command line with:

   ```
   # /usr/websm/bin/sminstkey <private key ring directory>
   ```

7. Enable the server system to only accept secure connections.

   In colta, log on as root user and start the Web-based System Manager and do the following steps:

   a. Select **Management Environment** -> **colta** -> **System Management Security** -> **Server Security**.

   b. From the task list, select **Configure this system as a secure Web-based System Manager server** and click **Next**.

   c. In the next window, select **Always use a secure connection**. Then, click **Next**.

   d. Select **I will configure the installed Web server to provide SSL communications**. Then, click **Next**.

   e. Click **Finish**.

   From the task list for **Certificate Authority**, the status of the Secure Web-based System Manager server should now be set to `Configured`.

   Alternatively, you can also do this task from the command line with:

   ```
   # /usr/websm/bin/wsmserver -sslalways
   ```

8. Install the Web-based System Manager Remote Client Security.

   In clientA, launch a Web browser and connect to colta by specifying its fully qualified domain name or IP address in the following URL:

   ```
   http://colta.itsc.austin.ibm.com/remote_client_security.html
   ```

9. On the Web page, click on the Linux link and save the setupsecl.exe file in a directory of your choice, for example, /tmp.

10. On the Linux command line, run the following commands:

```
# chmod +x /tmp/setupsecl.exe
# /tmp/setupsecl.exe
```

The InstallShield Wizard will start the setup of your Web-based system Manager Remote Client Security. To complete the installation, proceed through the installation wizard by clicking **Next** on each window.

11. Run the Web-based System Manager Client in your Linux system.

To activate the Web-based System Manager Client, type the following command in clientA:

```
# /opt/websm/bin/wsm -host colta &
```

In the login panel, you will see the option **Enable secure connection** is checked by default (refer to Figure 10-8) and you cannot modify this setting. Once you log in successfully, security is indicated by a secure (locked key) icon on the status line at the bottom of the window panel (see Figure 10-9 on page 241).

Figure 10-8 Web-based System Manager Client logon window (security enabled)

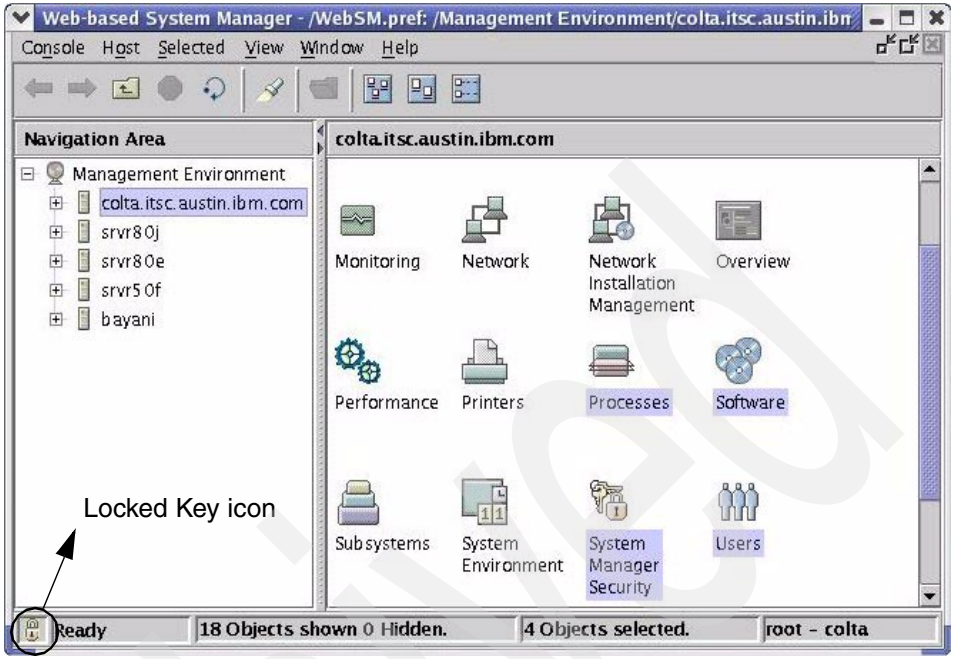

Figure 10-9 Web-based System Manager Client secure connection

10.2.2 Viewing configuration properties

After the security configuration has been completed, you can view the properties of the Certificate Authority (CA), any server, and any client's public key ring.

To view CA properties, do the following:

1. Open the **Management Environment** and select your local host.
2. Select the **Web-based System Manager Security** object.
3. Select the **Certificate Authority** object.
4. Select **Properties** from the task list.
5. Enter the password that you set during the setup of the CA. This dialog provides read-only information for the CA.

You can also perform this task from the command line using the command:

```
# /usr/websm/bin/smcaprop
```

Detailed information on all operations executed by the CA (for example, key ring generation or certificate signing) can be found in the /var/websm/security/SMCa.log CA log file.

To view the CA certificate included in the CA public key ring, you can use the command:

```
# /usr/websm/bin/smlistcerts <public key ring directory>
```

10.3 Webmin

Webmin is a Web-based interface for system administration for UNIX systems and commonly used in Linux systems. It consists of a simple Web server and CGI forms that are written in Perl Version 5, and use a non-standard Perl modules.

Webmin can be easily accessed through a Web browser that supports tables and forms on any client machine connected to the network. It provides a graphical interface to different services and tasks needed to maintain and administer a UNIX system. It does not modify files if there is no need to do it. If Webmin does not understand a particular option or command in your existing configuration, it will leave it as it is in the configuration file.

Webmin should not be viewed as a complete replacement for some of the system administration tools and techniques (like using the command line), but as an aid to a system administration to accomplish some task faster and easier.

10.3.1 How to obtain Webmin

You can download the Webmin source from:

http://www.webmin.com/download.html

Here you will find the latest version of Webmin in a tarball package and in an RPM package. The tarball will work on nearly any UNIX version that has Perl, while the RPM package is known to work directly at least on any versions of Linux. The Webmin version that we used for this example is webmin-1.020-1.

For AIX systems, you can download the Toolbox version of Webmin from:

http://www-1.ibm.com/servers/aix/products/aixos/linux/download.html

> **Note:** The AIX toolbox for Linux applications contains a collection of open source and GNU software and applications that have already been recompiled to work with both AIX 5L and AIX Version 4.3.3.

For more information on the AIX toolbox for Linux applications, refer to *Linux Applications on pSeries*, SG24-6033.

10.3.2 Installing Webmin

Installation of Webmin differs slightly depending on which type of package you choose to install and which operating system you are running it on. Note that Webmin requires Perl Version 5 or later for any of installation to work.

Unpacking/Installing the Webmin software source for Linux

Extract the Webmin source and it will typically create its own subdirectory. Here is how you can extract the different sources that you downloaded from the Webmin Web site:

- For the webmin-1.020-1.tar.gz file, extract it with this command:

  ```
  # gunzip -c webmin-1.020-1.tar.gz | tar xvf -
  ```

- For the webmin-1.020-1.noarch.rpm file, you can directly install it with this command:

  ```
  # rpm -iv webmin-1.020-1.noarch.rpm
  ```

Installing the tarball source in Linux systems

For the tarball source, change to the directory that was created when you untarred the archive, in this case, /usr/local/webmin-1.020.

> **Note:** It is recommended that you do not use a temporary directory for the source file. The directory where the source was first extracted will be used by Webmin when it is running. These files do not get copied to another location upon installation.

To set up webmin, simply run the setup.sh shell script. The script will start the installation and ask your preferences for the installation. Generally, you only need to accept the defaults. Refer to Example 10-3.

Example 10-3 Installing Webmin source in Linux

```
[root@rhlinux webmin-1.020]# ./setup.sh
***********************************************************************
*          Welcome to the Webmin setup script, version 1.020        *
***********************************************************************
Webmin is a web-based interface that allows UNIX-like operating
systems and common UNIX services to be easily administered.

Installing Webmin in /tmp/webmin-1.020 ...

***********************************************************************
Webmin uses separate directories for configuration files and log files.
Unless you want to run multiple versions of Webmin at the same time
you can just accept the defaults.
```

```
Config file directory [/etc/webmin]: /etc/webmin
...(lines omitted)...
```

Installing Webmin in AIX systems

For AIX systems, the Toolbox version of Webmin downloaded from the Toolbox Web site, webmin-0.88-3.aix4.3.noarch.rpm, is also available in an rpm installable format. You can install this file with the following command:

```
# rpm -iv webmin-0.88-3.aix4.3.noarch.rpm
```

10.3.3 Starting the Webmin interface

Starting and stopping the Webmin server can be accomplished by using the standard **service** command:

```
/sbin/service webmin stop
/sbin/service webmin start
```

You can also use the standard Webmin stop and start scripts located in the Webmin /etc directory; for Linux and AIX, it is /etc/webmin:

```
/etc/webmin/stop
/etc/webmin/start
```

10.3.4 Logging in to Webmin

From the client's machine, open a Web browser and use the following URL to access the Webmin Server:

```
http://<hostname or IP>:10000
```

where hostname or IP is the host name of the Webmin server you wish to administer.

After entering the URL, you will directed to the authentication Web form (refer to Figure 10-10 on page 245). Enter your valid Webmin user name and password in the form.

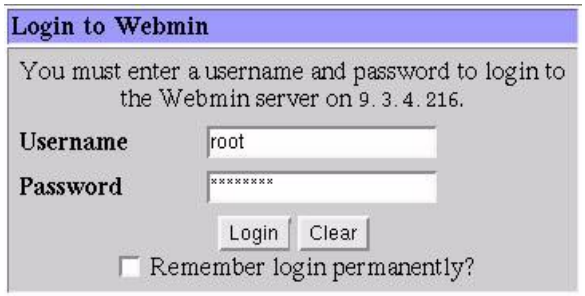

Figure 10-10 Webmin login window

After a successful authentication, the Webmin index page will open, as you can see in Figure 10-11 on page 246 for AIX systems and Figure 10-12 on page 247 for Linux systems, with Webmin version 1.020.

Note: The Webmin interface differs in different versions. You will have the same interface in AIX if you have installed the same version as the one in Linux. However, some of the modules will only work in Linux.

Your Web page always starts in the Webmin Index page. From this page, you will see a row of panels (categories) and several icons (modules) on each panel. These panels are labelled Webmin, System, Servers, and Others. You may also have additional panels depending on your OS and version (in an AIX system, we only have four panels available). In the Webmin category, you will find all your related configuration details for Webmin and its files.

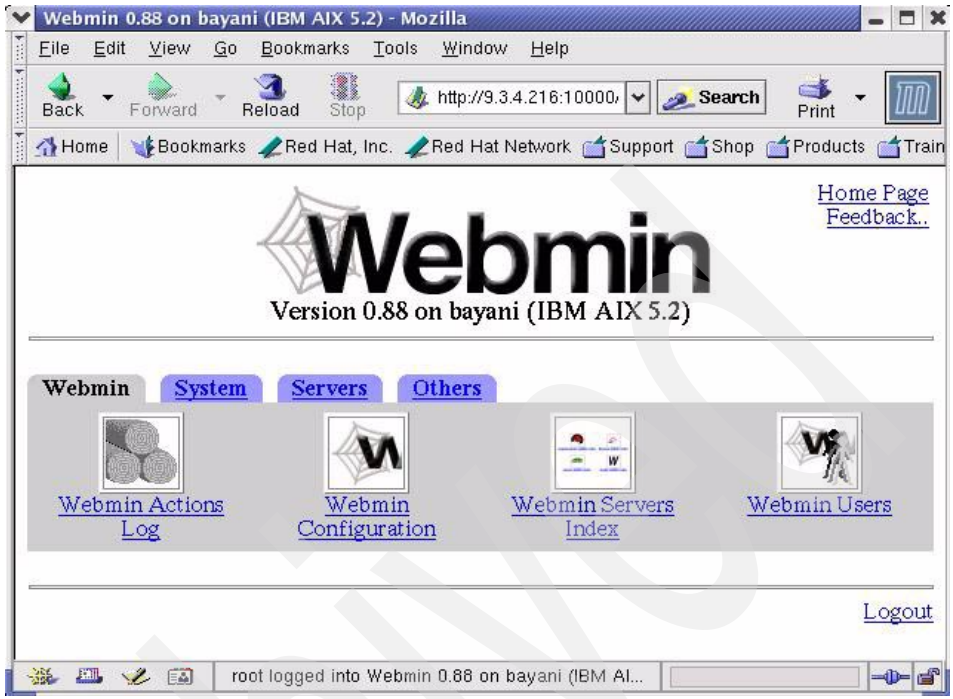

Figure 10-11 Webmin welcome page for AIX systems

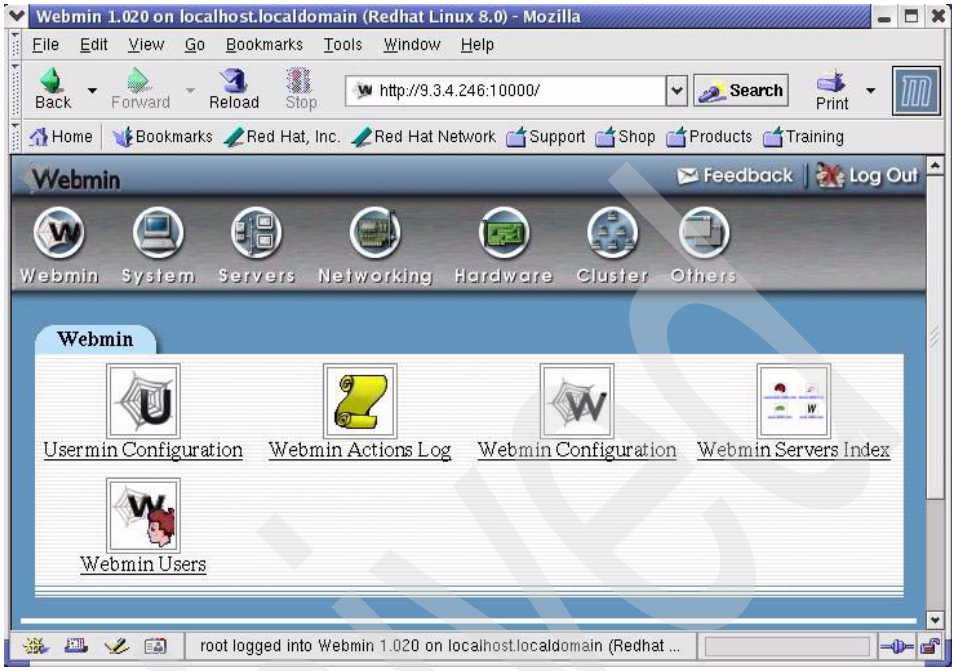

Figure 10-12 Webmin welcome Page for Linux systems

Now, let us look at some of the panels/categories of Webmin that are both present in AIX and Linux.

10.3.5 Webmin panels/categories

As discussed earlier, Webmin's Web interface consists of modules and tasks that are categorized into different panels or tabs to give it a more user friendly interface. This section will discuss the four basic categories that are present in both AIX and Linux systems.

Webmin category

In this category, you will see icons for Webmin Actions Log, Webmin Configuration, Webmin Servers Index, and Webmin Users. These icons include modules that provide a number of configurable options, access control features, and action logs to allow maximum flexibility and security of the Webmin server. It also includes other Webmin system administration modules. But, keep in mind that the modules located under this category are only for configuring the Webmin application alone and not the system it is running on. In the Webmin servers index page, you can access and add other Webmin servers that you want to

administer. An example Web page of the Webmin category is shown in Figure 10-13.

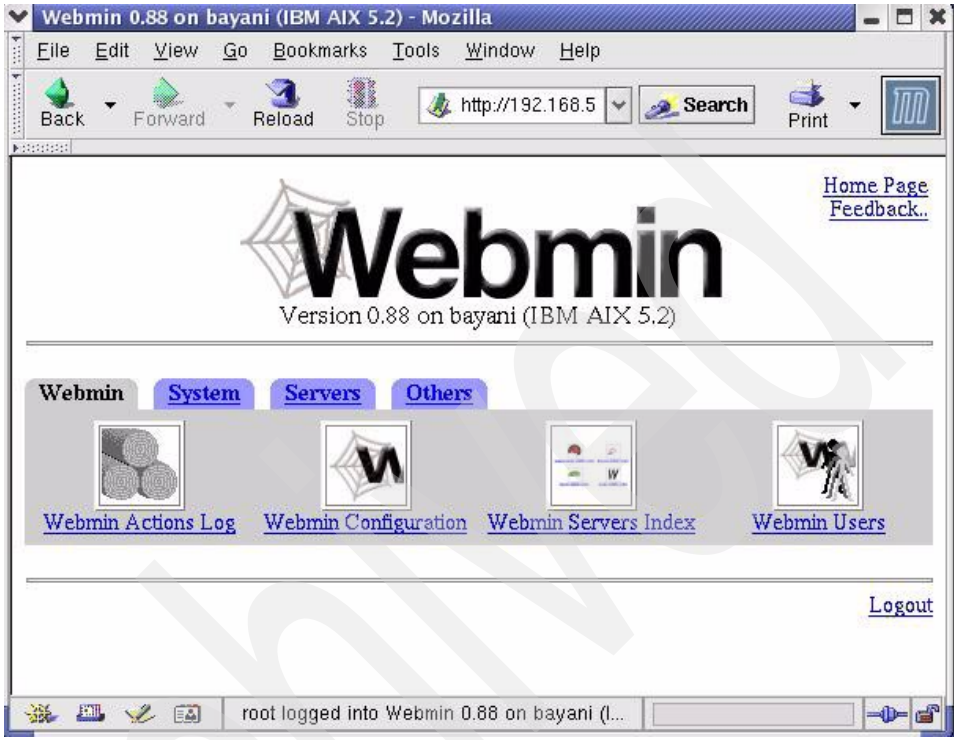

Figure 10-13 Webmin category

Systems category

The second category on the Webmin Web page is the System category. This category includes the most common system administration tasks. This interface will allow you to edit and monitor system features such as bootup and shutdown behavior, passwords, NFS Exports, system processes, cron job schedules, system documentations (which includes man pagers) and logs, and managing users and groups. Figure 10-14 on page 249 shows the options available on the Systems category Web page.

Figure 10-14 Systems category

Servers category

The third category in the Webmin Web page is the Servers category. It includes various applications and daemons that can be configured and monitored by Webmin.

Webmin includes different modules for different applications, allowing the user to administer and edit the configuration files for these applications. This will allow a system administrator to control the behavior and functionality of the applications installed on your system. The standard modules includes Apache, Bind Fetchmail, FTP, Postfix, Sendmail, Squid, NFS, and Samba.

Figure 10-15 on page 250 shows the options available on the Servers category Web page.

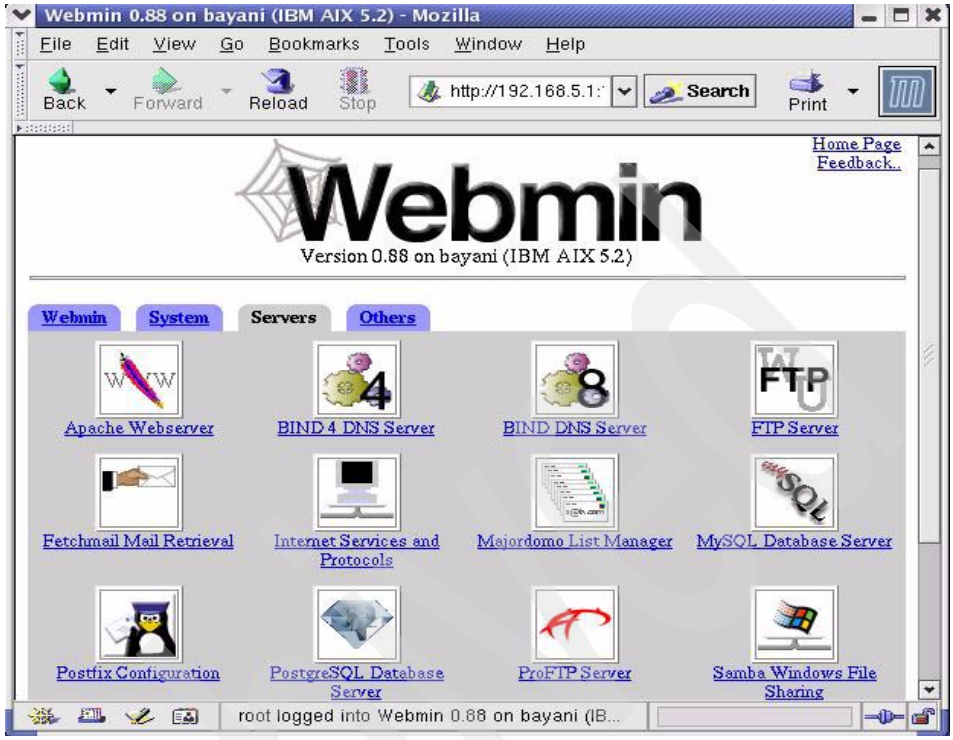

Figure 10-15 Servers category

Others category

The Others category includes miscellaneous tools that do not fit in any of the previous categories. This module include the System Command module, the Custom Commands module, a Java-based file manager module, the SSH/telnet Login module, and the Perl modules.

The Others category Web page is shown in Figure 10-16 on page 251.

Figure 10-16 Others category

10.3.6 Webmin security features

Because browsers often store authentication information and automatically resend it on demand from the Webmin server, anyone with malicious intent can easily trigger dangerous actions on your Webmin sever. And, since Webmin is a Web based application that can be accessed from any browser on the network, it is possible for a remote user to get into the Webmin server easily.

Putting this into consideration, Webmin provides security features to prevent this from happening and to maximize the security of your system. There are four features/options that you can use to implement security on your Webmin server:

- Authentication

  This feature will prevent password cracking attacks on your server, as well as protection against "forgetful users". This feature can be accessed by clicking on the **Webmin** panel and then on the **Webmin Configuration** icon. Here, you can enable password timeouts by configuring the allowable time delay between each failed login for the same user to a Webmin server and at the same time log the failed login attempts to syslog. Also, you can enable

session authentication with this feature. This will allow Webmin to track all logged in users and automatically logs out users that have been idle for a specified period of inactivity.

- Trusted referrers

  This page allows you to configure Webmin's referrer checking support. The referrer checking support will prevent malicious links from other Web sites from tricking your browser into doing dangerous things with your Webmin server. It also has an option where you can add trusted hosts or Web sites that may refer actions to your Webmin server. To enable this feature, just click on the Webmin configuration icon in the Webmin category panel

- SSL encryption

  This feature will increase the security of your system by sending user and password information in encrypted form. This feature will require OpenSSL libraries and the Net::SSLeay Perl module be installed. After installing the needed library and module, the SSL encryption icon will be added in the Webmin configuration section in the Webmin configuration panel. More information about this feature and a setup howto can be found at the following URL:

  http://www.webmin.com/ssl.html

- Using certificate authority

  If you were able to setup the SSL encryption, you can configure SSL certificates for Webmin servers. This will allow user logins to your Webmin server without a user name and password. From the Webmin Users module, clients can request a personal certificate and be able to authenticate themselves, using that certificate, through the browser. Although this feature is simpler to implement than the usual user name and password logons, there is a potential security issue here: Any user who has access to a client's machine that has a valid certificate can access the Webmin server with the same privileges as the real user. The auto-logout feature of Webmin will not address this issue because authentication is done automatically every time the user starts a browser session.

11

Printer sharing

In this chapter, we describe the methods of printing between AIX systems and Linux systems. We discuss scenarios where a printer is attached to AIX or Linux and how to access the remote printer from either of the systems. We have already described one way of sharing printers between AIX and Linux by using the SMB protocol in 6.3, “Samba file and print server on AIX and client on Linux” on page 125 and 6.4, “Samba file and print server on Linux and client on AIX” on page 136.

Printing on AIX and Linux is fairly well documented. You can read about AIX printing in the *AIX 5L Version 5.2 Guide to Printers and Printing.*

You can read about Linux printing from the Linux Printing HOWTO, found at:

http://www.linuxprinting.org/howto/how.html

In this chapter, we discuss the following topics:

- Printing from Linux to a printer attached to AIX
- Printing from AIX to a printer attached to Linux
- Configuring Directory-Enabled (LDAP) System V print on AIX

11.1 Printing from Linux to a printer attached to AIX

In this section we describe the scenario of printing from Linux to a printer attached to AIX.

- Printing on AIX
  - AIX queuing system
  - Adding a local printer on AIX
- Configuring remote printing on Linux using LPD

11.1.1 Printing on AIX

We describe the procedure to install a printer on AIX. For more information on AIX printing and the commands used for printing, refer to the AIX documentation *AIX 5L Version 5.2 Guide to Printers and Printing*.

AIX queuing system

AIX uses a general queuing system that may be used for any queue-related purpose, and printing is one of the ways of using this queuing system.

The user submits the print job to a queue. A server called qdaemon monitors those queues and schedules and initiates jobs. In the case of printing on the local system, qdaemon sends the job to a back-end program /usr/lpd/piobe. The output of the back-end program is sent to the specified physical device in the case of local printing. To send a print job to a remote printer, the /usr/lpd/rembak back-end program is used.

You can submit print jobs on AIX with the **enq** command. AIX also provides the **lp**, **lpr** or **qprt** commands for compatibility with BSD and AT&T-style printing. But **lp**, **lpr** and **qprt** are just the front ends to the **enq** command and do not represent different printing subsystems.

Adding a local printer on AIX

You can install a printer on AIX using SMIT. The following steps describe the setup:

1. Start SMIT with the following command:

   ```
   # smit pdp
   ```

   You will see a screen, as shown in Example 11-1 on page 255. Select the **Add a Printer/Plotter** option.

Example 11-1 Adding a local printer on AIX using SMIT

```
                        Printer/Plotter Devices

Move cursor to desired item and press Enter.

  List All Defined Printers/Plotters
  List All Supported Printers/Plotters
  Add a Printer/Plotter
  Move a Printer/Plotter to Another Port
  Change / Show Characteristics of a Printer/Plotter
  Remove a Printer/Plotter
  Configure a Defined Printer/Plotter
  Install Additional Printer/Plotter Software
  Generate Error Report
  Trace a Printer/Plotter

F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

2. You will have to select your printer from the **Printer/Plotter Type** interface menu. If you do not find your printer in the list, then select a printer that closely resembles your printer.
3. From the **Printer/Plotter Interface** menu, depending on your connection, select either **parallel**, **rs232**, or **rs422**.
4. In the next screen you should choose the **Parent Adapter** for your connection. For parallel connection you should see:

   ppa0 Available 00-00-0P Standard I/O Parallel Port Adapter

   For serial, there are two options:

   sa0 Available 00-00-S1 Standard I/O Serial Port 1

   sa1 Available 00-00-S2 Standard I/O Serial Port 2

   Choose the option applicable to you.
5. Next, you have to type in the **Port number** in the **Add a Printer/Plotter** menu. Press F4 for a list and choose the appropriate port number. Press Enter.
6. SMIT then adds the printer to the configuration database and displays **lp0 Available** in the Command Status screen.
7. In order to print from the printer just added, you need to create a queue. You can do this with SMIT. Go to the **AIX Print Spooling** section of SMIT. As shown in Example 11-2 on page 256, choose the **Add a Print Queue** option.

Example 11-2 Adding a queue for printing

```
                              AIX Print Spooling

Move cursor to desired item and press Enter.

  Start a Print Job
  Manage Print Jobs
  List All Print Queues
  Manage Print Queues
  Add a Print Queue
  Add an Additional Printer to an Existing Print Queue
  Change / Show Print Queue Characteristics
  Change / Show Printer Connection Characteristics
  Remove a Print Queue
  Manage Print Server
  Programming Tools

  Change / Show Current Print Subsystem


F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

8. You should select the attachment type **local**.
9. Select the appropriate Printer from the **Printer Type** menu.
10. In the **Printer** menu, select from the **existing printer**.
11. You have to create different queues, depending upon the print formats your printer supports. Example 11-3 shows two queues for PCL and PostScript supported by the IBM InfoPrint 40 printer.

    Press Enter after you type in the queue names.

Example 11-3 Adding two queues for PCL and Postscript

```
                              Add a Print Queue

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                      [Entry Fields]
  Description                                        IBM InfoPrint 40
  Printer name                                       lp0

  Names of NEW print queues to add
     PCL 5E Emulation                               [pclq]
     PostScript                                     [psq]
```

```
F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

12. You should get the following output on the Command Status screen:

```
Added print queue 'pclq'.
Added print queue 'psq'.
```

13. When you execute the **lpstat** or **lpq** command, you will see the following output:

```
# lpstat
Queue   Dev   Status    Job Files              User         PP %   Blks  Cp
Rnk
------- ----- --------- --- ------------------ ---------- ---- -- ----- ---
---
pclq    lp0   READY
psq     lp0   READY
```

14. In order to let other systems use the printer, you have to add the host name/IP address of each print client. You can do this through SMIT as follows:

    a. Start the print server section of SMIT by executing:

    ```
    # smit server
    ```

    You will see a screen similar to Example 11-4.

Example 11-4 Managing the print server on AIX through SMIT

```
                              Manage Print Server

Move cursor to desired item and press Enter.

  List all Remote Clients with Print Access
  Add Print Access for a Remote Client
  Remove Print Access for a Remote Client
  Start the Print Server Subsystem (lpd daemon)
  Stop the Print Server Subsystem
  Show Status of the Print Server Subsystem

F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

    b. Select the **Add Print Access for a Remote Client** option and press Enter. You will see a screen similar to Example 11-5 on page 258.

Example 11-5 Adding print access for a remote client

```
                  Add Print Access for a Remote Client

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                        [Entry Fields]
* Name of REMOTE CLIENT                                 []
  (Hostname or dotted decimal address)



F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

c. Type in the host name or IP address of the client and press Enter. You should see the OK on the Command Status screen.

Now you are ready to print from AIX and also accept print requests from clients you added to your remote print list. You can check local printing by printing a print job using the **enq**, **lp**, or **lpr** commands. In case of difficulties, refer to the AIX documentation.

In the next section, we describe how to print from the Linux system to the printer we just configured.

11.1.2 Configuring remote printing on Linux

We describe the configuration of the printing software LPRng on Linux. LPRng can be used to print to local as well as remote printers.

Installing LPRng on Linux

In most Linux distributions, the LPRng printer software is installed by default. If you have to install LPRng, you can install it using a precompiled binary package. Depending upon your Linux distribution and machine architecture, you can download the RPM from:

http://www.rpmfind.net/linux/rpm2html/search.php?query=LPRng

You can also install LPRng from the source code. You can find the LPRng source code and the documentation to install it at:

http://sourceforge.net/projects/lprng/

Configuring the /etc/printcap file to print on a remote printer

LPRng uses /etc/printcap as the configuration file. In order to print to a remote printer on AIX, the /etc/printcap file should contain lines similar to the following:

```
lp|pclq|aix_printer:\
:sd=/var/spool/lpd/pclq:\
:rm=aix_host:\
:rp=pclq:\
:sh:
```

The various entries are:

- pclq: The printer name or the name of the AIX queue.
- /var/spool/lpd/: The local spool directory on Linux.
- aix_host: The resolvable system name of AIX.

For every new printer, you have to add a similar entry in the /etc/printcap file. For more information, refer to the man pages of **printcap**.

After you make the changes in the printcap file, restart the lpd daemon. You can do that by executing the **/etc/rc.d/init.d/lpd restart** or **/etc/rc.d/lpd restart** command, depending on your Linux distribution.

If you have just the above entry in the /etc/printcap file, you can print to the remote printer on AIX using the **lpr** command, as the default printer is the remote printer. For example:

```
# lpr /etc/printcap.local
# lpq
Printer: lp@localhost 'aix_printer' (dest pclq@aix)
 Queue: no printable jobs in queue
 Server: no server active
 Status: job 'root@localhost+849' saved at 14:49:13.013
 Rank   Owner/ID                Class Job Files                 Size Time
done   root@localhost+849           A   849 /etc/printcap.local    603
16:39:33
Queue   Dev   Status    Job     Name           From           To
                         Submitted        Rnk Pri       Blks  Cp          PP %
 ------- ----- --------- ---------        --- ---      ----- ---        ---- --
 pclq    lp0   READY
```

Most of the entries are self-explanatory. Refer to the **lpq** man pages for a description of all the fields.

11.2 Printing from AIX to a printer attached to Linux

In this section, we describe the scenario of printing from AIX to a printer attached to Linux:

- Installing a printer on Linux
- Configuring remote printing on AIX

11.2.1 Installing a printer on Linux

We have already described how to install the LPRng software on Linux. The procedure to install the printer on Linux depends upon your Linux distribution and is fairly straightforward for major Linux distributions, as they provide a Graphical User Interface (GUI). Refer to the Linux Printing HOWTO in section **Vendor Solutions** to find out how to install a printer for your Linux distribution.

For example, we describe printer installation on Red Hat Linux Version 8.0. Red Hat has a GUI printer administration tool called printtool, which can add remote printers and printers on local devices.

To install the printer do the following steps:

1. Start the printtool tool:

   ```
   # printtool
   ```

   You will get Figure 11-1.

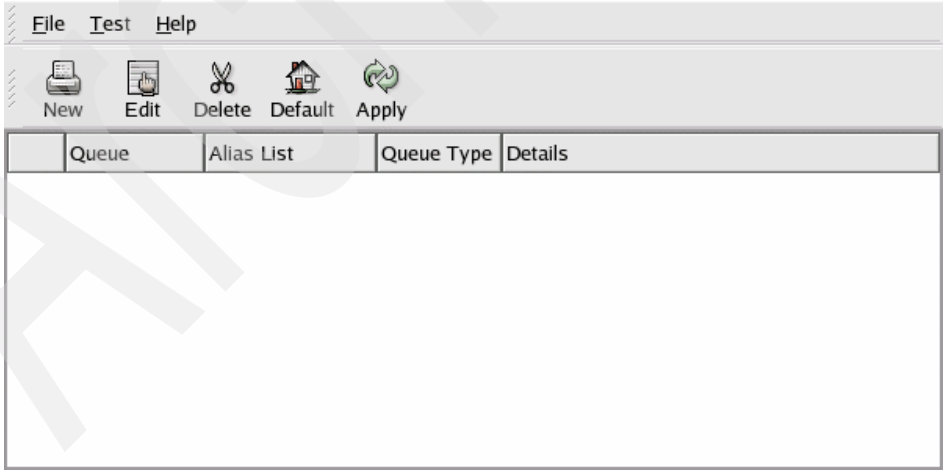

Figure 11-1 First window of printtool on Red Hat Linux Version 8.0

2. Click **New** to get Figure 11-2 on page 261.

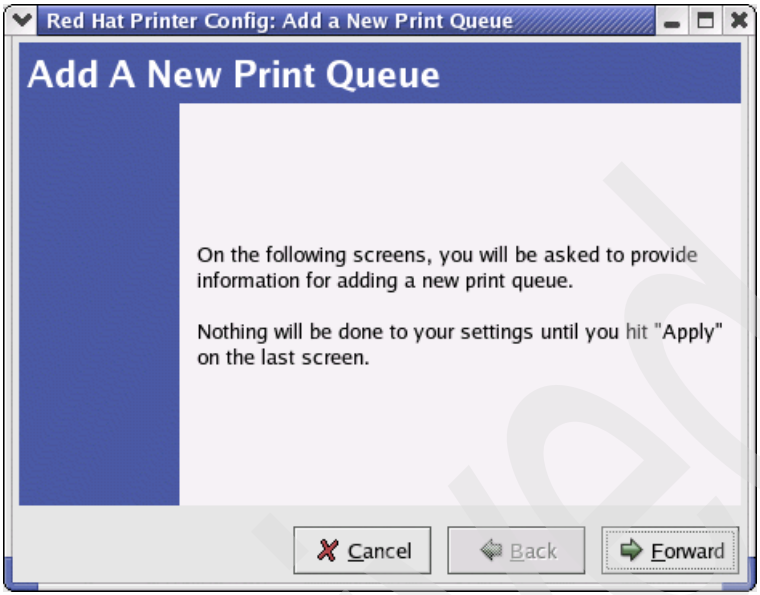

Figure 11-2 Adding a new print queue through printtool

3. Click **Forward** to get Figure 11-3. Type in the printer queue name and select **Local Printer**. Click **Forward** when done.

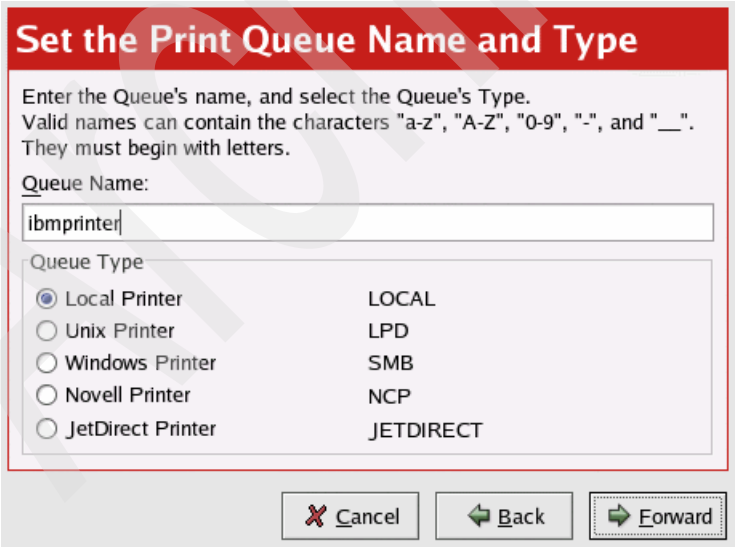

Figure 11-3 Setting the print queue name and type

4. Figure 11-4 shows that Red Hat automatically detected the printer IBM 4332-40. You should use the **Custom Device** option if the printer is not listed. Refer to the HOWTO in case you face problems in detecting the printer. Click **Forward** once you are done.

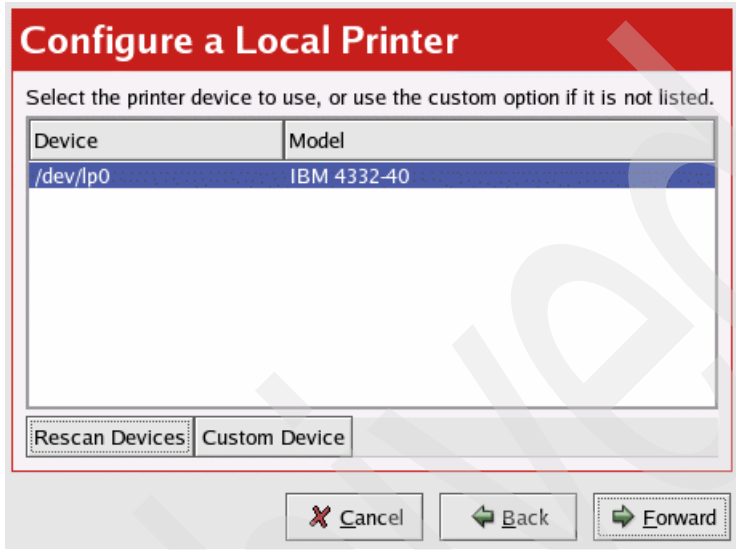

Figure 11-4 Configuring a local printer

5. Select an appropriate driver for your printer, as shown in Figure 11-5 on page 263. Click **Forward** once you are done.

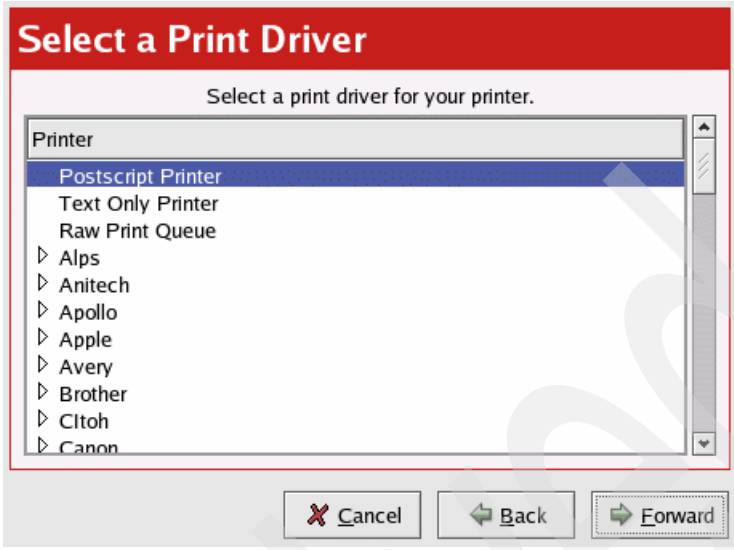

Figure 11-5 Selecting a print driver

6. Confirm the settings in the next window, as shown in Figure 11-6. Click **Apply**.

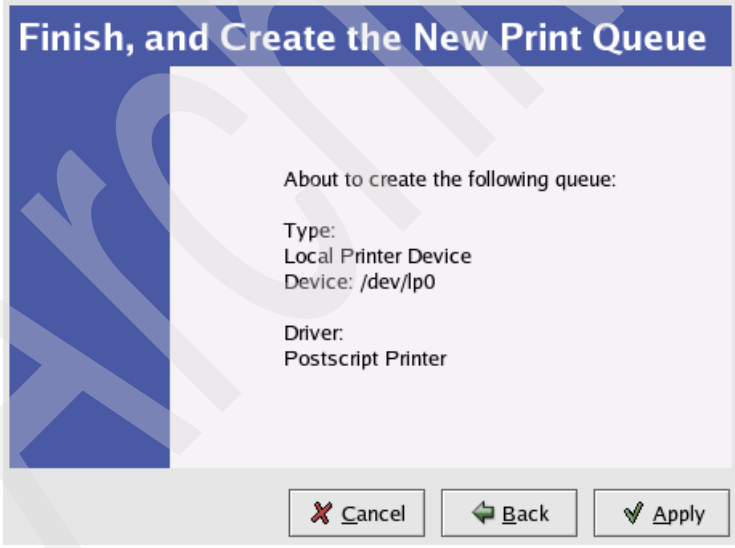

Figure 11-6 Confirming the creation of a new queue

7. You will get a window showing the queue that you configured. This can be seen in Figure 11-7.

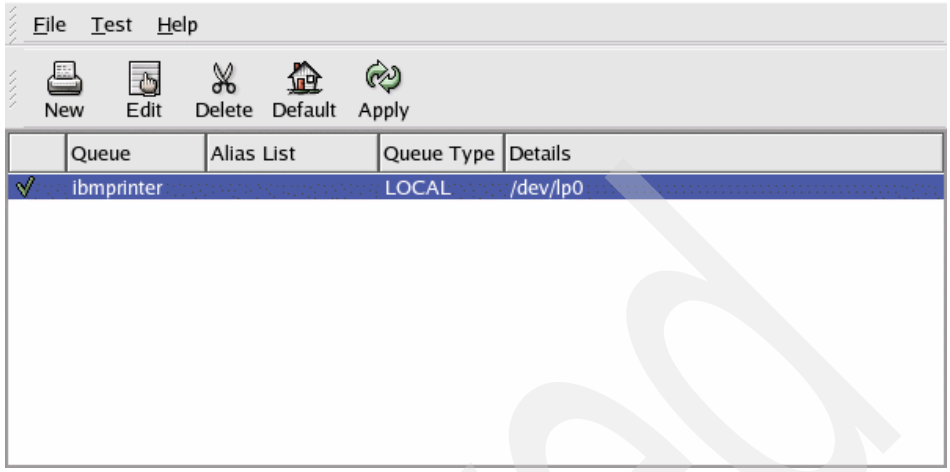

Figure 11-7 A properly configured print queue

8. You can test the printer setup by clicking on **Test** and then printing the **US Letter PostScript Test page** print job (see Figure 11-8).

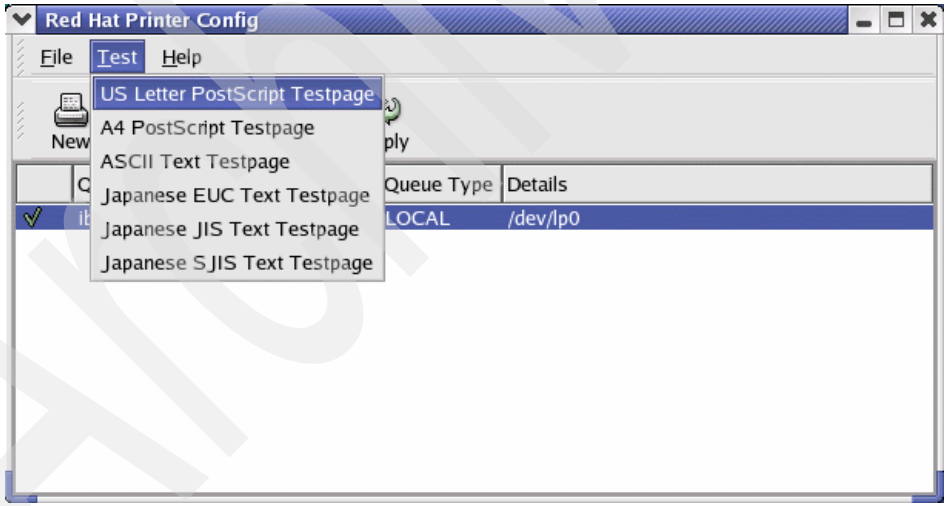

Figure 11-8 Testing the installed printer

After setting up the printer, you need to configure it to enable lpd to receive print jobs from remote. You should edit the lpd.perms file, normally stored in /etc directory. You can do fine tuning with the lpd.perms file to allow certain users/systems/networks to access the print queue. Refer to the man pages of lpd and read the lpd.perms file for more information. In our example, we simply put in

the following line, commenting all other lines, as shown in Example 11-6. This will enable anyone from any network to print to the Linux printer.

Example 11-6 Example of the /etc/lpd.perms file

```
# Printer permissions data base
## #
##                  LPRng - An Enhanced Printer Spooler
##                     lpd.perms file
##                   Patrick Powell <papowell@lprng.com>
##
## VERSION=3.8.9
##
## Access control to the LPRng facilities is controlled by entries
## in a set of lpd.perms files.  The common location for these files
## are: /etc/lpd.perms,  /usr/etc/lpd.perms, and  /var/spool/lpd/lpd.perms.
## The locations of these files are set by the perms_path entry
## in the lpd.conf file or by compile time defaults in the
## src/common/defaults.c file.
##
## Each time the lpd server is given a user request or carries out an
## operation,  it searches to the perms files to determine if the action
## is ACCEPT or REJECT.  The first ACCEPT or REJECT found terminates the
search.
## If none is found,  then the last DEFAULT action is used.
##
## Permissions are checked by the use of 'keys' and matches.  For each of
## the following LPR activities,  the following keys have a value.
.........
.........

## # All operations allowed except those specifically forbidden
 DEFAULT ACCEPT
```

This sets up the printer on Linux. In the next section, we describe how to access this printer from AIX.

11.2.2 Configuring remote printing on AIX

AIX handles remote printing in a fashion similar to local printing, except that the back-end program used for remote printing is rembak instead of piobe.

To configure a remote printer using SMIT, do the following steps:

1. Start SMIT and go to the AIX print Spooling screen to add a queue for the remote printer. The screen is similar to Example 11-7. Select the **Add a Print Queue** option and press Enter.

Example 11-7 Screen to add a print queue using SMIT

```
                         AIX Print Spooling

Move cursor to desired item and press Enter.

  Start a Print Job
  Manage Print Jobs
  List All Print Queues
  Manage Print Queues
  Add a Print Queue
  Add an Additional Printer to an Existing Print Queue
  Change / Show Print Queue Characteristics
  Change / Show Printer Connection Characteristics
  Remove a Print Queue
  Manage Print Server
  Programming Tools

  Change / Show Current Print Subsystem


F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

2. You have to select the **remote** attachment type among the options shown in Example 11-8 and press Enter.

Example 11-8 Selection of the attachment type for the queue

```
                         AIX Print Spooling

Move cursor to desired item and press Enter.

  +-------------------------------------------------------------------------+
  |                            Add a Print Queue                            |
  |                                                                         |
  | Move cursor to desired item and press Enter. Use arrow keys to scroll.  |
  |                                                                         |
  |   # ATTACHMENT TYPE      DESCRIPTION                                    |
  |     local                Printer Attached to Local Host                 |
  |     remote               Printer Attached to Remote Host                |
  |     xstation             Printer Attached to Xstation                   |
  |     ascii                Printer Attached to ASCII Terminal             |
  |     hpJetDirect          Network Printer (HP JetDirect)                 |
```

```
  ¦     file                 File (in /dev directory)                        ¦
  ¦     ibmNetPrinter        IBM Network Printer                             ¦
  ¦     ibmNetColor          IBM Network Color Printer                       ¦
  ¦     other                User Defined Backend                            ¦
  ¦                                                                           ¦
  ¦ F1=Help                 F2=Refresh              F3=Cancel                 ¦
  ¦ F8=Image                F10=Exit                Enter=Do                  ¦
F1¦ /=Find                  n=Find Next                                       ¦
F9+---------------------------------------------------------------------------+
```

3. In the next screen, you select the type of remote printing. Example 11-9 shows the selection options. The options are self-explanatory. You can select the **Standard processing** option and press Enter.

Example 11-9 Type of remote printing

```
                            AIX Print Spooling

Move cursor to desired item and press Enter.

  Start a Print Job
  Manage Print Jobs
  List All Print Queues
  Manage Print Queues
  Add a Print Queue
  Add an Additional Printer to an Existing Print Queue
  Change / Show Print Queue Characteristics
  +---------------------------------------------------------------------------+
  ¦                         Type of Remote Printing                           ¦
  ¦                                                                           ¦
  ¦ Move cursor to desired item and press Enter.                              ¦
  ¦                                                                           ¦
  ¦   Standard processing                                                     ¦
  ¦   Standard with NFS access to server print queue attributes               ¦
  ¦   Local filtering before sending to print server                          ¦
  ¦                                                                           ¦
  ¦ F1=Help                 F2=Refresh              F3=Cancel                 ¦
  ¦ F8=Image                F10=Exit                Enter=Do                  ¦
F1¦ /=Find                  n=Find Next                                       ¦
F9+---------------------------------------------------------------------------+
```

4. In case the **Local filtering before sending to print server** option needs to be selected, we need to select the remote printer type from a menu, as shown in Example 11-10 on page 268. You may need to go through one or two screens until you find your remote printer. When you are done, press Enter.

Example 11-10 Selecting the remote printer type for local print filtering

```
                                AIX Print Spooling

Move cursor to desired item and press Enter.
  +-------------------------------------------------------------------------+
  |                            Remote Printer Type                          |
  |                                                                         |
  | Move cursor to desired item and press Enter.                            |
  |                                                                         |
  |   Bull                                                                  |
  |   Canon                                                                 |
  |   Dataproducts                                                          |
  |   Hewlett-Packard                                                       |
  |   IBM                                                                   |
  |   Lexmark                                                               |
  |   OKI                                                                   |
  |   Printronix                                                            |
  |   QMS                                                                   |
  |   Texas Instruments                                                     |
  |   Other (Select this if your printer type is not listed above)          |
  |                                                                         |
  | F1=Help                 F2=Refresh              F3=Cancel               |
  | F8=Image                F10=Exit                Enter=Do                |
F1| /=Find                  n=Find Next                                     |
F9+-------------------------------------------------------------------------+
```

5. Type in an appropriate name for the remote queue, the host name of the remote server, and the print spooler on the remote server.
   Example 11-11shows the entries for our scenario. When you are done, press Enter and you should see OK in the Command Status screen.

Example 11-11 Local queue, remote print server, and remote print spooler names

```
                    Add a Standard Remote Print Queue

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                        [Entry Fields]
* Name of QUEUE to add                                 [rlpq]
* HOSTNAME of remote server                            [redhat]
* Name of QUEUE on remote server                       [ibmprinter]
  Type of print spooler on remote server               AIX Version 3 or 4     +
  Backend TIME OUT period (minutes)                    []                      #
  Send control file first?                              no                    +
  To turn on debugging, specify output                 []
      file pathname
  DESCRIPTION of printer on remote server              []
```

```
F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

6. You can now print from AIX to the printer on Linux by using the **enq**, **lpr**, or **lp** command. For example:

```
# lpq
Queue   Dev   Status    Job Files              User         PP %   Blks  Cp
Rnk
------- ----- --------- --- ------------------ ---------- ---- -- ----- ---
---
rpclq   @redh READY
rpclq: ibmprinter@localhost 0 jobs
```

   This shows that rpclq queue is ready to accept print jobs. For more information on **lpq**, refer to its man pages.

11.3 Configuring Directory-Enabled (LDAP) System V printing on AIX

LDAP provides a specialized database (directory) that is suited to the read-many, write-seldom model. The IBM directory is an LDAP directory server. Using the IBM Directory, we can have a centralized storage of print information from where we can share print, print queues, and system information in a client-server environment. In order to achieve this centralized storage of print information, we can use the AIX (Version 5.2.0) Print V subsystem.

The **mkprtldap** command configures IBM Directory as a server containing System V print information, and one or more clients that use the IBM Directory (LDAP) for print information.

Refer to the “Configuring Director-Enabled (LDAP) System V Print on AIX” section in the *AIX 5L Version 5.2 Guide to Printers and Printing* for more details.

We provide an overview of the procedure to manage the System V print subsystem using information stored in the LDAP directory:

1. Install the IBM Directory Server if it is not installed already. For information on how to install IBM Directory Server, refer to the documentation provided with the IBM Directory product.

2. Configure the IBM Directory to store System V information using the **mkprtldap** command. The syntax is:

   ```
   mkprtldap -s -a AdminDN -p Adminpasswd -w ACLBindPasswd [-f] [-d node DN]
   ```

   Most of the options are self-explanatory. The -f option is used to force the creation of print subtree when one or more AIX information trees already exist on the AIX directory. Refer to the AIX documentation for all the server options.

3. Set up the System V print information tree. System V print information is stored under the print subtree, which in turn is stored under a default AIX Information tree (cn=aixdata) on the directory. It is recommended to store the print information in the default location on the directory. For a detailed setup and example, refer to the AIX documentation.

4. You should configure the client that uses the IBM Directory Server for access to System V printing on AIX:

   a. Install the IBM Directory client software on the system to be used as the client.

   b. You can use the **mkprtldap** command with the following syntax:

      ```
      mkprtldap -c -h DirectoryServerHostname -w ACLBindPasswd [ -d
      PrintBindDN ] [-U]
      ```

      The -c option specifies the client action.

      This command makes, creates, or modifies the client files /etc/ldapsvc/server.print and /etc/ldapsvc/system.print.

This sets up the System V print services on server and client to read the print information from the IBM Directory (LDAP).

It is also possible to store printer information on the OpenLDAP server and access that information to generate printcap or qconfig printer configuration files on printer clients using a programming language like Perl. There is a book titled *Network Printing* by Radermacher, et al., which can be found at http://www.oreilly.com/catalog/netprint/, which has detailed information about sharing printer information using OpenLDAP. This book is also an excellent reference material for printing.

12

Linux for IBM eServer pSeries and RS/6000

The high level of activity on the UNIX-based systems and Linux fronts during the past few years is allowing Linux to establish itself as a mainstream UNIX player. This chapter discusses the point of integration where Linux meets UNIX, specifically the AIX operating system, and how to achieve the best of both worlds.

This chapter will provide a brief introduction on the following topics:

- Linux on a logical partition (LPAR)
- AIX toolbox for Linux applications

This chapter will not cover the installation and implementation of Linux on LPAR and the AIX toolbox for Linux applications. To get more information on these topics, several links will be provided in each section.

Also, more information concerning Linux on pSeries can be found at this URL:

http://www.ibm.com/servers/eserver/pseries/linux/

12.1 Linux on a logical partition (LPAR)

Logical partitioning was originally supported for IBM @server zSeries and iSeries™ systems, allowing the division of a single server into several completely independent and individual “virtual” servers or partitions. In 2001, IBM announced the first IBM @server pSeries POWER4-based server with LPAR capability.

The IBM @server pSeries 690 introduced several advanced technologies. Its logical partitioning capability offers true flexibility when selecting resources for partitions, and its dynamic logical partitioning (DLPAR on AIX 5L Version 5.2) allows to move resources non-disruptively between logical partitions. Each logical partition is independent of the operations occurring within other partitions and can run its own version of the operating system or a different one, either AIX 5L or Linux, to accommodate multiple application requirements. The IBM @server pSeries 690 is the first enterprise UNIX server capable of running Linux in a logical partition.

As seen in Figure 12-1, the LPAR technology is mostly used in an environment with a server consolidation requirement. Server consolidation enables customers to run different operating systems (AIX or Linux), application code levels, and workloads (for example, OLTP, Web servers, and databases) while using varied resource configurations.

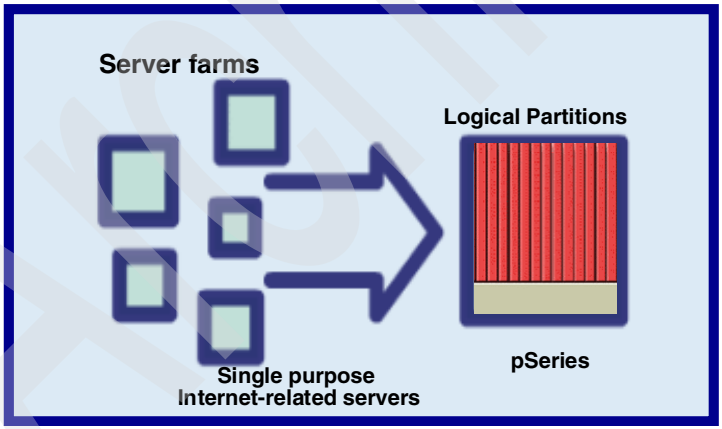

Figure 12-1 Server consolidation

AIX and Linux can run concurrently in separate partitions on an LPAR-enabled system. This enables separate physical servers to consolidate their workloads on a single system.

Here are the benefits of using LPAR for server consolidation:

- Reduce costs

  With server consolidation, you can achieve reduced total cost of ownership and lower costs of computing. This setup is ideal for consolidating small to medium scale infrastructures. It also caters to smaller server consolidation, because of its partition flexibility with the minimum of 1 processor and 1 GB requirement. Therefore, the use of resources is more efficient.

  It reduces costs not only on the server itself, but also for power, cooling, floor space, and software licenses.

- Fewer systems reduce systems management requirements

  Moving the functions of several servers to one physical system will result in a more economical system that is easier to operate and less complex to manage. In effect, you get less floor space, less servers to manage, and less skills needed to administer the system.

- Better workload management flexibility

  Keeping track of your system's resources is a vital task in working with a multiple partitioned server. The IBM Hardware Management Console (HMC) for pSeries is included with the p690 system, as shown in Figure 12-2 on page 274. The HMC is the central point for system management of the logical partitions. It performs several functions, including creating and maintaining partitions, shared resources, and power control over partitions, as well as the managed system, and providing virtual terminals for working within partitions. It also monitors and stores hardware changes while serving as a focal point for system administration on the p690.

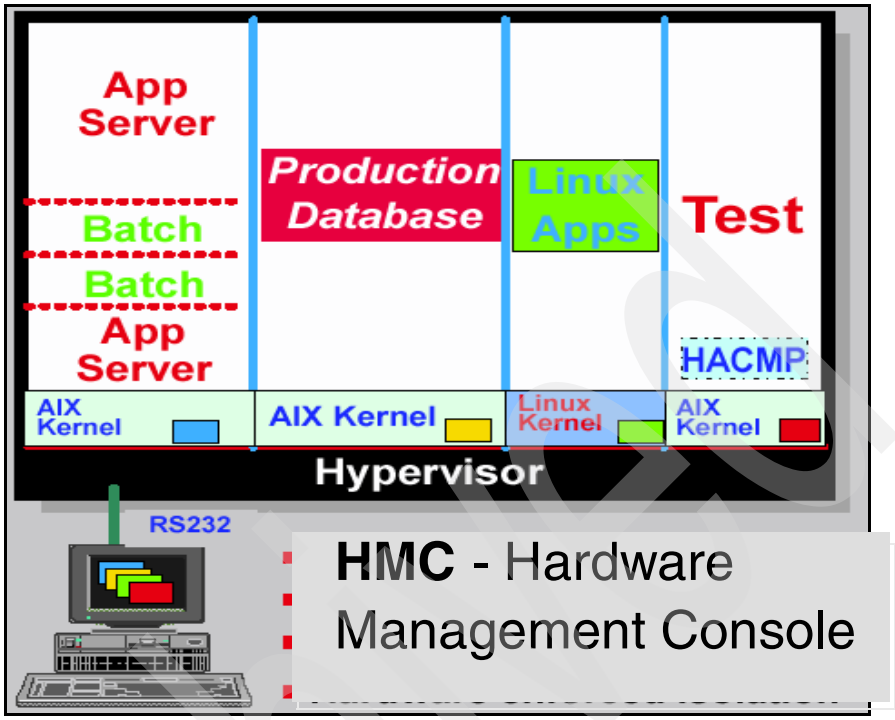

Figure 12-2 Workload Management through HMC

When to use Linux on an LPAR

Looking at Figure 12-3 on page 275 as an example, a typical service provider with a Web hosting environment can have several front-end severs to handle proxy, DNS, and firewall. They may also have small systems that run as Web application servers and a UNIX server that serves as their database. Using LPAR, it is possible to consolidate some of these servers on a single physical system by allocating several Linux instances on the logical partitions, and the remaining capacity can be devoted to AIX instances for the databases for scalability and performance. You can also provide small partitions to be used for testing new versions of applications and operating systems (AIX or Linux) while the production environment continues to run. This eliminates the need for additional servers dedicated to testing, and provides more confidence that the test versions will migrate smoothly into production, because they are tested on the same hardware system.

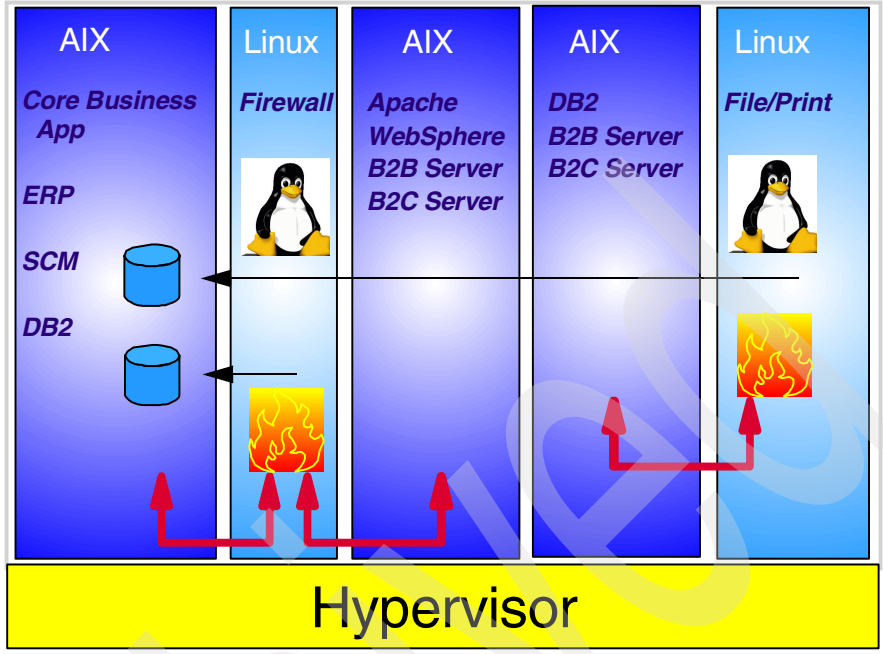

Figure 12-3 AIX and Linux on an LPAR

Some scenarios to consider for the logical partitions:

- Linux is a good candidate to host Web serving, networking, directory and security management, file/print serving, proxy/caching, and firewalls. These applications take advantage of Linux's strength, excellent networking and web serving capabilities, and single application environments.
- Linux is also be a good candidate for mail servers, instant messaging, and groupware functionality. Linux has unique features that are well suited for networking support, directory management and security, and large file systems support.
- Linux has the storage support capabilities to handle database and data warehouse requirements. However, if performance and response time is a factor, then AIX should be considered for these type of applications.
- AIX better handles transaction environments and core application environments that require large amount of storage, as well as outages that are big impact to the business.

Other related information and links

For more information on logical partitioning technology, you can read the following documentations:

- *Partitioning for the IBM @server pSeries 690 System* white paper. The following Web site provides an overview of the logical partitioning technology:

  http://www.ibm.com/servers/eserver/pseries/hardware/whitepapers/lpar.html

- The *IBM @server pSeries 670 and pSeries 690 System Handbook*, SG24-7040. This redbook is a good reference for technical specialists who support the p670 and p690. It also includes information on the architecture of the models and their unique features.
- SuSE for Linux Enterprise Server (SLES) 7 for iSeries and pSeries systems. This Web site contains product details of SLES 7 for iSeries and pSeries systems:

  http://www.suse.com/us/products/suse_business/sles/sles_iSeries_pSeries/index.html

For instructions on installing Linux on a p690 logical partition, read *Linux Applications on pSeries*, SG24-6033; Chapter 5, “Native Linux on pSeries”, describes the procedure to install a native Linux operating system on several pSeries hardware types (including p690) in detail, including some common Linux applications.

12.2 AIX toolbox for Linux applications

To support the best of both worlds, UNIX and Linux, IBM is bringing Linux application interoperability to AIX 5L. In Figure 12-4 on page 277, It shows the two complementary methods in providing Linux application interoperability: Using the AIX toolbox for Linux applications, and including additional Linux-compatible Application Programming Interfaces (APIs), libraries, and commands in AIX 5L for robust performance and implementation.

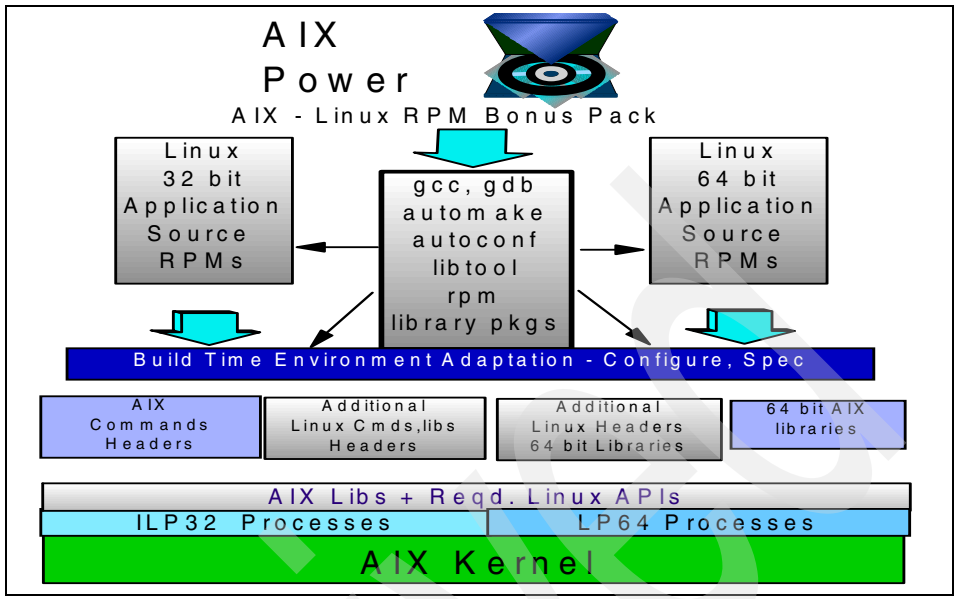

Figure 12-4 AIX toolbox for Linux applications

The AIX toolbox for Linux applications provides the capability to easily compile and run Linux applications, providing the Linux applications full access to AIX functionality and features.

The AIX Toolbox for Linux Application provides Linux source code compatibility by porting open source software and GNU software to run on AIX, that is, GNU application development tools, libraries, shells, utilities, and Graphical User Interface (GUI) desktops. The compiled Linux applications then become AIX binaries.

Here are some of the tools included in the AIX toolbox for Linux applications:

Graphical desktops	KDE and Gnome desktops
Development tools	automake, gcc, g++, gdb, make, autoconf, libtools, m4, patch, rpm-build, and more
Development libraries	libxml, db, gtk+, libtiff, libjgpeg, ncurses, qt, zlib, and more
User interface for desktops	Sawfish and WindowMaker (window manager for XWindow System), Koffice (a set of office application for KDE), the Gnome control-center, enlightenment window manager, and more
Database application	MySQL, including client and libraries

System applications	Samba client, Gnome and KDE utility programs, GnoRPM, mtools, and more
System shells	bash, tcsh, zsh, mc, and the GNU utilities for shell scripts
Programming languages	Python, C and C++ compilers, and PHP
Internet applications	Fetchmail, lynx, ncftp, pine, xchat, ytalk, wget, elm, and more
Archiving applications	tar, cpio, cdrecord, zip, unzip, bzip2, gzip, and more
Graphical games	GNU chess program, KDE games, and the XWindow graphical chessboard
Editing applications	emacs, gnotepad+, vim, and more

The AIX toolbox for Linux applications media is included in all AIX media purchases. All the tools are packaged in RPM format.

Aside from the Toolbox media that is shipped with AIX, you can also get a complete listing, or download all available tools included in the AIX toolbox for Linux applications, at the following URL:

http://www.ibm.com/servers/aix/products/aixos/linux/download.html

or directly from the FTP site:

ftp://ftp.software.ibm.com/aix/freeSoftware/aixtoolbox

The FTP site contains the new and old versions of the tools, the spec files, and the patches that were used to build the tools.

Cryptographic Content (SSL) for certain Toolbox packages can be obtained from the following Web site:

http://www6.software.ibm.com/dl/aixtbx/aixtbx-p

The AIX affinity for Linux provides easy migration of Linux applications to AIX systems using the GNUPro application development tools. With these tools, you can recompile Linux-based Open Source Software (OSS) and GNU software for use on AIX. The Toolbox provides all the tools necessary for effective software compiling and debugging.

For packages governed by open source licenses (such as the GPL license) and requiring source redistribution, you may get the source from the official package site, or you can get it at:

http://www6.software.ibm.com/dl/tbxsrc/tbxsrc-p

The graphical desktops available in the AIX toolbox for Linux applications are composed of different elements, which provide a specific graphical development framework that differs between the desktops.

With the graphical desktops installed, you can enjoy a Linux look and feel interface (KDE and Gnome) and at the same time take advantage of the features and benefits that AIX offers. Figure 12-5 is an example of the KDE Display Manager (KDM) running on AIX after installing and configuring the KDE package that is provided by the Toolbox.

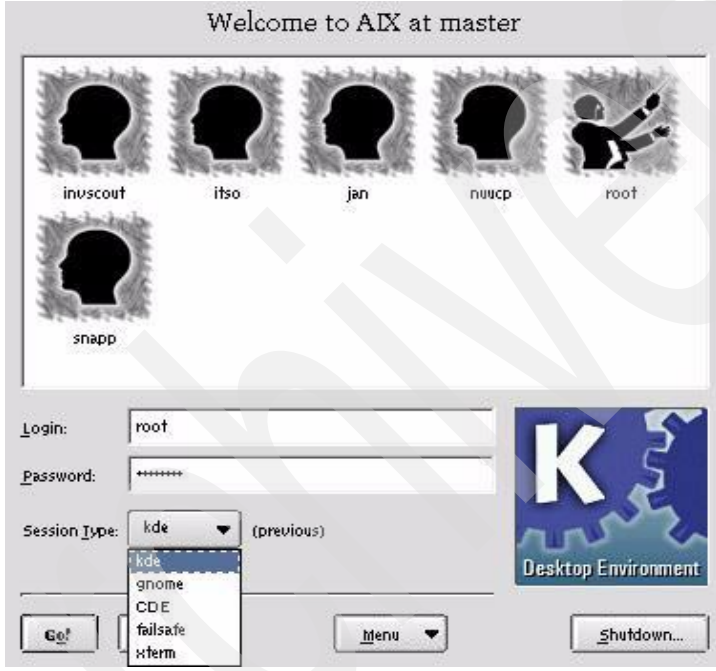

Figure 12-5 KDE Display Manager on AIX

Other related information and links

For more information on the AIX toolbox for Linux applications, you can read the following documents:

- The AIX toolbox for Linux applications Web site. This is the official Web site of the Toolbox. The site includes links for the tools included in the Toolbox, and the licenses associated with the various packages:

  http://www.ibm.com/servers/aix/products/aixos/linux/

- *Linux Applications on pSeries*, SG24-6033. This redbook discusses how Linux plays a major role in IBM's strategy. It also includes installation of the Toolbox and some porting application techniques.
- The *AIX Affinity with Linux Technology* whitepaper. The whitepaper summarizes the Linux application interoperability to AIX 5L and can be found at:

  http://www.ibm.com/servers/aix/products/aixos/linux/affinity_linux.pdf
- The IBM Linux and Open source Web site. This site features all Linux-related products and services that IBM offers. It can be found at:

  http://www.ibm.com/linux

For information on porting applications on UNIX, you can read the following documents:

- *AIX 5L Porting Guide*, SG24-6034. This redbook provide details on the type of problems that are likely to be encountered when porting applications from other UNIX-based platforms to the AIX 5L operating system.
- The UNIX Porting Guide Web site. This Web site provides links and tips on porting applications to different UNIX platforms. It can be found at:

  http://unixporting.com/porting-guides.html

Abbreviations and acronyms

ACL	Access Control List
AH	Authentication Header
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
CA	Certificate Authority
CBC	Cipher Block Chaining
CIFS	Common Internet File Systems
CRL	Certificate Revocation List
DAP	Directory Access Protocol (X.500, DS)
DCE	Distributed Computing Environment
DES	Data Encryption Standard
DH	Diffie-Hellman
DHCP	Dynamic Host Configuration Protocol
DLPAR	Dynamic Logical Partitioning
DN	Distinguished Name (X.500)
DNS	Domain Name System
DNSSEC	DNS Security
DSA	Digital Signature Algorithm
ECB	Electronic Code Block
EIM	Enterprise Identity Mapping
ESP	Encapsulating Security Payload
FQDN	Fully Qualified Domain Name
GID	Group ID
GNU	GNU's Not UNIX
GSS	Generic Security Service
GUI	Graphical User Interface
HMC	Hardware Management Console
HTTP	Hypertext Transfer Protocol
IBM	International Business Machines Corporation
IETF	Internet Engineering Task Force
IHS	IBM HTTP Server
IKE	Internet Key Exchange
IPSec	IP Security
ISAKMP	Internet Security Association and Key Management Protocol
ITSO	International Technical Support Organization
JFS	Journaled File System
KDC	Key Distribution Center
KDM	KDE Display Manager
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
LDIF	LDAP Data Interchange Format
LPAR	Logical Partitioning
MAC	Message Authentication Code
MIB	Management Information Base
MIME	Multipurpose Internet Mail Extensions
MIT	Massachusetts Institute of Technology
NAS	Network Authentication Service
NFS	Network File System
NIS	Network Information Service

NS	Name Server
NSS	Name Service Switch
NTP	Network Time Protocol
OFB	Output Feedback
OSI	Open Systems Interconnect
OSS	Open Source Software
PAM	Pluggable Authentication Module
PEM	Privacy Enhanced Mail
PKI	Public Key Infrastructure
POSIX	Portable Operating System Interface
PTR	Pointer
PSK	Public Shared Key
RFC	Request For Comments
RPC	Remote Procedure Call
RPM	Red Hat Package Manager
RSA	Rivest, Shamir and Adleman (cryptography)
SA	Security Associations
SASL	Simple Authentication and Security Layer
SDK	Software Development Kit
SMB	Server Message Block
SMBFS	Server Message Block File System
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SOA	Start of Authority
SPI	Security Parameter Indices
SSH	Secure Shell
SSL	Secure Socket Layer
SSO	Single Sign-On
SWAT	Samba Web Administration Tool
TCP/IP	Transmission Control Protocol/Internet Protocol
TLS	Transport Layer Security
TSIG	Transaction Signature
UDF	Universal Data Format
UID	User IDentifier
VPN	Virtual Private Network

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 on ordering these publications, see “How to get IBM Redbooks” on page 291.

- *AIX 4.3 Elements of Security Effective and Efficient Implementation*, SG24-5962
- *AIX 5L Differences Guide Version 5.2 Edition,* SG24-5765
- *AIX 5L Porting Guide*, SG24-6034
- *AIX 5L and Windows 2000: Side by Side,* SG24-4784
- *AIX 5L and Windows 2000: Solutions for Interoperability,* SG24-6225
- *The Complete Partitioning Guide for IBM @server pSeries Servers,* SG24-7039
- *A Comprehensive Guide to Virtual Private Networks, Volume III: Cross-Platform Key and Policy Management*, SG24-5309
- *IBM @server pSeries 670 and pSeries 690 System Handbook,* SG24-7040
- *Linux Applications for pSeries,* SG24-6033
- *Managing AIX Server Farms*, SG24-6606
- *Samba Installation, Configuration, and Sizing Guide,* SG24-6004

Other resources

These publications are also relevant as further information sources:

- AIX manuals, which can be found at:

  http://www.ibm.com/servers/aix/library

  These manuals include:

  AIX 5L Version 5.2 Commands Reference, Volume 1

  AIX 5L Version 5.2 General Programming Concepts: Writing and Debugging Programs

AIX 5L Version 5.2 Guide to Printers and Printing

AIX 5L Version 5.2 Installation Guide and Reference

AIX 5L Version 5.2 Installation Guide and Reference: Creating System Backups

AIX 5L Version 5.2 Network Information Services (NIS and NIS+) Guide

AIX 5L Version 5.2 Security Guide

AIX 5L Version 5.2 System Management Guide: Communications and Networks

- Costales, et al., *Sendmail, 3rd Edition*, O'Reilly & Associates, Incorporated, 2002, ISBN 1565928393
- Frisch, *Essential System Administration*, O'Reilly & Associates, Incorporated, 2002, ISBN 0596003439
- Gast, et al., *Network Printing*, O'Reilly & Associates, Incorporated, 2000, ISBN 0596000383
- Maxwell, *UNIX Network Management Tools*, McGraw-Hill Professional, 1999, ISBN 0079137822.
- Stern et al., *Managing NFS and NIS, 2nd Edition*, O'Reilly & Associates, Incorporated, 2001, ISBN 1565925106
- Viega, et al, *Network Security with OpenSSL*, O'Reilly & Associates, Incorporated, 2002, 059600270X

Referenced Web sites

These Web sites are also relevant as further information sources:

- *AIX Affinity with Linux* Technology Paper, found at:

  http://www.ibm.com/servers/aix/products/aixos/linux/affinity_linux.pdf
- AIX ToolBox Download Page: RPM Group Classifications

  http://www.ibm.com/servers/aix/products/aixos/linux/rpmgroups.html
- AIX toolbox for Linux applications Web site

  http://www.ibm.com/servers/aix/products/aixos/linux
- AIX toolbox for Linux applications: Cryptographic Content

  http://www6.software.ibm.com/dl/aixtbx/aixtbx-p
- AIX toolbox for Linux applications: Source images for AIX toolbox for Linux applications

  http://www6.software.ibm.com/dl/tbxsrc/tbxsrc-p

- Bull's Freeware Web site

  http://www.bullfreeware.com

- cf/README for sendmail

  http://www.sendmail.org/m4/intro.html

- *Compliance Defects in Public-Key Cryptography*, found at:

  http://world.std.com/~dtd/compliance/compliance.ps

- Different database methods in Heimdal

  http://www.nada.kth.se/~joda/publications/heimdal-2000.ps

- The Directory Interoperability Forum

  http://www.opengroup.org/directory/

- Directory Services for Linux in comparison with Novell NDS and Microsoft Active Directory

  http://www.daasi.de/staff/nk/thesis/index.html

- Downloading and installing Webmin

  http://www.webmin.com/download.html

- EIM related Web sites:

  http://submit.boulder.ibm.com/pubs/html/as400/cur/v5r2/ic2924/index.htm?info/rzalv/rzalvapis.htm
  http://submit.boulder.ibm.com/pubs/html/as400/cur/v5r2/ic2924/

- Ethereal Network Analyzer Web site

  http://www.ethereal.com/

- Extended attributes and access control lists for Linux

  http://acl.bestbits.at/

- FreeS/WAN: Other configuration possibilities

  http://www.freeswan.org/freeswan_trees/freeswan-1.98b/doc/adv_config.html

- IBM Directory Portfolio

  http://www.ibm.com/software/network/directory/library/index.html

- IBM Directory Server Product Overview

  http://www.ibm.com/software/network/directory/server/index.html

- *IBM Directory Server Version 4.1 Installation and Configuration Guide for Multiplatforms*, found at:

  http://www-3.ibm.com/software/network/directory/library/publications/41/config/ldapinst.htm

- IBM @server Information Center

  http://publib.boulder.ibm.com/eserver/index.html

- The IBM Linux and Open source Web site

  http://www.ibm.com/linux

- IBM Tivoli NetView

  http://www-3.ibm.com/software/sysmgmt/products/support/IBMTivoliNetView.html

- IBM Tivoli Netview PDF

  http://www.tivoli.com/products/documents/datasheets/netview.pdf

- The IMAP Connection

  http://www.imap.org/

- IPSec related Web sites:

  http://lists.freeswan.org/pipermail/users/2002-October/014948.html
  http://lists.freeswan.org/pipermail/users/2002-October/015435.html
  http://www.freeswan.org/freeswan_trees/freeswan-1.98b/doc/examples
  http://citeseer.nj.nec.com/304623.html
  http://www.secforum.com.br/article.php?sid=1033&mode=thread&order=0

- JFS related Web sites:

  http://acl.bestbits.at/
  http://oss.software.ibm.com/developer/opensource/jfs/project/pub/jfsroot.html
  http://lwn.net/Articles/9487/

- Journaled File System Project: Patches

  http://oss.software.ibm.com/developerworks/patch/?group_id=35

- Journaled File System Technology for Linux

  http://www-124.ibm.com/jfs

- Kerberos FAQ

  http://www.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html

- Kerberos/GSSAPI Support in OpenSSH

  http://www.sxw.org.uk/computing/patches/openssh.html

- Kerberos: The Network Authentication Protocol

  http://web.mit.edu/kerberos/www/

- LDAP documentation

  http://www.umich.edu/~dirsvcs/ldap/doc

- LDAP Schema Viewer

  http://ldap.akbkhome.com/
- *Limitations of the Kerberos Authentication System*, found at:

  http://www.research.att.com/~smb/papers/kerblimit.usenix.pdf
- Linux LDAP HOWTO

  http://en.tldp.org/HOWTO/LDAP-HOWTO/
- Linux NIS HOWTO

  http://www.linux-nis.org/nis-howto/
- Linux NFS HOWTO

  http://nfs.sourceforge.net/nfs-howto/index.html
- A Linux-PAM page

  http://www.kernel.org/pub/linux/libs/pam/
- Linux Printing HOWTO

  http://www.linuxprinting.org/howto/how.html
- Linux on pSeries Web site

  http://www.ibm.com/servers/eserver/pseries/linux/
- LPRNg

  http://sourceforge.net/projects/lprng/
- Monitoring with tcpdump

  http://www-iepm.slac.stanford.edu/monitoring/passive/tcpdump.html
- The NTOP Community Page

  http://snapshot.ntop.org
- NTOP Web site

  http://www.ntop.org
- OpenLDAP home page

  http://www.openldap.org/
- OpenLDAP, OpenSSL, SASL, and KerberosV HOWTO

  http://www.bayour.com/LDAPv3-HOWTO.html
- OpenLDAP RPMs

  ftp://ftp.software.ibm.com/aix/freeSoftware/aixtoolbox/RPMS/ppc/openldap
- OpenSSH on AIX Images Project

  http://oss.software.ibm.com/developerworks/projects/opensshi/

- OpenSSL related Web sites:

  http://www.unixreview.com/documents/s=2428/uni1025067917864/
  http://wp.netscape.com/eng/ssl3/draft302.txt
  http://www.nisu.org/tut/ossl/indice
  http://developer.netscape.com/docs/manuals/security/sslin/contents.htm

- PADL Software Pty Ltd: Heimdal

  http://www.padl.com/Research/Heimdal.html

- PADL Software Pty Ltd: Migration Tools, PAM LDAP, and NSS LDAP

  http://www.padl.com

- PADL.com tar respository

  ftp://ftp.padl.com/pub/

- PAM

  http://www.softpanorama.org/Security/pam.shtml

- PAM authenticator module for Squid

  http://squid.sourceforge.net/hno/pam_auth-2.0.c

- *Partitioning for the IBM @server pSeries 690 System* white paper, found at:

  http://www.ibm.com/servers/eserver/pseries/hardware/whitepapers/lpar.html

- *Password Security: A Case History*, found at:

  http://citeseer.nj.nec.com/morris79password.html

- rdist Web site

  http://www.redhat.com/swr/i386/rdist-6.1.5-14.i386.html

- *Red Hat Linux 8.0 Release Notes*, found at:

  http://www.redhat.com/docs/manuals/linux/RHL-8.0-Manual/pdf/rhl-relnotes-x86-en-80.pdf

- RFC1094: NFS: Network File System Protocol Specification

  http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc1094.html

- RFC2222: Simple Authentication and Security Layer (SASL)

  ftp://ftp.rfc-editor.org/in-notes/rfc2222.txt

- RFC2251: Lightweight Directory Access Protocol (Version 3)

  ftp://ftp.rfc-editor.org/in-notes/rfc2251.txt

- RFC2252: Lightweight Directory Access Protocol (Version 3): Attribute Syntax Definitions

  ftp://ftp.rfc-editor.org/in-notes/rfc2252.txt

- RFC2253: Lightweight Directory Access Protocol (Version 3): UTF-8 String Representation of Distinguished Names

  ftp://ftp.rfc-editor.org/in-notes/rfc2253.txt
- RFC2254: The String Representation of LDAP Search Filters

  ftp://ftp.rfc-editor.org/in-notes/rfc2254.txt
- RFC2255: The LDAP URL Format

  ftp://ftp.rfc-editor.org/in-notes/rfc2255.txt
- RFC2256: A Summary of the X.500(96) User Schema for use with LDAP Version 3

  ftp://ftp.rfc-editor.org/in-notes/rfc2256.txt
- RFC2743: Generic Security Service Application Program Interface (Version 2)

  ftp://ftp.rfc-editor.org/in-notes/rfc2743.txt
- RFC2829: Authentication Methods for LDAP

  ftp://ftp.rfc-editor.org/in-notes/rfc2829.txt
- RFC2830: Lightweight Directory Access Protocol (Version 3): Extension for Transport Layer Security

  ftp://ftp.rfc-editor.org/in-notes/rfc2830.txt
- RFC3377: Lightweight Directory Access Protocol (Version 3): Technical Specification

  ftp://ftp.rfc-editor.org/in-notes/rfc3377.txt
- RPM resource LPRng

  http://www.rpmfind.net/linux/rpm2html/search.php?query=LPRng
- RPM resource NFS

  http://speakeasy.rpmfind.net/linux/rpm2html/search.php?query=NFS
- RPM resource Samba

  http://www.rpmfind.net/linux/rpm2html/search.php?query=Samba+
- RPM resource Sendmail

  http://speakeasy.rpmfind.net/linux/rpm2html/search.php?query=Sendmail
- RPM resource sendmail-cf

  http://www.rpmfind.net/linux/rpm2html/search.php?query=sendmail-cf
- RPMfind Web site

  http://www.rpmfind.net/linux/RPM

- rsync Web site

  http://samba.anu.edu.au/ftp/rsync/binaries
- Samba Web site

  http://www.samba.org/
- Sendmail Web site

  http://www.sendmail.org
- *Single Sign-On*, found at:

  http://www.tcm.hut.fi/Opinnot/Tik-110.501/1997/single_sign-on.html
- *The SLAPD Administrator's Guide*, found at:

  http://www.umich.edu/~dirsvcs/ldap/doc/
- Squid related software

  http://squid.sourceforge.net/hno/software.html
- SuSE for Linux Enterprise Server (SLES) 7 for iSeries and pSeries systems

  http://www.suse.com/us/products/suse_business/sles/sles_iSeries_pSeries/index.html
- tcpdump utility

  http://www.tcpdump.org/
- UDF support on Linux

  http://www.trylinux.com/projects/udf/
- *Unified Login With Pluggable Authentication Modules (PAM)*, found at:

  http://www.opengroup.org/tech/rfc/rfc86.0.html
- *Unified Single Sign-On*, found at:

  http://www.tml.hut.fi/Opinnot/Tik-110.501/1998/papers/3singlesignon/singlesign-on.htm
- The UNIX Porting Guide Web site

  http://unixporting.com/porting-guides.html
- Using SSL with Webmin

  http://www.webmin.com/ssl.html
- What are Access Control Lists (ACLs)?

  http://acl.bestbits.at/about-acl.html

How to get IBM Redbooks

You can order hardcopy Redbooks, as well as view, download, or search for Redbooks at the following Web site:

ibm.com/redbooks

You can also download additional materials (code samples or diskette/CD-ROM images) from that site.

IBM Redbooks collections

Redbooks are also available on CD-ROMs. Click the CD-ROMs button on the Redbooks Web site for information about all the CD-ROMs offered, as well as updates and formats.

Index

Symbols

Numerics

A

B

C

D

E

L

Q

R

S

T

U

V

W

X

Y

Z

AIX and Linux Interoperability

(0.5” spine)
0.475”<->0.875”
250 <-> 459 pages

AIX and Linux Interoperability

Effective centralized user management in AIX 5L and Linux environments

Sharing files and printers between AIX 5L and Linux systems

Learn interoperable networking solutions

This IBM Redbook discusses interoperability in terms of UNIX to UNIX cross platform data sharing and user/system management. This redbook also demonstrates the similarities and differences between the AIX and Linux operating systems.

This redbook is intended to help IT specialists who have AIX systems and Linux systems in their environments understand how to integrate and optimize AIX systems in a Linux environment and share AIX resources with Linux systems. We have focused our descriptions on the following areas:

- User management
- Networking
- Data sharing
- System management
- Security
- Linux for IBM @server pSeries and RS/6000

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-6622-00 ISBN 0738426822