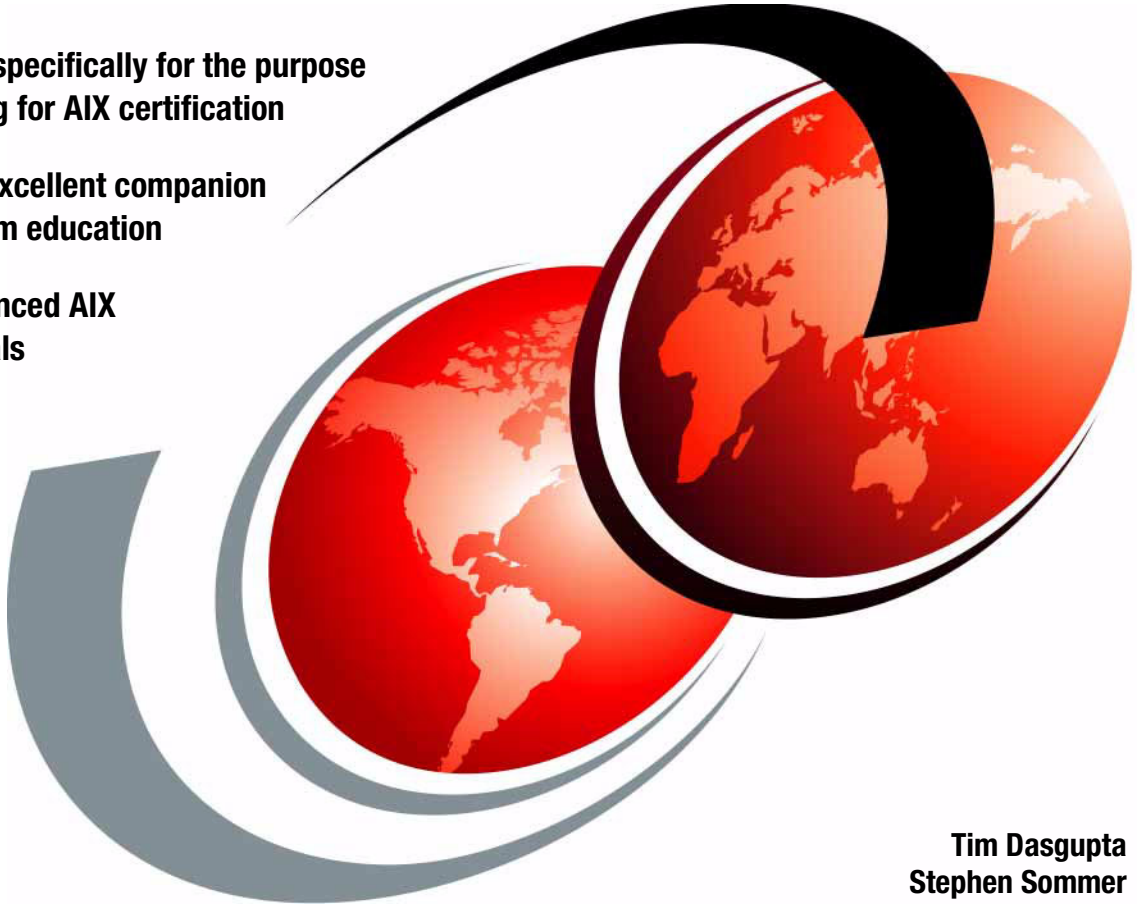
IBM

IBM Certified Specialist

IBM eserver Certification Study Guide - AIX 5L Communications

- Developed specifically for the purpose of preparing for AIX certification
- Makes an excellent companion to classroom education
- For experienced AIX professionals



Tim Dasgupta
Stephen Sommer

Redbooks

ibm.com/redbooks

International Technical Support Organization

IBM @server Certification Study Guide - AIX 5L Communications

December 2002

SG24-6186-01

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

Second Edition (December 2002)

This edition applies to AIX 5L Version 5.1 (5765-E61) and subsequent releases running on an IBM @server pSeries or RS/6000 server.

This document was updated on January 6, 2004.

Contents

Figures

Tables

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:

AIX®	Micro Channel®	QMF™
AIX 5L™	PowerPC®	Redbooks(logo)™ 
IBM®	PowerPC Reference Platform®	Redbooks™
DFS™	PS/2®	RS/6000®
@server™	pSeries™	SP™
LoadLeveler®	PTX®	Versatile Storage Server™

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

Domino™

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

The AIX and IBM @server pSeries Certifications, offered through the Professional Certification Program from IBM, are designed to validate the skills required of technical professionals who work in the powerful and often complex environments of AIX and IBM @server pSeries. A complete set of professional certifications are available. They include:

- IBM Certified AIX User
- IBM Certified Specialist - Business Intelligence for RS/6000
- IBM Certified Specialist - Domino for RS/6000
- IBM @server Certified Specialist - p690 Solutions Sales
- IBM @server Certified Specialist - p690 Technical Support
- IBM @server Certified Specialist - pSeries Sales
- IBM @server Certified Specialist - pSeries AIX System Administration
- IBM @server Certified Specialist - pSeries AIX System Support
- IBM @server Certified Specialist - pSeries Solution Sales
- IBM Certified Specialist - RS/6000 SP and PSSP V3
- IBM Certified Specialist - Web Server for RS/6000
- IBM @server Certified Specialist - pSeries HACMP for AIX
- IBM @server Certified Advanced Technical Expert - pSeries and AIX 5L

Each certification is developed by following a thorough and rigorous process to ensure the exam is applicable to the job role and is a meaningful and appropriate assessment of skill. Subject matter experts who successfully perform the job participate throughout the entire development process. They bring a wealth of experience to the development process, making the exams much more meaningful than the typical test that only captures classroom knowledge and ensuring the exams are relevant to the *real world*. Thanks to their effort, the test content is both useful and valid. The result of this certification is the value of appropriate measurements of the skills required to perform the job role.

This IBM Redbook is designed as a study guide for professionals wishing to prepare for the AIX 5L Communications certification exam as a selected course of study in order to achieve the IBM @server Certified Advanced Technical Expert - pSeries and AIX 5L certification.

This redbook provides a combination of theory and practical experience needed for a general understanding of the subject matter. It also provides sample questions that will help in the evaluation of personal progress and provide familiarity with the types of questions that will be encountered in the exam.

This publication does not replace practical experience, nor is it designed to be a stand-alone guide for any subject. Instead, it is an effective tool that, when combined with education activities and experience, can be a very useful preparation guide for the exam.

For additional information about certification and instructions on how to register for an exam, visit our Web site at:

http://www.ibm.com/certify

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.

Tim Dasgupta is an IBM Certified AIX Advanced Technical Expert (CATE). He works as a Senior Systems Architect at IBM Global Services in Canada. He has over eight years of experience in the areas of AIX, RS/6000, and pSeries. He is currently the Team Leader of Midrange Architecture Group in Montreal, Canada.

Stephen Sommer is an IBM Certified AIX Advanced Technical Expert (CATE), AIX Version 4.3.3 and 5.1. He works as a Senior IT Specialist at Faritec Services, an IBM Business Partner in Johannesburg, South Africa. He has eight years of experience in Midrange Support for AIX, RS/6000 and pSeries both in South Africa and the United Kingdom.

The authors of the first edition were:

Thomas Herlin	IBM Denmark
André de Klerk	IBM South Africa
Thomas C. Cederlöf	IBM Sweden
Tomasz Ostaszewski	Prokom Software SA in Poland

The project that produced this publication was managed by:

Scott Vetter	International Technical Support Organization, Austin Center

Special thanks to:

Shannan L DeBrule	IBM Atlanta
Darin Hartman	Program Manager, AIX Certification

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

Jesse Alcantar, Greg Althaus, Karl Borman, Larry Brenner, Greg Flaig, Shawn Mullen, Brian Nicholls
IBM Austin

Michelle Page-Rivera
IBM Atlanta

Edward Geraghty
IBM Boston

Federico Vagnini
IBM Italy

Adnan Ikram
IBM Pakistan

Christopher Snell
IBM Raleigh

Peter Mayes
IBM U.K.

Malin Cederberg and Robert Olsson
ILS Sweden

Yesid Jaramillo
Sistemas Integrales de Informactica S.A. Columbia

Karl Jones
Systems Analyst - Designed Business Systems

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

Certification overview

This chapter provides an overview of the skill requirements needed to obtain an IBM Advanced Technical Expert certification. The following chapters are designed to provide a comprehensive review of specific topics that are essential for obtaining the certification:

IBM @server Certified Advanced Technical Expert - pSeries and AIX 5L

This level certifies an advanced level of pSeries and AIX knowledge and understanding, both in breadth and depth. It verifies the ability to perform in-depth analysis, apply complex AIX concepts, and provide resolution to critical problems, all in a variety of areas within AIX, including the hardware that supports it.

1.1 Certification requirements

To attain the IBM @server Certified Advanced Technical Expert - pSeries and AIX 5L certification, you must pass four tests.

One test is the prerequisite for certification as a Specialist in either pSeries AIX System Administration or pSeries AIX System Support. The other three tests are selected from a variety of pSeries and AIX topics. These requirements are explained in greater detail in the sections that follow.

1.1.1 Required prerequisite

Prior to attaining the IBM @server Certified Advanced Technical Expert - pSeries and AIX 5L certification, you must be certified as either an:

- IBM @server Certified Specialist - pSeries AIX System Administration

  or

- IBM @server Certified Specialist - pSeries AIX System Support

1.1.2 Recommended prerequisite

A minimum of six to 12 months' experience in performing in-depth analysis and applying complex AIX concepts in a variety of areas within AIX is a recommended prerequisite.

1.1.3 Information and registration for the certification exam

For the latest certification information, see the following Web site:

http://www.ibm.com/certify

1.1.4 Core requirements

You must select three of the following exams. You will receive a Certificate of Proficiency for tests when passed.

AIX 5L Installation and System Recovery

Test 233 was developed for this certification.

Preparation for this exam is the topic of *IBM @server Certification Study Guide - AIX 5L Installation and System Recovery*, SG24-6183.

AIX 5L Performance and System Tuning

Test 234 was developed for this certification.

Preparation for this exam is the topic of *IBM @server Certification Study Guide - AIX 5L Performance and System Tuning*, SG24-6184.

AIX 5L Problem Determination Tools and Techniques

Test 235 was developed for this certification.

Preparation for this exam is the topic of *IBM @server Certification Study Guide - AIX 5L Problem Determination Tools and Techniques*, SG24-6185.

AIX 5L Communications

The following objectives were used as a basis when the certification test 236 was developed. Some of these topics have been regrouped to provide better organization when discussed in this publication.

Preparation for this exam is the topic of this publication.

Section I - Planning

1. Determine network architecture:
   a. Determine which type of adapter(s) will be used in the network.
   b. Identify the network cable type to be used for the network.
   c. Determine slot location and interface name for the adapter that is being used.
2. Determine address and naming scheme, subnet masks, and routes:
   a. Obtain IP address to be used for the adapter from network administrator.
   b. Determine network class for IP address being used from first octet of address.
   c. Determine required network class based on number of desired subnets and hosts per network.
   d. Determine network mask to be used to satisfy subnet requirements.
   e. Identify default gateway that will be used to get outside local network.
   f. Decide if this machine will be an IP gateway.
   g. Select a host name for the machine and possible aliases.
   h. Determine names if there are additional adapters.
3. Identify required services:
   a. Decide if this machine will be a DNS server or client.

b. Determine if this machine will be an NFS, NIS, Mail, NTP, or DHCP server or client.

c. Evaluate which remote services this machine will support (**rlogin**, **ftp**, **tftp**, **bootp, ppp**)

Section II - Network Configuration

1. Verify communication adapter is available:
   a. Run **lsdev -Cc adapter** to see which adapters are available
   b. Install device drivers for network adapter if it is not available.
2. Configure adapter and verify connectivity:
   a. Adjust network adapter attributes.
   b. Enter the host name, IP address, subnet mask, default gateway, domain name, and DNS server address on SMIT TCP/IP minimum configuration screen.
   c. Run **netstat -in** and **ifconfig** to verify proper configuration.
   d. Run the **ping** command on another machine on the local network to verify connectivity.
3. Add static routes:
   a. Determine the static network routes to add due to network segments not accessible through the default route.
   b. Configure the static routes using SMIT route and providing the destination address, network mask, next hop, and metric of the route.
4. Configure network options:
   a. Determine which network options need to be adjusted to improve network performance.
   b. Edit /etc/rc.net to enable network options at system boot.

Section III - Services Configuration

1. DNS
   a. Select lookup order using either $NSLOOKUP or /etc/netsvc.com
   b. Update /etc/resolv.conf with correct information.
   c. Determine domains to be served from this server.
   d. Set up a primary DNS server.
   e. Edit /etc/rc.tcpip to autostart named at system boot.
   f. Verify Domain Name System services are working properly.

2. NFS
    a. Configure the appropriate number of nfsd and biod daemons to start.
    b. Configure file systems to export to NFS clients.
    c. Configure file systems to NFS mount from remote servers.
3. NIS
    a. Determine if this machine is to be a client or server.
    b. Set yp domain name using **domainname** command.
    c. Determine which configuration files need to be served.
    d. Create map files for NIS server.
    e. Configure ypbind, ypserv, and yppasswdd daemons to start.
4. DHCP
    a. Enable dhcpsd in /etc/rc.tcpip to autostart on system boot.
5. MAIL
    a. Verify /etc/sendmail.cf contains correct host and domain information.
    b. Update /etc/aliases with appropriate user redirects.
    c. Refresh sendmail daemon.
6. Remote Services (inetd.conf)
    a. Update /etc/inetd.conf to turn desired services on or off.
    b. Refresh inetd.
7. SNMP
    a. Edit /etc/rc.net to enable SNMP.
    b. Update /etc/snmpd.conf to set community names and servers to receive SNMP traps.

Section IV - Security

1. Assess Risk Analysis
    a. Disable all daemons not directly required on this server.
    b. Configure remote access authorization in /etc/host.equiv and .rhost files.
    c. Determine if security services are current and no maintenance is required.
2. Security Options
    a. Consider other independent security control systems that may improve security.
    b. Decide if Kerberos will be used.

Section V - Asynchronous Communication

1. Configuration of asynchronous adapters:
   a. Install necessary device drivers and updates for asynchronous adapters.
   b. Make sure asynchronous adapters are available.
2. Configure TTYs as terminals, printers, or modems:
   a. Attach TTY to desired port.
   b. Use smit to add TTY with desired configuration options.
   c. Verify TTY is working properly.
3. Configure uucp, cu, ppp, ate, and so on:
   a. Configure appropriate device files for each desired service.
   b. Verify each desired service is working properl.

Section VI - Troubleshooting

1. Identify connectivity problems:
   a. Use **`netstat`** or **`ifconfig`** to verify proper IP setup for local adapter
   b. Use **`netstat`**, **`ping`**, and **`traceroute`** to isolate routing issues
   c. Use **`iptrace`** to verify packet delivery
2. Identify and resolve network performance issues:
   a. Use **`netstat`** to gather statistics
   b. Update **`no`** and **`chattr`** to adjust network and device options
3. Determine NFS mounting and performance problems:
   a. Adjust number of biod and nfsd daemons if necessary
   b. Adjust NFS options with **`nfso`** command if necessary
   c. Verify file systems are exported from server
   d. Verify name resolution
4. Identify modem connectivity problems:
   a. Verify TTY for modem is available
   b. Check login enable attribute
   c. Check port settings
   d. Use **`cu`** to verify configuration with modem

pSeries HACMP for AIX

Test 187 was developed for this certification.

Preparation for this exam is the topic of *IBM @server Certification Study Guide - pSeries HACMP for AIX*, SG24-6187.

RS/6000 SP and PSSP V3.1

Test 188 was developed for this certification.

Preparation for this exam is the topic of *IBM Certification Study Guide - RS/6000 SP*, SG24-5348.

p690 Technical Support

Test 195 was developed for this certification.

An IBM Redbook is planned for first quarter 2003 on this subject.

1.2 Certification education courses

Courses are offered to help you prepare for the certification tests. For a current list, visit the following Web site, locate your test number, and select the education resources available:

http://www.ibm.com/certify/tests/info.shtml

2

Network interfaces and protocols

One of the most important aspects of the modern business machine is the network connectivity. With small businesses setting up networks that range from two or three workstations through global corporations that connect tens of thousands of workstations to hundreds of servers, often of different platforms, it is critical to understand the differences between the different protocols and interfaces. It is not uncommon for businesses to have various platforms, each running a different network protocol and interfacing with the other systems through an intermediate system.

2.1 Networking basics

The most common way of describing a network is the International Standards Organization's Open Systems Interconnection (OSI) Reference Model, also referred to as the OSI seven-layer model. The seven layers of the OSI model are as follows:

7 Application

6 Presentation

5 Session

4 Transport

3 Network

2 Data Link

1 Physical

Levels 1 through 3 are network specific, and will differ depending on what physical network you are using. Levels 4 through 7 comprise network-independent, higher level functions. Each layer describes a particular function (instead of a specific protocol) that occurs in data communications. The seven layers function in order from highest to lowest are defined as follows:

Application	Comprises the applications that use the network.
Presentation	Ensures that data is presented to the applications in a consistent fashion.
Session	Manages the connections between applications.
Transport	Ensures error-free data transmission.
Network	Manages the connections to other machines on the network.
Data Link	Provides reliable delivery of data across the Physical Layer (which is usually inherently unreliable).
Physical	Describes the physical media of the network. For example, the GigaBit Ethernet cable is part of the Physical Layer.

While the OSI Reference Model is useful for discussing networking concepts, many networking protocols do not closely follow the OSI model. For example, when discussing Transmission Control Protocol/Internet Protocol (TCP/IP), the Application and Presentation Layer functions can be combined into a single level, as can the Session and Transport Layers, as well as the Data Link and Physical Layers.

Each layer in the OSI model defines a communications protocol with the corresponding layer on the remote machine. The layers pass data only to the layers immediately above and below. As shown in Figure 2-1, each layer adds its own header (and, in the case of the Data Link Layer, footer) information, effectively encapsulating the information received from the higher layers. Ethernet and token-ring are the most common network interfaces; however, there are others that exist.

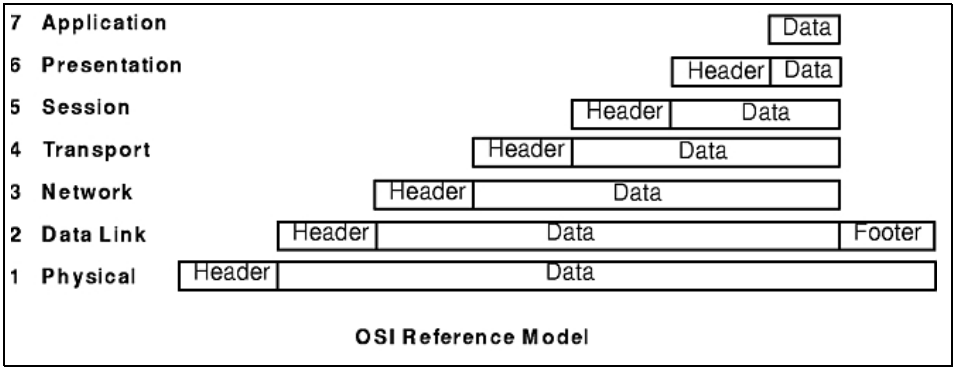


Figure 2-1 OSI reference model

Token-ring, originally developed by IBM, uses a token-passing mechanism to regulate traffic on the ring. It is defined by the IEEE 802.5 standard.

Ethernet is a broadcast-based protocol that uses collision detection and avoidance for network traffic regulation. Ethernet, defined by the IEEE 802.3 standard, was originally developed by the Xerox Palo Alto Research Center.

FDDI is similar to token-ring in that it also passes a token over a ring, except that it is a fiber optic ring.

Serial Line Internet Protocol (SLIP) and Point-to-Point Protocol (PPP) are protocols that use serial ports and modems to communicate.

Asynchronous Transfer Mode (ATM) is a full duplex cell-switching protocol that supports end-to-end connections.

2.2 Ethernet standards overview

Ethernet is the most popular type of network in the world. It is popular because it is easy to implement, and the cost of ownership is relatively lower than that of other technologies. It is also easy to manage, and the Ethernet products are readily available.

2.2.1 Access method

Hosts send messages on an Ethernet LAN using a Network Interface Layer protocol, with carrier sense and multiple access with collision detect (CSMA/CD). The CSMA/CD ensures that all devices communicate on a single medium, but that only one transmits at a time, and that they all receive simultaneously. If two devices try to transmit at the same instant, the transmit collision is detected, and both devices wait a random period before trying to transmit again using a *backoff algorithm* shown in Figure 2-2.

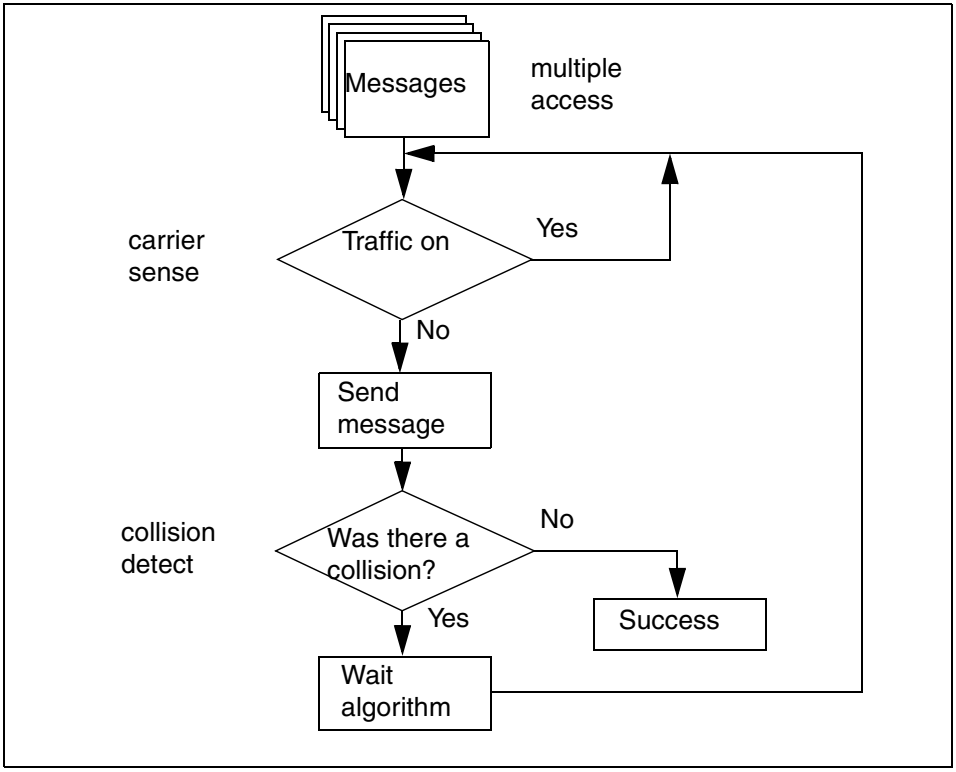


Figure 2-2 CSMA/CD algorithm

The chance of a collision depends on the following:

- The number of workstations on the network. The more workstations, the more likely collisions will occur.
- The length of the network. The longer the network, the bigger the chance for collisions due to the time needed for signals to reach all devices.
- The length of the data packet, that is, the MTU size. A larger packet length takes a longer time to transmit, which increases the chance of a collision.

The collision statistics for the particular Ethernet interface can be obtained by the **entstat** command:

```
# entstat -d en0
.....
Single Collision Count: 12
Multiple Collision Count: 11
.....
IBM PCI Ethernet Adapter Specific Statistics:
---------------------------------------------
Chip Version: 16
Packets with Transmit collisions:
 1 collisions: 12          6 collisions: 0          11 collisions: 0
 2 collisions: 2           7 collisions: 2          12 collisions: 0
 3 collisions: 3           8 collisions: 2          13 collisions: 0
 4 collisions: 0           9 collisions: 1          14 collisions: 0
 5 collisions: 0          10 collisions: 1          15 collisions: 0
```

2.2.2 Fast Ethernet

The Fast Ethernet, or the IEEE 802.3u standard, is 10 times faster than the 10 Mbps Ethernet. The cabling used for Fast Ethernet is 100BaseTx, 100BaseT4 and the 100BaseFx. The framing used in Fast Ethernet is the same as that used in Ethernet. Therefore, it is very easy to upgrade from Ethernet to Fast Ethernet. Because the framing and size are the same as that of Ethernet and the speed has been increased 10 times, the length of the network must be reduced, or else the collision would not be detected and would cause problems to the network.

2.2.3 Gigabit Ethernet

The Gigabit Ethernet, or IEEE 802.3z standard, is 10 times faster than the Fast Ethernet. To accelerate speeds from 100-Mbps Fast Ethernet to 1 Gbps, several changes need to be made to the physical interface. It has been decided that Gigabit Ethernet will look identical to Ethernet from the Data Link Layer upward. The physical media can be either a copper cable, but with shorter lengths, or a fiber optic cable.

2.3 Asynchronous Transfer Mode (ATM)

ATM is a high performance, cell-switching, connection-oriented technology. In ATM networks, end stations attach to the network using dedicated full-duplex connections. ATM can be used for voice and video as well as multimedia applications. Figure 2-3 shows an example of how to set up a network using ATM.

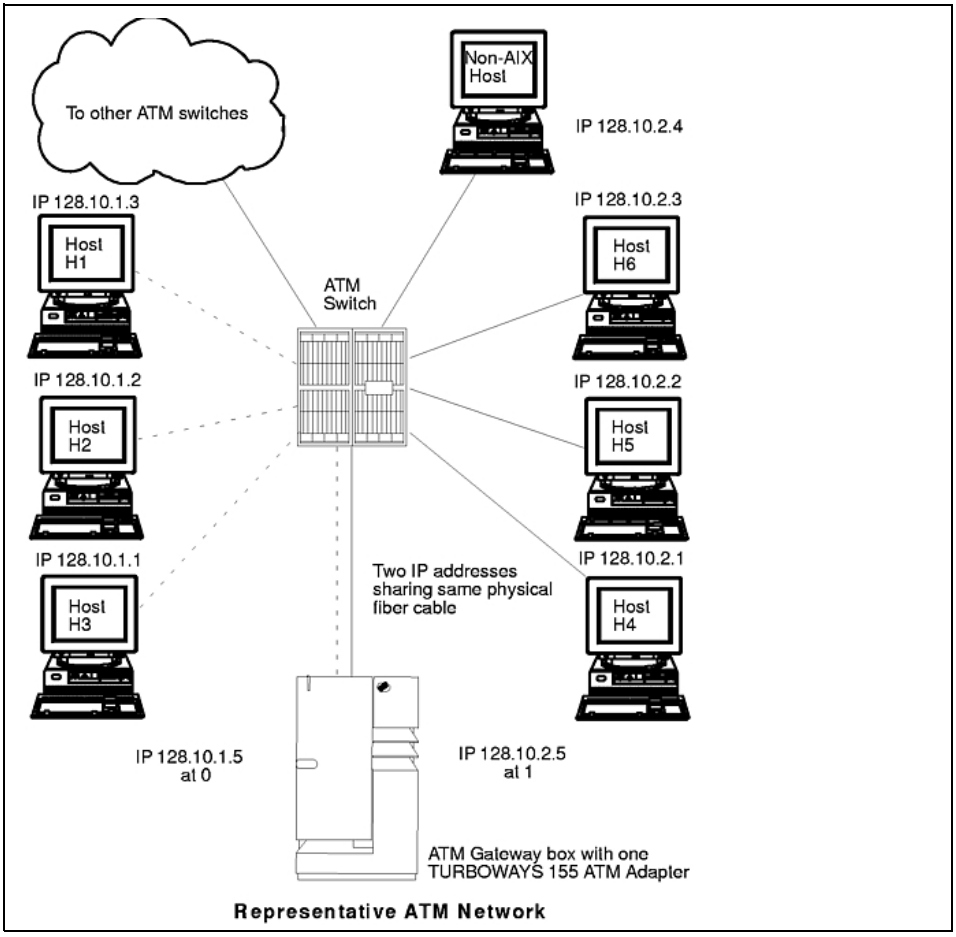


Figure 2-3 A representative ATM network

2.3.1 TCP/IP over ATM

The Internet Engineering Task Force RFC1577: *Classical IP and ARP over ATM* standard specifies the mechanism for implementing Internet Protocol (IP) over ATM. Since ATM is connection-oriented technology and IP is a datagram-oriented technology, mapping the IP over ATM is not trivial.

In general, the ATM network is divided into logical IP subnetworks (LISs). Each LIS is comprised of some number of ATM stations. LISs are analogous to traditional LAN segments and are interconnected using routers. A particular adapter (on an ATM station) can be part of multiple LISs. This feature may be very useful for implementing routers.

RFC1577 specifies RFC1483, which specifies Logical Link Control/Sub-Network Access Protocol (LLC/SNAP) encapsulation as the default. In Permanent Virtual Circuits (PVC) networks for each IP station, all PVCs must be manually defined by configuring VPI:VCI (VP and VC identifiers) values. If LLC/SNAP encapsulation is not being used, the destination IP address associated with each VPI:VCI must be defined. If LLC/SNAP encapsulation is being used, the IP station can learn the remote IP address by an InARP mechanism. For Switched Virtual Circuits (SVC) networks, RFC1577 specifies an ARP server per LIS. The purpose of the ARP server is to resolve IP addresses into ATM addresses without using broadcasts. Each IP station is configured with the ATM address of the ARP server. IP stations set up SVCs with the ARP server, which in turn sends InARP requests to the IP stations. Based on the InARP reply, an ARP server sets up IP-to-ATM address maps. IP stations send ARP packets to the ARP server to resolve addresses, which returns ATM addresses.

IP stations then set up an SVC to the destination station and data transfer begins. The ARP entries in IP stations and the ARP server age are based on a well-defined mechanism. For both the PVC and SVC environments, each IP station has at least one virtual circuit per destination address.

The TCP/IP and ARP services would need to be started for ATM to work.

2.4 Network media

Every transmission standard has some restrictions related to hardware capability. Even the quality of the cables can dictate the quality of the network solution.

10Base2

This is the lowest-cost form of networking. The system uses a BNC connector and needs to be terminated on both ends of the cable, irrespective of the number of users between the two termination points. One disadvantage is that if there is a problem anywhere in the network, it is very difficult to localize the problem to a specific segment to correct the problem. Below are some limitations for 10Base2 networks:

- The maximum length per segment is 185 meters or 607 feet.
- Maximum of 30 nodes per unrepeated network segment.
- Runs on RG-58 (thin coaxial) cable. Coax cable may require terminator resistors.
- Connects using BNC connectors.

10Base5

This standard runs on a thicker coaxial cable than 10Base2 and is better suited for the network backbone rather than the actual user segments. Below are some limitations for 10Base5 networks:

- Maximum length per segment is 500 meters or 1640 feet.
- Maximum of 100 users/devices per unrepeated network segment.
- Runs on RG-8 coaxial (thicknet) cable. Coax cable may require terminator resistors and disconnecting a coax cable may have negative consequences on the entire network.
- Connects using AUI connectors.

10BaseT

This is normally the best price versus performance option. It is a bit more expensive than either 10Base2 or 10Base5; however, the termination is done either on the network card or the hub, which makes reliability and scalability simpler. Below are some limitations for 10BaseT networks:

- Maximum length is 150 meters or 492 feet per segment, depending on cable specifications.
- Maximum of 1024 nodes per network.
- Runs on unshielded twisted pair (UTP) cable.
- Connects using RJ-45 connectors.

10BaseF

Using fiber optic is the most expensive option when setting up a network. Fiber optic cable has an advantage of being able to be run next to electrical lines because of lack of electromagnetic interference. This option will mostly be used when connecting two buildings to the same LAN, because it is not feasible to use it within a standard office environment. Even though a maximum of 2 kilometers can be reached per segment, this can depend on the equipment being used. Below are some limitations for 10BaseF networks:

- A maximum length of 2000 meters or 6562 feet per segment depending on equipment being used.
- Maximum of 1024 users/devices per network. This is the Ethernet user/device limit.
- Runs on fiber optic cable.
- Rough handling can affect fiber optic cable.

100BaseFx

The fiber optic version of 100BaseFx is also a rather expensive solution for networking in a small LAN environment, but could be used to connect two or more buildings on one site together. Below are some limitations for 100BaseFx networks:

- A maximum length of 500 meters or 1640 feet per segment depending on equipment being used.
- Maximum of 1024 users/devices per network. This is the Ethernet user/device limit.
- Runs on fiber optic cable.
- Rough handling can affect fiber optic cable.

100BaseTx

This standard is compatible with the 10BaseT, so it has become the most popular of the 100 Mbps standards. This makes it a less expensive option for implementation, since an existing network structure can be used to upgrade to the faster standard. Below are some limitations for 100BaseTx networks:

- Maximum length up to 150 meters or 492 feet per segment, depending on cable specifications.
- Maximum of two nodes per segment and 1024 nodes per network.
- Runs on unshielded twisted pair (UTP) cable.
- Connects using RJ-45 connectors.

100BaseT4

Although the 100BaseT4 is similar to the 100BaseT, it uses a four-pair twisted pair cable instead of the two-pair twisted pair of the 100BaseT standard and is not compatible with 10BaseTx. This incompatibility has ensured that it is not widely used. Below are some limitations for 100BaseT4 networks:

- Runs on unshielded four pair (UTP) cable.
- Connects using RJ-45 connectors.

The differences between the cables

When a cable is categorized as a cat 3 or cat 5, this refers to the transmission speed ratings of the cables (cat 5 being the fastest). Below are the main differences between the cables:

- Category 1 = No performance criteria
- Category 3 = Rated to 16 Mbps (used for 10BaseT, 100BaseT4)
- Category 4 = Rated to 20 Mbps (used for token-ring, 10BaseT)

- Category 5 = Rated to 100 Mbps (used for 100BaseTx, 10BaseT)

2.5 Ethernet frame types

There are two different Ethernet frame types: Ethernet II (also known as Standard Ethernet) and IEEE 802.3. They differ in the way that each frame identifies the upper layer protocol. Ethernet II uses a TYPE value for the identification and IEEE 802.3 uses a data LENGTH indicator.

Both Ethernet II and 802.3 can use the same physical component for communication. There are four transmission speeds and they are 10 Mbps, 100 Mbps, 1000 Mbps (Gigabit) and the new 10000 Mbps (10 Gigabit) standard.

The 10 Mbps standards

Below are some cable standards for 10 Mbps networks:

- 10Base2 runs over a thin 50 ohm baseband coaxial cable. It is also known as thin-Ethernet.
- 10Base5 runs over standard 50 ohm baseband coaxial cable.
- 10BaseF runs over fiber optic cable.
- 10BaseT runs over unshielded twisted-pair cable.

The 100 Mbps standards (also known as Fast Ethernet)

Below are some cable standards for 100 Mbps networks:

- 100BaseFx runs over a fiber optic cable.
- 100BaseT4 runs over a four-pair twisted-pair cable.
- 100BaseTx (also known as 10Base100) runs over a two-pair twisted-pair cable.

The 1000 Mbps (Gigabit) standard

Below are some cable standards for 1000 Mbps networks:

- 1000BaseT runs over unshielded twisted-pair cable.
- 1000BaseCX/LX/DX runs over a fiber optic cable.

The most commonly used frame type is Ethernet II, although some systems use the IEEE 802.3.

2.6 Hubs, bridges, switches, and routers

There are various ways to connect a network together as described below.

Hubs

A hub is a common connection point for devices in a network. Hubs are commonly used to connect segments of a LAN. A hub contains multiple ports. When a packet arrives at one port, it is copied to the other ports so that all segments of the LAN can see all packets.

A passive hub simply serves as a conduit for the data, enabling it to go from one device (or segment) to another. So-called intelligent hubs include additional features that enable an administrator to monitor the traffic passing through the hub and to configure each port in the hub. Intelligent hubs are also called manageable hubs.

A third type of hub, called a switching hub, actually reads the destination address of each packet and then forwards the packet to the correct port.

Bridges

A bridge is a device that connects two local area networks (LANs) or two segments of the same LAN. The two LANs being connected can be similar or dissimilar. For example, a bridge can connect an Ethernet with a token-ring network.

Unlike routers, bridges are protocol-independent. They simply forward packets without analyzing and re-routing messages. Consequently, they are faster than routers, but also less versatile.

Switches

A switch is a device that filters and forwards packets between LAN segments. Switches operate at the Data Link Layer (layer 2) of the OSI Reference Model and therefore support any packet protocol. LANs that use switches to join segments are called switched LANs or, in the case of Ethernet networks, switched Ethernet LANs.

Routers

A router is a device that connects any number of LANs.

Routers use headers and a forwarding table to determine where packets go, and they may communicate with each other in order to configure the best route between any two hosts.

Very little filtering of data is done through routers. Routers do not care about the type of data they handle.

Switched and non-switched 10BaseT systems

The following section discusses the main differences between non-switched 10BaseT networks using hubs and switched 10BaseT systems.

To understand why switches provide more functionality than hubs, a fundamental limitation of (non-switched) Ethernet should be understood. There can only be one device transmitting on a segment at any given time. If two or more devices attempt to transmit at the same time, a collision occurs. (An Ethernet segment where only one conversation can occur is called a collision domain.) After a collision, all devices must retransmit. As the number of devices on an Ethernet segment increases, the probability for collisions increase. Because devices must spend more time retransmitting data, the network is perceived to be slow.

Non-switched 10BaseT networks using hubs and repeaters

Before the advent of switches, a network could be divided into segments with a device called a bridge. Bridges have two Ethernet ports. As traffic flows through a network, a bridge learns which devices (identified by the MAC or hardware address) are on each side. The bridge then makes decisions to forward or not forward each packet to the other side based on where the destination device is located. A bridge thus divides a network into two collision domains, allowing two independent conversations to occur. If a bridge is placed intelligently, for example separating two departments and their respective file servers, they can improve network efficiency.

On non-switched networks, small mini-hubs may still be appropriate for offices where there are not enough jacks for every device.

Switched 10BaseT networks using switches

Hubs do no processing on network traffic; they simply repeat the incoming signal to all available ports. On a switch, every port acts as a bridge. If each switch port is connected to a single device, each device can, in principle, act independently of every other device.

For example, consider a switch with the following devices attached:

- Computer 1
- Computer 2
- Computer 3
- Printer
- File server
- Uplink to the Internet

In this case, computer 1 could be printing a document, while computer 2 connects to a file server, while computer 3 accesses the Internet. Because the switch intelligently forwards traffic only to the devices involved, there can be multiple independent simultaneous conversations.

2.7 Network protocols

All communications software uses protocols, sets of semantic and syntactic rules that determine the behavior of functional units in achieving communication. Protocols define how information is delivered, how it is enclosed to reach its destination safely, and what path it should follow. Protocols also coordinate the flow of messages and their acknowledgments.

Protocols exist at different levels within a UNIX kernel and cannot be directly manipulated. However, they are indirectly manipulated by what the user chooses to do at the application programming interface (API) level. 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.

There are various protocols available. With the Internet being so popular, the most common is TCP/IP, which is a combination of TCP and IP protocols.

To help understand the interaction between the different protocols and the layer on which they work, refer to Figure 2-4. This is the TCP/IP protocol suite, as this is the most common protocol being used.

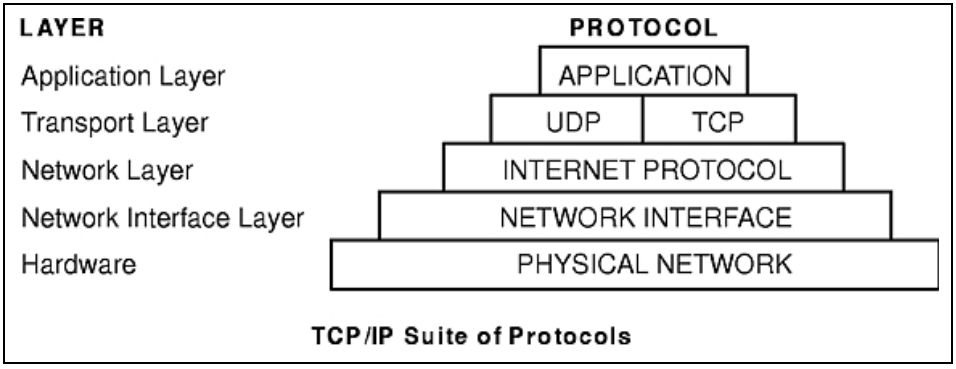


Figure 2-4 TCP/IP protocol suite

Address Resolution Protocol

Each network adapter has assigned a unique hardware address and the Hardware Layer uses them in order to define the destination of each network message within the same LAN. The ARP protocol is used to translate Internet

addresses into the hardware addresses on local area networks. Unlike most protocols, ARP packets do not have fixed-format headers. Instead, the message is designed to be used with a variety of network technologies. ARP is not used in point-to-point connections (for example Serial Line Internet Protocol (SLIP) or Serial Optical Channel Converter) since the destination of messages at the Hardware Layer is always the same.

The kernel maintains an IP address to hardware address translation table, and the ARP is not directly available to users or applications. When an application sends an Internet packet to one of the interface drivers, the driver requests the appropriate address mapping in order to define the destination from the Hardware Layer point of view. If the mapping is not in the table, an ARP broadcast packet is sent through the requesting interface driver to the hosts on the local area network. When any host that supports ARP receives an ARP request packet, it notes the IP and hardware addresses of the requesting system and updates its mapping table. If the receiving host does not match the requested IP address, it discards the request packet, otherwise it sends a response packet to the requesting system, containing its own hardware address. The requesting system learns in this way the new IP to hardware address mapping and stores it in the translation table.

Entries in the ARP mapping table are deleted after 20 minutes, while incomplete entries (ARP requests not answered) are deleted after three minutes. A permanent entry can be made in the ARP mapping tables using the **arp** command. The ARP cache works similar to a processor cache, using set associativity to determine cache replacement. Using the **no** command, it is possible to adjust the ARP table size if the number of systems on a subnet is very high.

Internet Control Message Protocol

The Internet Control Message Protocol (ICMP) is used to report communication errors or to test reachability from the source to the destination host. The **ping** command, for example, uses ICMP messages. ICMP uses the basic support of IP as though ICMP were a higher level protocol; however, ICMP is actually an integral part of IP and must be implemented by every IP module.

Internet Protocol

The Internet Protocol (IP) provides unreliable, connectionless packet delivery for the Internet. IP is connectionless because it treats each packet of information independently. It is unreliable because it does not guarantee delivery or have error recovery (that is, it does not require acknowledgments from the sending host, the receiving host, or intermediate hosts). It does provide basic flow control.

Simple Network Management Protocol

The Simple Network Management Protocol (SNMP) is a protocol for remotely performing administrative functions on a device.

Network Time Protocol

The Network Time Protocol (NTP) is available only in AIX Version 4.2 or later versions. It provides clock synchronization with time servers.

User Datagram Protocol

The User Datagram Protocol (UDP) is an unreliable user-level transport protocol for transaction-oriented applications. It handles datagram sockets and uses the IP for network services. It is up to the application that uses UDP to ensure transport reliability.

Transmission Control Protocol

TCP provides reliable stream delivery of data between Internet hosts. Like UDP, TCP uses the Internet Protocol, the underlying protocol, to transport datagrams, and supports the block transmission of a continuous stream of datagrams between process ports. Unlike UDP, TCP provides reliable message delivery. TCP ensures that data is not damaged, lost, duplicated, or delivered out of order to a receiving process. This assurance of transport reliability keeps applications programmers from having to build communications safeguards into their software.

Point-to-Point Protocol (PPP)

The Point-to-Point Protocol (PPP) is an open protocol for wide area network TCP/IP connectivity that can support both dial and leased lines. It can also be used to extend an enterprise intranet across multiple locations. PPP is a more robust alternative to Serial Line Internet Protocol (see Chapter 11, “Serial Line Internet Protocol” on page 247 for more information) when used as a dial-up protocol.

Point-to-point circuits in the form of asynchronous and synchronous lines have long been the mainstay for data communications.

PPP has three main components:

1. A method for encapsulating datagrams over serial links.
2. A Link Control Protocol (LCP) for establishing, configuring, and testing the data-link connection.
3. A family of Network Control Protocols (NCPs) for establishing and configuring different Network Layer protocols. PPP is designed to allow the simultaneous use of multiple Network Layer protocols.

PPP differentiates between client and server. This operating system can act as both a client and a server. The distinction is made to simplify configuration. PPP servers tend to allocate a pool of IP addresses among the connections that are being made. There is some correlation between the media devices. This implementation of PPP breaks this correlation. All server PPP connections are allocated on a first-available basis. This facilitates the separation of PPP from the media. The attachment process must request to be linked to the proper type of link. PPP links use a pool of IP addresses, so normal IP traffic can be confused with PPP IP addresses. It is recommended that PPP use a unique set of unused IP addresses and a machine with PPP active not have other services started.

2.7.1 Protocol summary

Table 2-1 summarizes which major protocols are used by which services and commands.

Table 2-1 Protocol summary

Protocol name	Description	Usage
Application Protocol	Provided by the program that uses TCP/IP for communication.	Used by TELNET, FTP, SNMP, and others.
TCP Protocol	Provides connection-oriented reliable data delivery, duplicate data suppression, congestion control, and flow control.	TCP function calls such as open, send, receive, and others.
UDP Protocol	Provides connectionless, unreliable, best-effort service.	UDP applications such as TFTP, DNS, NFS, RPC, and others.
Internet Protocol	Hides the underlying physical network by creating a virtual network view.	Includes IP addressing, IP subnet, IP routing, etc. Other internetwork layer protocols are IP, ICMP, IGMP, ARP and RARP.
Network Interfaces	Allow TCP/IP traffic to flow over various kinds of physical networks.	Includes Ethernet, token-ring, FDDI, SLIP, PPP, and others.

2.8 Networking hardware

The following sections discuss network adapters, drivers, and interfaces.

2.8.1 Network adapters

In AIX, TCP/IP networking is supported by several network adapter cards and connections, including:

- Ethernet adapters (10/100 MBps) (either built-in or adapter cards)
- Gigabit Ethernet
- Token-ring
- Fiber Distributed Data Interface (FDDI)
- Asynchronous Transfer Mode (ATM) Turboways 100/155
- Asynchronous adapters and native serial ports
- Serial Optical Channel Converter

Adding a network adapter

When an adapter is added to the system, a logical device is created in the ODM, for example Ethernet adapters, as follows:

```
# lsdev -Cc adapter | grep ent
ent0    Available 10-80    IBM PCI Ethernet Adapter (22100020)
ent1    Available 20-60    Gigabit Ethernet-SX PCI Adapter (14100401)
```

A corresponding network interface will allow TCP/IP to use the adapter. For auto-detectable adapters, such as Ethernet and token-ring, the network interface is automatically created. For other types (for example, ATM), an interface might need to be manually created.

To configure the new network interface, use the SMIT command **`smit mkinet`**.

To load additional drivers, if required, use the **`smit installp`** command.

AIX location codes

In the following, the AIX location codes are described for the purpose of identifying the location of network adapters on your system. The AIX location code is a way of identifying physical devices. The actual location code values vary among the different server architecture types such as MCA, PCI RSPC, and PCI CHRP, but the same format is used.

The location code consists of up to four fields of information depending on the type of device. The basic formats of the AIX location codes are:

AB-CD-EF-GH For planars, adapters and any non-SCSI devices

AB-CD-EF-G,H For SCSI devices/drives

For planars, adapter cards, and non-SCSI devices, the location code is defined as:

AB The AB value identifies a bus type or PCI parent bus as assigned by the firmware.

CD The CD value identifies adapter number, adapter's devfunc number or physical location. The devfunc number is defined as the PCI device number times 8 plus the function number.

EF The EF value identifies the connector ID used to identify the adapter's connector that a resource is attached to.

GH Identifies a port, address, device, or field replaceable unit (FRU).

Adapters such as network adapters and network cards are identified with just AB-CD.

The possible values for AB are:

00 Processor bus

01 ISA bus

02 EISA bus

03 MCA bus

04 PCI bus (used in the case where the PCI bus cannot be identified)

05 PCMCIA buses

xy For PCI adapters where x is equal to or greater than 1. The x and y are characters in the range of 0-9, A-H, J-N, P-Z (O, I, and lowercase are omitted) and are equal to the parent bus's ibm, aix-loc Open Firmware Property.

The possible values for CD depend on the adapter/card:

PCI adapters/cards CD is the device's devfunc number. The C and D are characters in the range of hexadecimal numbers 0-F.

Pluggable ISA adapters CD is equal to the order the ISA cards are defined/configured either by SMIT or the ISA Adapter Configuration Service Aid.

Integrated ISA adapters CD is equal to a unique code identifying the ISA adapter. In most cases this is equal to the adapter's physical location code. In cases where a physical location code is not available, CD will be FF.

To illustrate the usage of AIX location codes used for a network adapter, Table 2-2 on page 27 lists those for a RS/6000 7025 Model F50.

Table 2-2 RS/6000 7025 F50 AIX Location Codes

Location Code	Description
10-80	Ethernet Port.
20-58 to 20-5F	Any PCI card in slot 1. PCI 64-bit bus
20-60 to 20-67	Any PCI card in slot 2. PCI 64-bit bus
10-68 to 10-6F	Any PCI card in slot 3. PCI 32-bit bus
10-70 to 10-77	Any PCI card in slot 4. PCI 32-bit bus
10-78 to 10-7F	Any PCI card in slot 5. PCI 32-bit bus
30-60 to 30-67	Any PCI card in slot 6. PCI 32-bit bus
30-68 to 30-6F	Any PCI card in slot 7. PCI 32-bit bus
30-70 to 30-77	Any PCI/ISA card in slot 8.
30-78 to 30-7F	Any PCI/ISA card in slot 9

To identify the adapter location, list the adapters on the system using the **lsdev** command, as follows:

```
# lsdev -Cc adapter
ppa0    Available 01-R1    Standard I/O Parallel Port Adapter
sa0     Available 01-S1    Standard I/O Serial Port
sa1     Available 01-S2    Standard I/O Serial Port
sa2     Available 01-S3    Standard I/O Serial Port
siokma0 Available 01-K1    Keyboard/Mouse Adapter
fda0    Available 01-D1    Standard I/O Diskette Adapter
scsi0   Available 10-60    Wide SCSI I/O Controller
tok0    Available 10-68    IBM PCI Tokenring Adapter (14103e00)
ent0    Available 10-80    IBM PCI Ethernet Adapter (22100020)
mg20    Available 20-58    GXT130P Graphics Adapter
ent1    Available 20-60    Gigabit Ethernet-SX PCI Adapter (14100401)
scsi1   Available 30-58    Wide SCSI I/O Controller
sioka0  Available 01-K1-00 Keyboard Adapter
sioma0  Available 01-K1-01 Mouse Adapter
```

The network adapters on this system are tok0 (a PCI token-ring adapter card with location code 10-68), ent0 (a built-in Ethernet adapter with location code 10-80), and a PCI Gigabit Ethernet adapter card with location code 20-60. Using the location table, it is possible to see that the Gigabit Ethernet adapter card is located in the 64-bit PCI slot 2. The token-ring adapter card is located in 32-bit PCI slot 3.

Table 2-2 lists the location codes of a pSeries 640 Model B80.

Table 2-3 pSeries 640 Model B80 AIX location codes

Location code	Description
10-60	Ethernet port 1
10-80	Ethernet port 2
20-58 to 20-5F	Card in PCI Slot 1P
20-60 to 20-67	Card in PCI Slot 2P
10-68 to 10-6F	Card in PCI Slot 3P
10-70 to 10-77	Card in PCI Slot 4P
10-78 to 107F	Card in PCI Slot 5P
01-R1	Parallel Port
01-S2	Serial Port 2
01-S3	Serial Port 3
10-88	Internal SCSI
10-89	External SCSI

Note: Recommendations on the placement of adapter cards for the different server models can be found in *PCI Adapter Placement Reference*, SA38-0538.

The location code table is not valid for all RS/6000 PCI CHRP models. For a precise description of the AIX location code for a specific RS/6000 or pSeries model, refer to the user's guide of that system. You can also use the following URL:

http://www.ibm.com/servers/eserver/pseries/library/hardware_docs/index.html

The **lscfg** command displays configuration, diagnostic, location and vital product data (VPD) information about the system. Below is an example of the **lscfg** command:

```
# lscfg
INSTALLED RESOURCE LIST

The following resources are installed on the machine.
+/- = Added or deleted from Resource List.
*   = Diagnostic support not available.
```

```
  Model Architecture: chrp
  Model Implementation: Multiple Processor, PCI bus

+ sys0                          System Object
+ sysplanar0                    System Planar
* pci1             P1           PCI Bus
* pci6             P1           PCI Bus
+ ent0             P1/E1        IBM 10/100 Mbps Ethernet PCI Adapter (23100020)
* pci7             P1           PCI Bus
+ scsi2            P1-I8/Z1     Wide/Ultra-2 SCSI I/O Controller
+ hdisk2           P1-I8/Z1-A8  16 Bit LVD SCSI Disk Drive (9100 MB)
+ hdisk3           P1-I8/Z1-A9  16 Bit LVD SCSI Disk Drive (9100 MB)
+ ses0             P1-I8/Z1-Af  SCSI Enclosure Services Device
.....
.....
.....
```

Removing a network adapter

To remove a network adapter you first have to remove the network interfaces and remove the adapter device afterwards.

For an ent1 Ethernet adapter, perform the following steps (remember that both ent1 and et1 exists):

1. List the adapter:.

   ```
   # lsdev -Cl ent1
   ent1 Available 04-D0 IBM PCI Ethernet Adapter (22100020)
   ```

2. List the network interface definition:

   ```
   # lsdev -Cl en1
   en1 Available Standard Ethernet Network Interface
   ```

3. Bring the interface down:

   ```
   # ifconfig en1 down
   ```

4. Delete the network interface definition for the adapter:

   ```
   # ifconfig en1 detach
   ```

5. Delete the network interface driver for the adapter:

   ```
   # rmdev -l en1 -d
   en1 deleted
   # rmdev -l ent1 -d
   ent1 deleted
   ```

After this, you can shut down, power off the system, and physically remove the adapter, or, if you are using a PCI hot-swap slot, deactivate the PCI slot and remove the adapter while the system is running.

2.8.2 Network drivers

To verify which driver for your adapter is installed in your system, verify your network adapter type using the **lsdev** command and check the device ID of the adapter, which is the number in brackets after the adapter description. Search for the corresponding LPP using the **lslpp** command. The following example shows how to retrieve driver information for a Gigabit Ethernet Adapter:

```
# lsdev -Cc adapter | grep ent
ent0    Available 10-80    IBM PCI Ethernet Adapter (22100020)
ent1    Available 20-60    Gigabit Ethernet-SX PCI Adapter (14100401)

# lslpp -l | grep 14100401
  devices.pci.14100401.diag  4.3.3.0  COMMITTED  Gigabit Ethernet-SX PCI
  devices.pci.14100401.rte  4.3.3.10  COMMITTED  Gigabit Ethernet-SX PCI
  devices.pci.14100401.rte   4.3.3.0  COMMITTED  Gigabit Ethernet-SX PCI
```

You can use the **lppchk** command to verify that files for an installable software product (fileset) match the Software Vital Product Data (SWVPD) database information for file sizes, checksum values, or symbolic links. For example, to verify that all filesets have all required prerequisites and are completely installed, enter:

```
# lppchk -v
```

Missing driver

If the new hardware is not listed when using the **lsdev** command (for example, **lsdev -Cc adapter**), you can determine the missing software by running **cfgmgr** from a command window. The **cfgmgr** command will display a warning and indicate the missing driver filesets:

```
# cfgmgr
cfgmgr: 0514-621 WARNING: The following device packages are required for
        device support but are not currently installed.
devices.pci.token-ring:devices.pci.14101800:devices.pci.IBM.42H0658:devices.pci
.
IBM.25H3037:devices.pci.IBM.38H5818
```

Install the missing driver software and re-run **cfgmgr** or insert the first AIX CD and run **cfgmgr -i /dev/cd0**. If **cfgmgr** does not display a warning message, the adapter device was created using the correct driver.

Network driver attributes

To see the actual driver setting or list of attributes of a network driver, use the **lsattr** command. This will list all the available driver attributes names with their current values and a description of the purpose of the attribute. Each driver attribute has a flag indicating if the attribute is changeable or not.

```
# lsattr  -E -l ent1
busmem        0x3cfec000     Bus memory address                       False
busintr       7              Bus interrupt level                      False
intr_priority 3              Interrupt priority                       False
rx_que_size   512            Receive queue size                       False
tx_que_size   512            Software transmit queue size             True
jumbo_frames  no             Transmit jumbo frames                    True
use_alt_addr  no             Enable alternate ethernet address        True
alt_addr      0x000000000000 Alternate ethernet address               True
trace_flag    0              Adapter firmware debug trace flag        True
copy_bytes    256            Copy packet if this many or less bytes   True
tx_done_ticks 1000000        Clock ticks before TX done interrupt     True
tx_done_count 64             TX buffers used before TX done interrupt True
receive_ticks 50             Clock ticks before RX interrupt          True
receive_bds   6              RX packets before RX interrupt           True
receive_proc  16             RX buffers before adapter updated        True
stat_ticks    1000000        Clock ticks before statistics updated    True
rx_checksum   yes            Enable hardware receive checksum         True
```

This example lists the attributes of a Gigabit Ethernet Driver. Notice that the attributes **busmem**, **businter**, **intr_priority**, and **rx_que_size** are not changeable. The values for this PCI network card are set automatically by the system.

If the attribute flag is set to True, then the value can be changed by the **chdev** command, as follows:

```
# chdev -l ent1 -a rx_checksum=yes
ent1 changed
```

Before changing any network driver attribute, refer to the publications for the specific device driver. For best performance, interface settings must match the network settings.

The **lsattr** command can assist in setting the correct value for the network driver attributes. The -R flag provides information about the value range for a specific driver attribute:

```
# lsattr -R -l ent1 -a stat_ticks
1000...1000000 (+1)
```

This example shows that the attribute **stat_tick** (clock ticks before statistics updated) can be set from 1000 to 1000000 using integer numbers.

2.9 AIX network interfaces

The interfaces listed in Table 2-4 on page 32 are supported by AIX Version 4.3. There may be multiple devices of the same type in the system and each device

will have an interface. The x after the adapter and interface names indicates the number of the adapter or interface respectively, starting from 0. The number increases for each adapter added to the system.

Table 2-4 AIX Version 4.3 supported interfaces

Adapter	Interface	Description
-	lo0	Loopback
LAN		
ent*x*	en*x*	Standard Ethernet (all speeds 10/100/Gigabit)
ent*x*	et*x*	IEEE 802.3
tok*x*	tr*x*	Token-ring
atm*x*	at*x*	Asynchronous Transfer Mode (ATM)
fddi*x*	fi*x*	Fiber Distributed Data Interface (FDDI)
WAN		
sa*x*	sl*x*	Serial Line Internet Protocol (SLIP)
sa*x*	pp*x*	Point-to-Point Protocol (PPP)
sx25a*x*	xs*x*	X.25
SP or **Mainframe**		
css*x*	css*x*	SP Switch
ops*x*	so*x*	Serial Optical Channel Converter
cat*x*	ca*x*	370 Parallel Channel Network Interface

Note that for Ethernet adapters, the standard Ethernet (en), and 802.3 (et) network technologies use the same type of adapter.

The **lsdev** command can be used to list the available network interfaces on your system:

```
# lsdev -Cc if
en0 Available  Standard Ethernet Network Interface
et0 Defined    IEEE 802.3 Ethernet Network Interface
lo0 Available  Loopback Network Interface
tr0 Available  Token Ring Network Interface
```

Similar to the network adapter, the network interface attributes can be changed using a combination of the **lsattr** and **chdev** command.

For example:

```
# lsattr -E -l en0 -a netaddr
netaddr 10.47.1.5 Internet Address True
```

The following example shows how **chdev** can be used to change the IP address of the system.

```
# chdev -l en0 -a netaddr=10.47.1.6
en0 changed
# ifconfig en0
en0:
flags=e080863<UP,BROADCAST,NOTRAILERS,RUNNING,SIMPLEX,MULTICAST,GROUPRT,64BIT>
        inet 10.47.1.6 netmask 0xffff0000 broadcast 10.47.255.255
```

Note: Be aware of the following considerations:

- The Ethernet adapter can be used for either Ethernet or IEEE 802.3.
- There can be multiple adapters of the same type in the system and each will have its own interface.

2.10 Quiz

The following assessment questions help verify your understanding of the topics discussed in this chapter.

1. Which one of the following adapters may directly connect with fiber optic cables?

   A. Arcnet

   B. FDDI

   C. 16 MB token-ring

   D. 10 MB Ethernet

2. In some networks, an adapter may be required to use special end of the line termination resistors. Which one of the following indicates this type of network?

   A. Coaxial

   B. Fiber optic

   C. Shielded pair

   D. Unshielded twisted pair

3. Which one of the following cable types is not affected by electromagnetic interference?

   A. Coaxial

   B. Fiber optic

   C. Cat 3 UTP

   D. Cat 5 UTP

4. Which one of the following cable types is associated with an Ethernet card with a BNC connector?

   A. 10BaseT

   B. 10Base2

   C. 10Base5

   D. 10Base100

5. The IP address of the only FDDI adapter must be changed in a system. Which one of the following network interfaces should be modified?

   A. fda

   B. fd0

   C. fi0

   D. fddi0

6. In order to keep a host's clock synchronized with other host clocks, which one of the following services should be used?

   A. NIS

   B. NTP

   C. DHCP

   D. INETD

7. Which one of the following cable types indicates that an adapter setting of “BNC” is required?

   A. Coaxial

   B. Wireless

   C. Fiber optic

   D. Twisted pair

8. Which one of the following adapter information is provided by the **lsdev -C** command?

   A. Availability

   B. Firmware levels

   C. Hardware address

   D. Transmit queue size

9. Which one of the following commands will indicate if the device driver is installed?

   A. **ls**

   B. **lslpp**

   C. **netstat**

   D. **ifconfig**

10. Which one of the following commands should be used to correctly install a device driver?

    A. **rcp**

    B. **mkdev**

    C. **smitty inet**

    D. **smitty installp**

11. Which one of the following commands reveals the current setting for an adapter’s cable type?

    A. **route**

    B. **lsdev**

    C. **lslpp**

    D. **lsattr**

12. An Ethernet switch is set for half duplex on the port and leads to an adapter on an AIX machine. Which one of the following settings should result in maximum adapter performance?

    A. Half-duplex

    B. Full-duplex

    C. Autosense

    D. Autonegotiate

13. Which one of the following commands can temporarily shut down a network interface?

    A. **`route`**

    B. **`cfgmgr`**

    C. **`netstat`**

    D. **`ifconfig`**

14. On a large flat Class B network there are over 65,500 machines on the same unrouted wire. Which one of the following procedures must be performed to ensure adequate connectivity is maintained?

    A. Use multiple adapters

    B. Increase the default ARP table size

    C. Alias multiple IP addresses onto the adapters

    D. Enlarge the size of the default routing table size

15. To configure a switched virtual circuit classical IP interface on an ATM adapter, which one of the following ATM server addresses should be supplied?

    A. “ARP”

    B. “DNS”

    C. “LES”

    D. “LECS”

16. Scenario: A network administrator has been asked to integrate a new RS/6000 to be used as a corporate mail server into the network. There are five nodes on the Ethernet II network, with a network address of 193.3.7.0 and a subnet mask of 255.255.255.0. The machine contains ATM, token-ring and integrated Ethernet adapters.

    Which one of the following devices should be configured for this?

    A. at0

    B. tr0

    C. et0

    D. en0

17. An AIX box can ping successfully through a hub. However, when the same cable is disconnected from the hub and connected to a switch, the ping fails. In which one of the following is the problem most likely to be occurring?

    A. The cable

    B. The switch

    C. The AIX box

    D. The target of the ping

18. If AIX is configured with the same Internet address for the Ethernet network and PPP setup, which one of the following options is *true*?

    A. The **pppattachd** command will not need to be run.

    B. One of the network interfaces will be unusable for outbound traffic.

    C. A separate route will not need to be created for the PPP connection.

    D. It will automatically create a new IP address at the start of the PPP connection.

19. Given a subnet mask of 255.255.255.0, which one of the following addresses should be used to broadcast a packet to all subnets?

    A. 129.35.35.255

    B. 129.35.255.255

    C. 129.255.255.255

    D. 255.255.255.255

20. In a network with subnets, a packet with destination IP address 255.255.255.255 is:

    A. An invalid packet.

    B. A broadcast to a subnet.

    C. A broadcast to all hosts connected to the Internet.

    D. A broadcast to every host directly attached to the local network.

21. Which one of the following commands indicates physical slot locations for every adapter installed?

    A. **`lscfg`**

    B. **`lslpp`**

    C. **`lsattr`**

    D. **`ifconfig`**

22. Which one of the following commands can be used to check if the ent0 adapter is available?

    A. **`ifconfig en0`**

    B. **`lscfg -vl ent0`**

    C. **`lsattr -El ent0`**

    D. **`lsdev -Cl ent0`**

2.10.1 Answers

The following are the preferred answers to the questions provided in this section:

1. B
2. A
3. B
4. B
5. C
6. B
7. A
8. A
9. B
10. D
11. D
12. A
13. D
14. B
15. A
16. D
17. B
18. B
19. B
20. D
21. A
22. D

2.11 Exercises

The following exercises provide sample topics for self study. They will help ensure comprehension of this chapter.

1. What interfaces are used by AIX for the different protocols?
2. Explain the differences between the cables and in which type of system they will most likely be used.

3

Network addressing and routing

The following topics are discussed in this chapter:

- The IP addressing overview
- Routing concepts
- Setting up the router

This chapter contains an introduction to TCP/IP and discusses the network addressing and routing protocols.

3.1 Internet addressing

If you want your machines to communicate with each other across the TCP/IP network, you must give them unique IP addresses. Each host is assigned a unique 32-bit logical address (in the case of IPv4) that is divided into two main parts: the network number and the host number. The network number identifies a logical network to which the host belongs and must be the same across the subnet. The host number identifies a host on the specific logical network.

3.1.1 IP address format

The IP address is the 32-bit address, grouped eight bits at a time, separated by dots and represented in decimal format - *dotted decimal notation*. Each bit in the octet has a binary weight (128, 64, 32,16, 8, 4, 2, 1). The minimum value for an octet is 0, and the maximum value for an octet is 255. Figure 3-1 illustrates the basic format of an IP address.

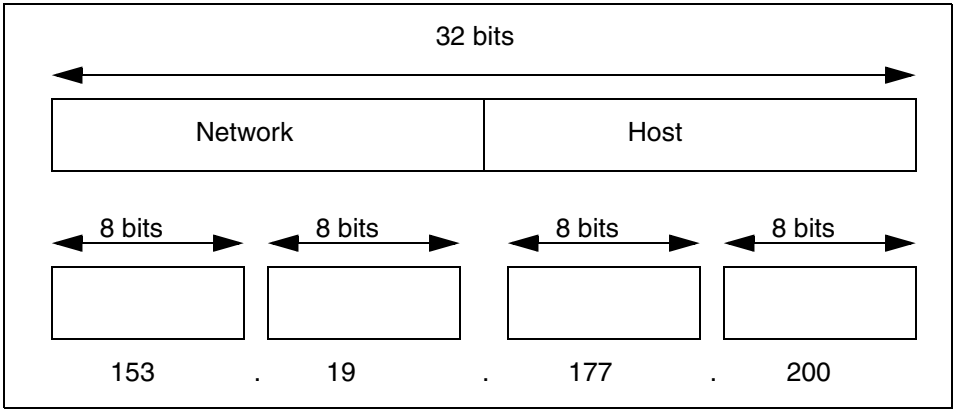


Figure 3-1 IP address format

Binary to decimal conversion review

The decimal value of the bits ranges from high to low with the leftmost bit in every byte having the highest value of 128. To convert from binary value to decimal value, add decimal values on the position where the bits have a value of 1. An example is shown in Figure 3-2 on page 43.

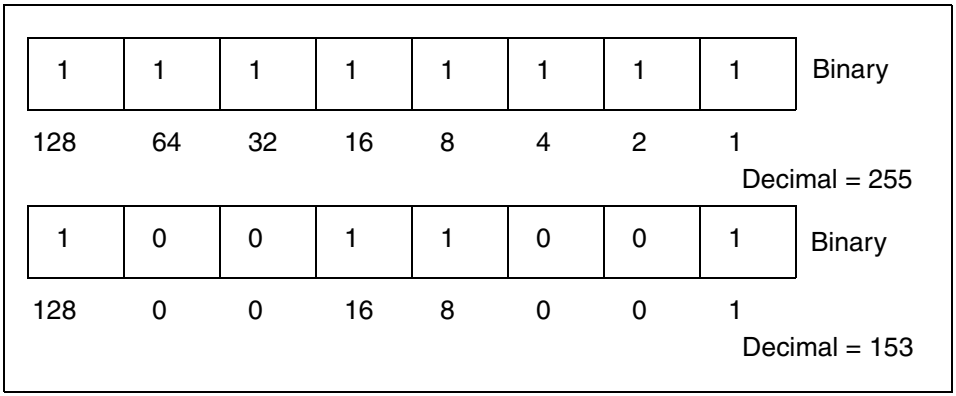


Figure 3-2 Binary to decimal review

If you are not sure, you can use the **bc** command. To make the conversion of value 195 to binary format, enter:

```
# bc
obase=2
195
11000011
```

To convert binary value `11001100` to decimal value, enter:

```
# bc
ibase=2
11001100
204
```

3.1.2 Internet address classes

IP addressing supports five different address classes: A, B, C, D and E. Classes A, B and C are available for commercial networking use. You can recognize the network class by first checking bits in the first octet of an address' network part.

After converting all of those bits to binary format and recalculating numbers of hosts and networks, you receive data as shown in Figure 3-3 on page 44.

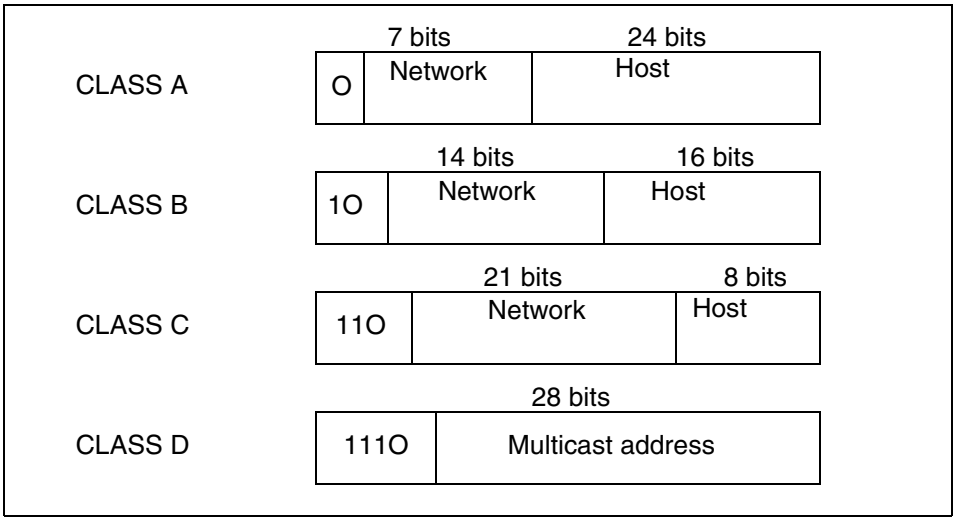


Figure 3-3 IP address classes

To determine an IP address's class use Table 3-1. For example, in the IP address 195.116.119.2, the first octet is 195. Because 195 falls between 192 and 223, 195.116.119.2 is a class C address.

Table 3-1 IP address classes

IP address class	Format	First octet	Address range	Number bits network / host	Number of hosts
A	N.H.H.H	0	1.0.0.0 127.0.0.0	7 / 24	2^{24} - 2
B	N.N.H.H	10	128.1.0.0 191.254.0.0	14 / 16	2^{16} - 2
C	N.N.N.H	110	192.0.1.0 223.255.254.0	22 / 8	2^{8} - 2
D	-	1110	224.0.0.0 239.255.255.255	-	-
N - Network number H - Host number					

Class A, B, and C provide address ranges that are useful to define a private network without INTERNIC authorization. A private network can have the following address ranges:

Class A	10.0.0.0 to 10.255.255.255
Class B	172.16.0.0 to 172.31.255.255
Class C	192.168.0.0 to 192.168.255.255

Internet Assigned Numbers Authority (IANA) can be contacted to get a public address. If a domain name is required, INTERNIC provides a list of authorized organizations who can allocate domain names. Visit the following IANA and INTERNIC sites for further details:

http://www.iana.org
http://www.internic.net

3.1.3 Special Internet addresses

There are a few IP addresses that cannot be used as a host address. Those addresses are used for special occasions.

The loopback address

The Internet Protocol defines the special network address, 127.0.0.1, as a local *loopback* address. Hosts use local *loopback* addresses to send messages to themselves.The loopback interface allows a client and server on the same host to communicate with each other using TCP/IP. The network class A with network address 127 is reserved for the loopback interface lo0. AIX assigns the IP address 127.0.0.1 to this interface and assigns it the name *localhost.* To check attributes of any interface use the **ifconfig** or **lsattr** command.

```
# ifconfig lo0
lo0:
flags=e08084b<UP,BROADCAST,LOOPBACK,RUNNING,SIMPLEX,MULTICAST,GROUPRT,64BIT>
        inet 127.0.0.1 netmask 0xff000000 broadcast 127.255.255.255
        inet6 ::1/0
# lsattr -El lo0
netaddr   127.0.0.1 Internet Address                      True
state     up        Current Interface Status              True
netmask             Subnet Mask                           True
mtu       16896     Maximum IP Packet Size for This Device True
netaddr6  ::1       N/A                                   True
prefixlen           Subnet Mask                           True
```

The network address

The *network address* is an IP address with all host address bits set to 0. If you have IP address 195.116.119.2, the network address for this will be 195.116.119.0. This type of address is used in the routing table as the network destination address. An example routing table as follows(0 is omitted in the routing tables):

```
# netstat -nr
Routing tables
Destination      Gateway           Flags   Refs     Use  If   PMTU  Exp  Groups

Route Tree for Protocol Family 2 (Internet):
default          9.3.240.1         UGc       0        0  tr0     -   -
9.3.240/24       9.3.240.58        U        30   130787  tr0     -   -
127/8            127.0.0.1         U        54     1300  lo0     -   -
195.116.119/24   195.116.119.2     U         0        2  en0     -   -
```

The broadcast address

TCP/IP can send data to all hosts on a local network or to all hosts on all directly connected networks. Such transmissions are called *broadcast messages*. For example, the routed routing daemon uses broadcast messages to query and respond to routing queries. Broadcast addresses are never valid as a source address. They must specify the destination address. The different types of broadcast addresses include:

- Limited broadcast address

  This uses the address 255.255.255.255 (all bits 1 in all parts of the IP address). It refers to all hosts on the local subnet. This is recognized by every host. The hosts do not need any IP configuration information. Routers do not forward this packet.

- Network-directed broadcast address

  This is used in an unsubnetted environment. The network number is a valid network number and the host number is all ones (for example, 128.2.255.255). This address refers to all hosts on the specified network. Routers should forward these broadcast messages. This is used in ARP requests on unsubnetted networks.

- Subnet-directed broadcast address

  If the network number is a valid network number, the subnet number is a valid subnet number and the host number is all ones, then the address refers to all hosts on the specified subnet. Since the sender's subnet and the target subnet may have different subnet masks, the sender must somehow find out the subnet mask in use at the target. The actual broadcast is performed by the router that receives the datagram into the subnet.

- All-subnets-directed broadcast address

  If the network number is a valid network number, the network is subnetted and the local part is all ones (for example, 128.2.255.255), then the address refers to all hosts on all subnets in the specified network. In principle, routers may propagate broadcasts for all subnets, but are not required to do so. In practice, they do not. There are very few circumstances where such a broadcast is desirable. If misconfigured, it can lead to problems. Consider the

misconfigured host 9.180.214.114 in a subnetted Class A network. If the device was configured with the address 9.255.255.255 as a local broadcast address instead of 9.180.214.255, all of the routers in the network will forward the request to all clients.

If routers do respect all-subnets-directed broadcast address, they use an algorithm called *reverse path forwarding* to prevent the broadcast messages from multiplying out of control.

For example, to check the broadcast setting for interface en0, enter:

```
# ifconfig en0
en0:
flags=4e080863<UP,BROADCAST,NOTRAILERS,RUNNING,SIMPLEX,MULTICAST,GROUPRT,64
BIT,PSEG>
        inet 9.3.4.100 netmask 0xfffffe00 broadcast 9.3.5.255
```

The multicast address

The use of *Internet Protocol (IP) multicasting* enables a message to be transmitted to a group of hosts, instead of having to address and send the message to each group member individually. Internet addressing provides for Class D addressing that is used for multicasting. IP multicast is a routing technique that allows IP traffic to be sent from one source or multiple sources and delivered to multiple destinations. Instead of sending individual packets to each destination, a single packet is sent to a multicast group, which is identified by a single IP destination group address. The intent of multicasting is to reduce the load on hosts not required to receive the message. IP multicasting is used with Internet Chat, Internet Talk Radio, Internet Phone, and Video conferencing.

Every network traffic IP multicast also needs to be routed between networks. AIX uses the mrouted daemon that multicasts traffic between multicast-capable subnetworks. The /etc/mrouted.conf configuration file contains entries that provide configuration information used by the mrouted daemon.

The last column of Table 3-1 on page 44 shows the number of hosts in the appropriate network class. The reason for subtracting two hosts is that one address is reserved for the broadcast address, and one address is reserved for the network address.

3.1.4 Subnetting

Subnet addressing allows an autonomous system made up of multiple networks to share the same Internet address class. The subnetwork capability of TCP/IP also makes it possible to divide a single network into multiple logical networks (subnets). This makes sense for class A and class B addresses, since attaching thousands of hosts to a single network is impossible.

A standard IP address has two fields (see 3.1.1, “IP address format” on page 42): a network address and a host address. A subnet address is created by *borrowing* bits from the host field and designating them as the subnet field. The number of borrowed subnet bits varies and it depends of the chosen subnet mask. Figure 3-4 shows how bits are borrowed from the host address field to create the subnet address field and how the subnet mask works.

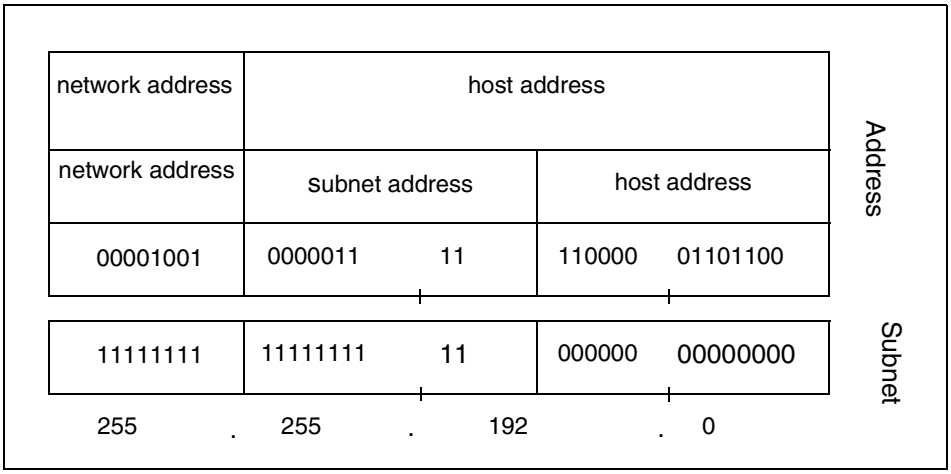


Figure 3-4 Subnetting example

When deciding how to partition the host address into the subnet address and host address, you should consider the number of subnets and the number of hosts on those subnets.

You have great flexibility when assigning subnet addresses and host addresses. The bits of the host address can be divided according to the needs and potential growth of the organization and its network structure. The only restrictions are:

- Network address is constant for all its subnets.
- Subnet address is constant throughout the physical network.
- Host address is a field that is normally at least 2 bits wide.

If the width of the subnet address field is 0, the network is not organized into subnets, and addressing to the network is performed using the Internet network address as mentioned in 3.1.1, “IP address format” on page 42.

Note: It is generally desirable for the subnet bits to be contiguous and located as the most significant bits of the host address.

Subnet mask

The subnet mask tells the system what the subnet partitioning scheme is. This bit mask consists of the network address portion and subnet address portion of the IP address.

The host number part of the IP address is subdivided into a second network number and a host number. This second network is termed a subnetwork or subnet. The main network now consists of a number of subnets. The IP address is interpreted as:

<network number><subnet number><host number>

The combination of subnet number and host number is often termed the local address or the local portion of the IP address. Subnetting is implemented in a way that is transparent to remote networks. A host within a network that has subnets is aware of the subnetting structure. A host in a different network is not. This remote host still regards the local part of the IP address as a host number.

When a host sends a message to a destination, the system must determine whether the destination is on the same network as the source or if the destination can be reached through a gateway. The system compares the destination address to the host address using the subnet mask. If the destination is not on the local network, the system sends the packet to a gateway. The gateway performs the same comparison to see if the destination address is on a network it can reach locally.

Table 3-2 shows how to calculate the subnet mask from binary format to the dotted decimal notation.

Table 3-2 Subnet mask calculation

Bits of octet								**Mask**
128	**64**	**32**	**16**	**8**	**4**	**2**	**1**	
1	0	0	0	0	0	0	0	128
1	1	0	0	0	0	0	0	192
1	1	1	0	0	0	0	0	224
1	1	1	1	0	0	0	0	240
1	1	1	1	1	0	0	0	248
1	1	1	1	1	1	0	0	252
1	1	1	1	1	1	1	0	254
1	1	1	1	1	1	1	1	255

A subnet mask is 32 bits long. A bit set to 1 in the subnet mask indicates that bit position is part of the network address portion of the IP address. A bit set to 0 in the subnet mask indicates that bit position is part of the host address portion of the IP address.

There are default subnet mask sets (Figure 3-5) for each network class address. Using an address with a default subnet mask for an address class indicates that subnets are not set up for the network.

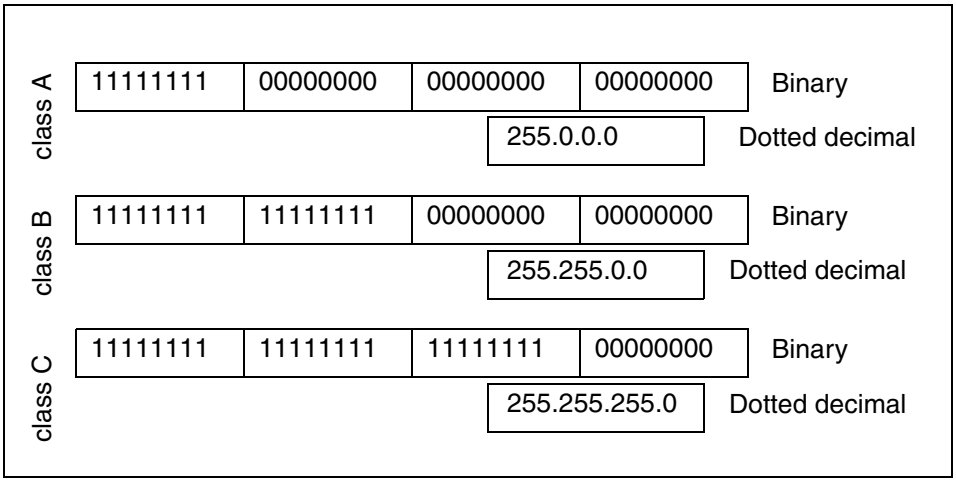


Figure 3-5 Default subnet mask for network classes

The class A address subnetting example

Take, for example, a subnet mask of 255.255.255.192 (or 11111111 11111111 11111111 11000000 in bit notation). Note that, by convention, the <network address> is included in the mask as well.

Because of the all bits 0 and all bits 1 restrictions, this defines 2^{18}-2 (from 1 to 262143) valid subnets. This split provides 262142 subnets, each with a maximum of 2^{6}-2 (62) hosts.

The value applied to the subnet number takes the value of the full octet with non-significant bits set to zero. For example, the hexadecimal value 01 in this subnet mask assumes an 8-bit value 01000000. This provides a subnet value of 64.

Applying the 255.255.255.192 to the sample class A address 9.67.38.1 provides the following information:

```
00001001 01000011 00100110 00000001 = 9.67.38.1 (class A address)
```

```
 11111111 11111111 11111111 11------   255.255.255.192 (subnet mask)
====================================  logical AND
 00001001 01000011 00100110 00------ = 9.67.38.0(subnet base address)
```

This leaves a host address of:

```
-------- -------- -------- --000001 = 1        (host address)
```

IP will recognize all host addresses as being on the local network for which the logical AND operation described above produces the same result. This is important for routing IP datagrams in subnet environments.

The actual subnet number is:

```
-------- 01000011 00100110 00------ = 68760    (subnet number)
```

This subnet number is a relative number. That is, it is the 68760th subnet of network 9 with the given subnet mask. This number bears no resemblance to the actual IP address that this host has been assigned (9.67.38.1). It has no meaning in terms of IP routing.

The division of the original <host address> into <subnet><host> is chosen by the network administrator. The values of all zeroes and all ones in the <subnet> field are reserved.

The class B address subnetting example

The default subnet mask for a class B address that has no subnetting is 255.255.0.0, while the subnet mask for a class B address 172.16.0.0 that specifies 3 bits of subnetting is 255.255.224.0. The reason for this is that 3 bits of subnetting give 2^3- 2= 6 (1 for the network address and 1 for the broadcast address) subnets possible. You have 5 bits from the 3rd octet and 8 bits from the last octet forming a total of 13 bits for the host's address. This gives you 2^{13}- 2 = 8190 hosts per subnet. Figure 3-6 on page 52 shows a subnetting scenario for this address.

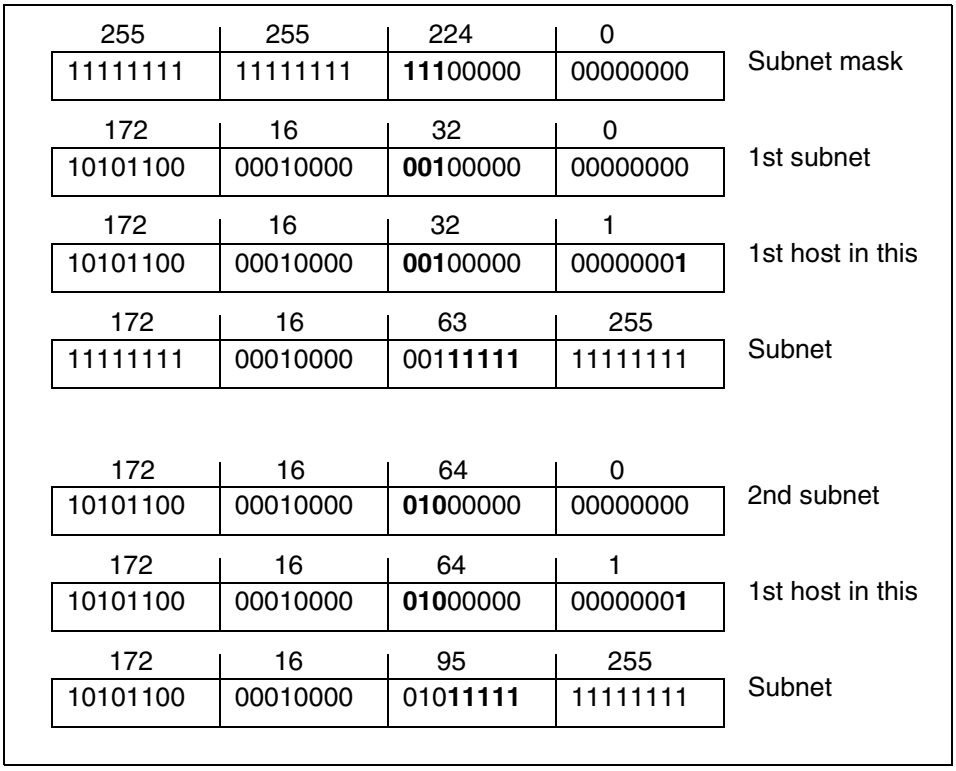


Figure 3-6 Subnetting scenario

Table 3-3 shows the subnet mask, the number of subnets, and the number of hosts depending on the numbers of bits for subnet for network class B.

Table 3-3 Class B subnetting reference chart

Numbers of bits for subnet	Subnet mask	Number of subnets	Number of hosts
2	255.255.192.0	2	16382
3	255.255.224.0	6	8190
4	255.255.240.0	14	4094
5	255.255.248.0	30	2046
6	255.255.252.0	62	1022
7	255.255.254.0	126	510
8	255.255.255.0	254	254

Numbers of bits for subnet	Subnet mask	Number of subnets	Number of hosts
9	255.255.255.128	510	126
10	255.255.255.192	1022	62
11	255.255.255.224	2046	30
12	255.255.255.240	4096	14
13	255.255.255.248	8190	6
14	255.255.255.252	16382	2

The class C address subnetting example

The subnet mask for a class C address 192.168.2.0 that specifies 5 bits of subnetting is 255.255.255.248. With 5 bits available for subnetting, $2^5 - 2 = 30$ subnets possible. Now you have 3 bits left for the hosts part and it gives $2^3 - 2 = 6$ hosts per subnet. Table 3-4 shows number of hosts, number of subnets, and subnet mask depending on numbers of bits for subnet.

Table 3-4 Class C subnetting reference chart

Number of bits for subnet	Subnet mask	Number of subnets	Number of hosts
2	255.255.255.192	2	62
3	255.255.255.224	6	30
4	255.255.255.240	14	14
5	255.255.255.248	30	6
6	255.255.255.252	62	2

The class D address subnetting example

These addresses are reserved for multicasting (a sort of broadcasting, but in a limited area, and only to hosts using the same class D address). See “The multicast address” on page 47 for more details.

3.1.5 Supernetting

Whereas subnetting takes part of the host portion of the IP address and adds it to the network part portion, supernetting works the opposite way. It effectively reduces the number of bits used for the network portion. This technique allows a

number of class C addresses to be aggregated into a single address for routing purposes.

3.1.6 Address Resolution Protocol (ARP)

Machines on the same network must know each other's physical (or MAC) addresses in order to communicate. By broadcasting Address Resolution Protocol (ARP) packets, a host can dynamically discover the MAC-layer address corresponding to a particular IP Network Layer address.

To check the ARP addresses of interfaces on your system, enter:

```
# netstat -iv
Name  Mtu   Network     Address           Ipkts     Ierrs Opkts     Oerrs Coll
lo0   16896 link#1                        180084    0     180138    0     0
lo0   16896 127         loopback          180084    0     180138    0     0
lo0   16896 ::1                           180084    0     180138    0     0
tr0   1492  link#2      0.4.ac.61.73.f7   579283    0     38394     167   0
tr0   1492  9.3.240     server4           579283    0     38394     167   0
en0   1500  link#3      8.0.5a.fc.d2.e1   1690      0     2292      0     0
en0   1500  10.47       10.47.1.1         1690      0     2292      0     0
```

After detecting an IP-to-MAC address mapping, the system updates its ARP cache table to store the mapping, thus avoiding the need to broadcast ARP packets each time the system wants to contact the same network device. If the device is not recontacted or it does not broadcast ARP packets for a specified time, the cache entry is flushed. This is needed because if the device's adapter has been changed, it has a new MAC address with the same IP address and your system would still have the old entry in the table.

To check the ARP cache on your system, enter the **arp** command:

```
# arp -a
  server3.itsc.austin.ibm.com (9.3.240.58) at 0:6:29:be:d2:a2 [token ring]
  ? (9.3.240.108) at 0:20:35:fe:49:18 [token ring]
  eagle.itsc.austin.ibm.com (9.3.240.68) at 0:20:35:7c:9:fa [token ring]
  ? (9.3.240.100) at 0:6:29:f0:e1:c [token ring]
  ? (9.3.240.75) at 0:6:29:1:a:ba [token ring]
  itso240.itsc.austin.ibm.com (9.3.240.1) at 8:0:5a:fe:21:7 [token ring]
  dhcp240.itsc.austin.ibm.com (9.3.240.2) at 0:20:35:29:b:6d [token ring]
  ? (9.3.240.103) at 0:20:35:fe:4b:5b [token ring]
  server1.itsc.austin.ibm.com (9.3.240.56) at 0:6:29:be:b1:dc [token ring]
  server2.itsc.austin.ibm.com (9.3.240.57) at 0:4:ac:61:9d:c5 [token ring]
```

The ARP cache table entry contains the:

- Host name, if it only can be resolved.
- IP address.

- MAC address.
- Hardware interface type, such as token-ring or Ethernet.

Entries in the ARP mapping table are deleted after 20 minutes; incomplete entries are deleted after 3 minutes. To make a permanent entry in the ARP mapping tables, use the **arp** command with the pub parameter. Following is an incomplete ARP entry example:

```
server5.mycompany.example (9.3.4.29) at (incomplete)
```

When any host that supports ARP receives an ARP request packet, the host notes the IP and hardware addresses of the requesting system and updates its mapping table, if necessary. If the receiving host IP address does not match the requested address, the host discards the request packet. If the IP address does match, the receiving host sends a response packet to the requesting system. The requesting system stores the new mapping and uses it to transmit any similar pending Internet packets.

3.2 Routing

Routing allows information to be directed from a source host to a destination host in another network. There are two types of routing in TCP/IP: static routing and dynamic routing.

If you want two networks to communicate with each other, you can connect them through one machine, called a router (gateway). This machine must be physically on both networks. A router contains the addressing and routing information (routing table) for each host on its network, and may use routing daemons to broadcast routing information to, and receive routing information from, other routers. TCP/IP routes packets to the appropriate computer on the network, using its destination IP address by consulting a routing table.

TCP/IP searches the routing table for a best-fit match in following order:

1. *Host route* defines a gateway that can forward packets to a specific host or gateway on another network.
2. *Network route* defines a gateway that can forward packets to any of the hosts on a specific network.
3. *Default route* defines a gateway to use when a host or network route to a destination is not otherwise defined.

3.2.1 An introduction to static and dynamic routing

Static routing is simple table mappings established by the network administrator prior to the beginning of routing. These mappings do not change unless the network administrator alters them. Algorithms that use static routes are simple to design and work well in environments where network traffic is relatively predictable and where network topology is simple.

Because static routing systems cannot react to network changes, they generally are considered unsuitable for today's large, changing networks. Most of the dominant routing algorithms now are dynamic routing algorithms, which adjust to changing network circumstances by analyzing incoming routing update messages. If the message indicates that a network change has occurred, the routing software recalculates routes and sends out new routing update messages. These messages cross the network, stimulating routers to rerun their algorithms and change their routing tables accordingly.

Dynamic routing algorithms can be supplemented with static routes where appropriate. A *router of last resort* (a router to which all unroutable packets are sent), for example, can be designated to act as a repository for all unroutable packets, ensuring that all of them are at least handled in some way. There are two daemons in AIX responsible for dynamic routing: routed and gated.

The gated daemon supports Routing Information Protocol (RIP), Exterior Gateway Protocol (EGP), Border Gateway Protocol (BGP), Defense Communications Network Local-Network Protocol (HELLO), Open Shortest Path First (OSPF), and many others. The routed daemon supports only Routing Information Protocol (RIP).

Routing daemons can operate in one of two modes, passive or active, depending upon the options you use when starting the daemons. In active mode, routing daemons broadcast routing information periodically about their local network to gateways and hosts and receive routing information from hosts and gateways. In passive mode, routing daemons receive routing information from hosts and gateways, but do not attempt to keep remote gateways updated (they do not advertise their own routing information).

Note: You may decide to use a combination of static and dynamic routing. That is, you might want to give static definitions to a few specific routes, while allowing other routes to be updated by the daemons. The static routes you create are not advertised to other gateways and are not updated by the routing daemons.

3.2.2 Static routing

Routes are defined in the kernel routing table. These route definitions include information on networks reachable from the local host, gateways that can be used to reach remote networks, and the hop count (or distance metric) to those networks. When a gateway receives a packet, it checks the routing tables to obtain where to send the packet next along the path to its destination. To display the routing table on your machine, use the **netstat** command:

```
# netstat -nr
Routing tables
Destination      Gateway           Flags    Refs     Use  If   PMTU  Exp  Groups

Route Tree for Protocol Family 2 (Internet):
default          9.3.240.1         UGc       0        0  tr0     -   -
9.3.240/24       9.3.240.58        U        33   128221  tr0     -   -
10.47/24        9.3.240.59        UGc 0        0  tr0     -   -
127/8            127.0.0.1         U        54     1284  lo0     -   -
195.116.119/24   195.116.119.2     U         6    21313  en0     -   -

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

Using the **netstat** command output shown above, you can find out that:

- The default gateway for that machine is the router with IP address 9.3.240.1.
- To reach hosts on the local network 9.3.240.0, the machine will use its own interface tr0 with IP address 9.3.240.58.
- To reach hosts on the remote network 10.47.0.0, the machine will forward all packets to the host with IP 9.3.240.59 through interface tr0.
- To reach hosts on the local network 195.116.119.0, the machine will forward all packets to its own interface en0 with IP address 195.116.119.2.

As shown, entries have different flags that show the state of the route, as follows:

U Up.

H The route is to a host rather than to a network.

G The route is to a gateway.

D The route was created dynamically by a redirect.

M The route has been modified by a redirect.

L The link-level address is present in the route entry.

c Access to this route creates a cloned route.

W The route is a cloned route.

There are three methods to add a route to a routing table: implicit and explicit methods, or by adding a dynamic routing protocol such as RIP. The implicit method is performed when you configure the adapter. Follow the example to see how the implicit method works. First remove the en0 interface and then check which network interfaces are already configured:

```
# ifconfig en0 detach
# netstat -i
Name  Mtu   Network     Address            Ipkts Ierrs    Opkts Oerrs  Coll
lo0   16896 link#1                         201414     0   201508     0     0
lo0   16896 127         localhost.austin.  201414     0   201508     0     0
lo0   16896 ::1                           201414     0   201508     0     0
tr0   1492  link#2      0.4.ac.61.73.f7    632486     0    49983   167     0
tr0   1492  9.3.240     server4f.itsc.aus  632486     0    49983   167     0
```

As shown, there are two network interfaces: lo0 and tr0. To check current routing table, use the **netstat** command:

```
# netstat -nr
Routing tables
Destination      Gateway           Flags   Refs     Use  If   PMTU  Exp  Groups

Route Tree for Protocol Family 2 (Internet):
default          9.3.240.1         UGc       0        0  tr0     -   -
9.3.240/24       9.3.240.59        Uc        0        0  tr0     -   -
127/8            127.0.0.1         U         8     3489  lo0     -   -

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

As shown, the routing table contains three route definitions. Next add the new interface en0:

```
# ifconfig en0 10.47.1.1 netmask 255.255.0.0 up
```

Now the routing table has one entry more. This is a route associated with new interface en0:

```
# netstat -nr
Routing tables
Destination      Gateway           Flags   Refs     Use  If   PMTU  Exp  Groups

Route Tree for Protocol Family 2 (Internet):
default          9.3.240.1         UGc       0        0  tr0     -   -
9.3.240/24       9.3.240.59        Uc        0        0  tr0     -   -
10.47/16         10.47.1.1         Uc        0        0  en0     -   -
127/8            127.0.0.1         U         8     3489  lo0     -   -

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

The explicit routes are added by the network administrator. There are a few methods to add an entry to the routing table. The easiest way is to use **smitty mkroute**, shown in Figure 3-7. Configuring static routes through **smitty** adds them to the ODM databases and makes them permanent even after a system reboot.

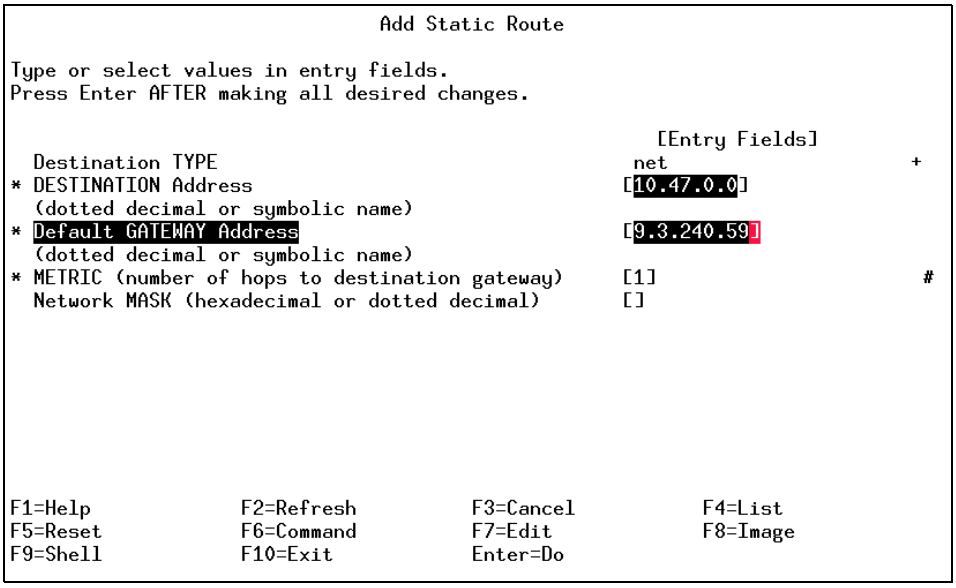


Figure 3-7 Configuring routing through smitty

The **smitty mkroute** command uses the **chdev** command so you can do the same job with the following commands:

```
chdev -l inet0 -a route='10.47.0.0','9.3.240.59'
```

The -C flag shows the routing tables, including the user-configured and current costs of each route.

The cost (also known as hopcount) prioritizes routes going to the same destination. If one or more routes go to a particular destination with a cost of 0 (zero), those routes are used, and routes with higher costs are not used. You should always give routes a cost greater than 0 to specify them as backup routes. If the lower cost routes are deleted, or if Dead Gateway Detection discovers a problem and raises their costs, the backup routes are used instead.

The user-configured cost is set using the -hopcount flag of the **route** command. The current cost may be different from the user-configured cost if Dead Gateway Detection has changed the cost of the route. The example below adds an entry to the routing table with a hopcount of 2:

```
# chdev -l inet0 -a route=net,-hopcount,2,9.3.4.100
inet0 changed
```

Another way to add an entry to the routing table is the **route** command. These entries are not permanent and will be lost after the next system reboot.

Routes to a particular host are distinguished from those to a network by interpreting the IP associated with the destination. The optional keywords -net and -host force the destination to be interpreted as a network or a host.

The **route** command does not update the ODM database, so if you want to make it permanent, include the **route** command entry in the /etc/rc.net (/etc/rc.bsdnet for Berkeley-style network configurations) file.

The following are examples using the **route** command:

- To establish a route to the computer with IP address 10.47.1.2 through the gateway with IP address 9.3.240.59, enter:

```
# route add 10.47.1.2 9.3.240.59
9.3.240.59 host 10.47.1.2: gateway 9.3.240.59
```

- To establish a route to network 10.47.0.0 through the gateway with IP address 9.3.240.59, enter:

```
# route add -net 10.47 9.3.240.59
9.3.240.59 net 10.47: gateway 9.3.240.59
```

- To establish a default gateway, enter:

```
# route add 0 9.3.240.1
9.3.240.1 net 0: gateway 9.3.240.1
```

The value `0` or the `default` keyword for the destination parameter means that any packet sent to destinations not previously defined and not on a directly connected network goes through the default gateway. The 9.3.240.1 address is that of the gateway chosen to be the default.

- To clear the host gateway table, enter:

```
# route -f
default             9.3.240.1           done
10.47               9.3.240.59          done
```

Configuring a system to work as static router

If your system is going to be configured as a router (it has two or more network interfaces), then it needs to be enabled as a router by the **no** command. The network option that controls routing from one network to another is **ipforwarding** and by default is disabled. To enable it, enter:

```
# no -o ipforwarding=1
```

This is not a permanent setting and after the next system reboot will be lost. To make this permanent, add this command to the end of /etc/rc.net file.

To check other network options and their values, enter the **no -a** command.

If your system has only one network interface, you can still use it as a router. Establish an additional network address for the interface using the **ifconfig** command with the **alias** parameter. To setup an additional IP address for interface en0, type:

```
ifconfig en0 10.50.1.1 netmask 255.255.0.0 alias
```

Check the settings for en0 interface:

```
# ifconfig en0
en0:
flags=e080863<UP,BROADCAST,NOTRAILERS,RUNNING,SIMPLEX,MULTICAST,GROUPRT,64IT>
        inet 10.47.1.1 netmask 0xffff0000 broadcast 10.47.255.255
        inet 10.50.1.1 netmask 0xffff0000 broadcast 10.50.255.255
```

Now the system has two different addresses; however, it can route packets between networks using one interface. If you check the routing table, you will find a new entry associated with network 10.50.0.0 and with interface en0:

```
# netstat -nr
Routing tables
Destination      Gateway           Flags   Refs     Use  If   PMTU  Exp  Groups

Route Tree for Protocol Family 2 (Internet):
10.47/16         10.47.1.1         Uc        0        0  en0     -   -
10.50/16         10.50.1.1         Uc        0        0  en0     -   -
127/8            127.0.0.1         U         7     3630  lo0     -   -

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

3.2.3 Dynamic routing

This section discuss the dynamic routing protocol.

Link-state versus distance-vector protocol

Link-state algorithms flood routing information to all nodes in the internetwork. Each router, however, sends only the portion of the routing table that describes the state of its own links. Distance-vector algorithms call for each router to send all or some portion of its routing table, but only to its neighbors. Link-state algorithms send small updates everywhere, while distance-vector algorithms send larger updates only to neighboring routers. Link-state algorithms converge more quickly and are less prone to routing loops than distance-vector algorithms.

On the other hand, link-state algorithms require more CPU power and memory than distance vector algorithms.

Routed daemon

The routed daemon is responsible for managing the network routing tables in the kernel. If multiple interfaces are present, the routed daemon assumes that the local host forwards packets between networks and transmits a RIP request packet on each interface, using a broadcast message.

The routed daemon then listens for RIP routing requests and response packets from other hosts. When the routed daemon supplies RIP information to other hosts, it sends RIP update packets every 30 seconds (containing copies of its routing tables) to all directly connected hosts and networks.

When the routed daemon receives a RIP request packet to supply RIP routing information, it generates a reply in the form of a response packet. Each route is marked with a hop-count metric, which is the number of gateway hops between the source network and the destination network. The metric for each route is relative to the sending host. A metric of 16 or greater is considered infinite or beyond reach.

Besides the ability of the routed daemon to manage routes to directly connected hosts and networks, it also uses distant and external gateways. These gateways cannot be identified by RIP queries, so the routed daemon reads the /etc/gateways file for information about these distant and external gateways. Its format is:

<destination> <name1> gateway <name2> metric <value> <type>

Following is a brief description of each element in a gateways file entry:

destination Keyword that indicates whether the route is to a network or a specific host. The two possible keywords are net and host.

name1 The name or IP address of destination.

name2 The name or IP address of the gateway host to which messages should be forwarded.

value The hop count, or number of gateways from the local network to the destination network.

type Keyword that indicates whether the gateway should be treated as active, passive, or external.

To specify a route to the network 10.47.0.0, through the gateway server4, add the following entry:

```
net 10.47.0.0 gateway server4 metric 1 passive
```

The routed daemon is a subsystem controlled by SRC and is a member of the tcpip system group. To start it in passive mode, enter:

```
# startsrc -s routed -a "-q"
0513-059 The routed Subsystem has been started. Subsystem PID is 22500.
```

The routed daemon is disabled by default, but if you uncomment the appropriate line in the /etc/rc.tcpip file, routing will start automatically after a system reboot.

You can also set up and start the routed daemon using the **smitty routed** command, as shown Figure 3-8.

```
          Change / Show Restart Characteristics of routed Subsystem

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                  [Entry Fields]
* LOG DEBUGGING information                       no                +
* This host is acting as a GATEWAY                no                +
* SUPPRESS sending routing information            yes               +
* DO supply routing information                   no                +
* Write all packets sent and received to STDOUT   no                +
  Write all PACKETS to LOGFILE                    []

F1=Help          F2=Refresh       F3=Cancel        F4=List
F5=Reset         F6=Command       F7=Edit          F8=Image
F9=Shell         F10=Exit         Enter=Do
```

Figure 3-8 smitty routed screen

Gated daemon

As mentioned in 3.2.1, “An introduction to static and dynamic routing” on page 56, the gated daemon provides gateway routing functions for a few routing protocols.

The gated daemon can be controlled by the SRC and it is a member of the SRC tcpip system group. This daemon is disabled by default. To permanently enable it, uncomment the appropriate line in the /etc/rc.tcpip and the gated daemon will start automatically after system reboot.

The default configuration file for the gated daemon is the /etc/gated.conf file. This file is read by the gated daemon at initialization time. By default, if the gated

daemon is started without specifying any information in the configuration file, the RIP protocol will be turned to active mode.

To start the gated daemon, use **smitty chgated** as shown Figure 3-9 or use the SRC command.

To start the gated daemon and log messages to /var/tmp/gated.log file, enter:

```
# startsrc -s gated -a "-tall /var/tmp/gated.log"
```

To stop the gated daemon normally, enter:

```
# stopsrc -s gated
```

```
            Change / Show Restart Characteristics of gated Subsystem

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                      [Entry Fields]
* LOG EGP EXTERNAL errors, routing errors             no                +
   and EGP state changes
* TRACE HELLO packets received                        no                +
* LOG INTERNAL errors and routing errors              no                +
* TRACE SNMP transactions                             no                +
* TRACE EGP packets sent and received                 no                +
* TRACE RIP packets received                          no                +
* TRACE all routing CHANGES                           no                +
* TRACE all routing UPDATES                           no                +
  LOGFILE name                                       []

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure 3-9 smitty chgated screen

Note: Results are unpredictable when the gated and routed daemons run on the same host.

3.2.4 ICMP redirects

ICMP generates several kinds of useful messages, including Destination Unreachable, Echo Request and Reply, Redirect, Time Exceeded, and Router Advertisement and Router Solicitation. If an ICMP message cannot be delivered, no second one is generated. This is to avoid an endless flood of ICMP messages.

An ICMP redirect message is sent by the router to the source host to stimulate more efficient routing. The router still forwards the original packet to the destination. ICMP redirects allow host routing tables to remain small because it is necessary to know the address of only one router, even if that router does not provide the best path. Even after receiving an ICMP redirect message, some devices might continue using the less efficient route.

If an ICMP redirect message is received from an intermediate router, it means that the host should send future datagrams for the network to the router whose IP address is specified in the ICMP message. This preferred router will always be on the same subnet as the host that sent the datagram and the router that returned the IP datagram. The router forwards the datagram to its next hop destination. This message will not be sent if the IP datagram contains a source route.

Figure 3-10 shows three IP networks connected by two routers, R1 and R2. For different destinations, workstation A uses different ways to send its IP packets to the destinations.

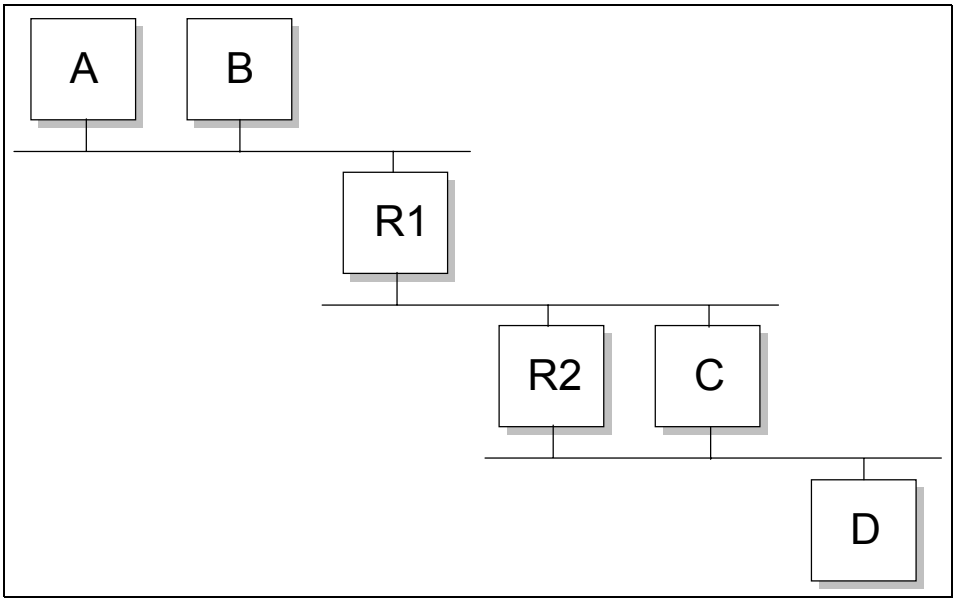


Figure 3-10 Routed network

In the above example, workstation C can elect either R1 or R2 as its default router. In the event that R1 is elected as the default router, C will send data to R1 when it needs to talk to A, B or D. Sending to A and B is straightforward: it passes the data to R1, which proceeds to forward the traffic to A or B. The tricky

part is when C wants to forward data to D. SInce R1 is the default router, all data will be forwarded to R1 from C. R1 is then going to realize that in order to reach D, it has to forward the traffic to R2. This bouncing of traffic from R1 to R2 will create extra delay and also extra traffic on the network.

To overcome this situation, routers implement the ICMP redirect, which informs workstation D that instead of sending the data to R1, it should instead send to R2. This would require workstation C to have the ability to handle ICMP redirect messages that were sent out by R1. Not all workstations support this feature and therefore, it is better to avoid designing the network in this manner.

To disable ICMP redirects on your server, use the following example:

```
# no -o ipsendredirects=0
```

This will update the network configurable attributes. The change will take effect immediatly and will be effective until the next boot. To make the change permanent, add the **no** command to /etc/rc.net.

3.2.5 Routing debugging

If you are not able to ping by host name or IP address, you may have a routing problem.

First, check the routing tables as follows:

- Use the **netstat -rn** command to show the content of your local routing table using IP addresses.
- Check the netmask on display and ensure that it is correct (ask the network administrator what it should be if you are unsure).
- If there is a default route, attempt to ping it.
- If you have more than one network interface, attempt to determine if any interfaces are working.

If you cannot ping your default route, either it is down, or your local network connection may be down. Attempt to ping all of the other gateways listed in the routing table to see if any portion of your network is functioning:

```
# netstat -nr
Routing tables
Destination      Gateway           Flags   Refs     Use  If   PMTU  Exp  Groups

Route Tree for Protocol Family 2 (Internet):
default          9.3.240.1         UGc       0        0  tr0     -   -
9.3.240/24       9.3.240.58        U        31   142091  tr0     -   -
10.47.1.2        9.3.240.59        UGH       0        2  tr0     -   -
127/8            127.0.0.1         UR        0        3  lo0     -   -
```

```
127.0.0.1        127.0.0.1         UH        3      761  lo0     -   -
195.116.119/24   195.116.119.2     U         2      406  en0     -   -

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

If you cannot ping any host or router interface from among those listed in the routing table, try to ping your loopback interface lo0 with the following command:

```
# ping localhost
```

If the ping is successful, you have either an adapter or network hardware problem or a routing problem.

If the ping is not successful, you need to:

- Ensure that the inetd process is active using the **lssrc -g tcpip** command. If inetd is not active, issue the **startsrc -s inetd** or **startsrc -g tcpip** commands.
- Check the state of the loopback interface (lo0) with the **netstat -i** command. If you see `lo0*` in the output, check the /etc/hosts file for an uncommented local loopback entry as follows:

  ```
  127.0.0.1 loopback localhost # loopback (lo0) name/address
  ```

  An asterisk (*) after the interface name in the output from the **netstat** command indicates that the interface is down. Use the following command to start the lo0 interface:

  ```
  # ifconfig lo0 inet 127.0.0.1 up
  ```

If you cannot reach a host which is in a different network, you can check the connection using the **traceroute** command. The traceroute output shows each gateway that the packet traverses on its way to find the target host. If possible, examine the routing tables of the last machine shown in the traceroute output to check if a route exists to the destination from that host. The last machine shown is where the routing is not functioning as intended.

```
# traceroute 9.3.240.56
traceroute to 9.3.240.56 (9.3.240.56), 30 hops max, 40 byte packets
 1  server4e (10.47.1.1)  1 ms  1 ms  0 ms
 2  server1 (9.3.240.56)  1 ms  1 ms  1 ms
```

If the connections are performing poorly, packet fragmentation may be a problem. AIX Version 4.3 has a service that allows automatic path MTU discovery. A fixed MTU size can also be set with the **no** command.

3.3 Command summary

The following section provides a list of the key commands discussed in this chapter. For a complete reference of the following commands, consult the AIX product documentation.

3.3.1 The ifconfig command

The **ifconfig** command configures or displays network interface parameters for a network using TCP/IP. The command has the following syntax:

```
ifconfig Interface [ AddressFamily [ Address [ DestinationAddress  ] ]
[ Parameters... ] ]
```

The commonly used flags are provided in Table 3-5.

Table 3-5 Commonly used flags of the ifconfig command

Flag	Description	
AddressFamily	Specifies which network address family to change.	
Parameters	alias	Establishes an additional network address for the interface.
	delete	Removes the specified network address.
	detach	Removes an interface from the network interface list.
	down	Marks an interface as inactive (down), which keeps the system from trying to transmit messages through that interface.
	netmask *Mask*	Specifies how much of the address to reserve for subdividing networks into subnetworks.
	up	Marks an interface as active (up). This parameter is used automatically when setting the first address for an interface.
Address	Specifies the network address for the network interface.	

3.3.2 The netstat command

The **netstat** command shows network status. The command has the following syntax:

```
/bin/netstat [ -n ] [ { -r  -i  -I  Interface } ] [ -f  AddressFamily ]
[ -p  Protocol ] [ Interval ]
```

The commonly used flags are provided in Table 3-6.

Table 3-6 Commonly used flags of the netstat command

Flag	Description
-n	Shows network addresses as numbers.
-r	Shows the routing tables.
-i	Shows the state of all configured interfaces.
-I *Interface*	Shows the state of the configured interface specified by the Interface variable.
-f *AddressFamily*	Limits reports of statistics or address control blocks to those items specified by the AddressFamily variable.
-p *Protocol*	Shows statistics about the value specified for the Protocol variable.
-m	Shows statistics recorded by the memory management routines.

3.3.3 The route command

The **route** command manually manipulates the routing tables. The command has the following syntax:

```
route Command [ Family ] [ [ -net | -host ] Destination
[-netmask  [ Address ] ] Gateway ] [ Arguments ]
```

The commonly used flags are provided in Table 3-7.

Table 3-7 Commonly used flags of the route command

Flag	Description	
Command	add	Adds a route.
	flush or -f	Removes all routes.
	delete	Deletes a specific route.
	get	Looks up and displays the route for a destination.
-net	Indicates that the Destination parameter should be interpreted as a network.	
-host	Indicates that the Destination parameter should be interpreted as a host.	
Destination	Identifies the host or network to which you are directing the route.	
-netmask	Specifies the network mask to the destination address.	

Flag	Description
Gateway	Identifies the gateway to which packets are addressed.

3.3.4 The chdev command

The **chdev** command changes the characteristics of a device. The command has the following syntax:

```
chdev -l Name [ -a Attribute=Value ... ]
```

The commonly used flags are provided in Table 3-8.

Table 3-8 Commonly used flags of the chdev command

Flag	Description
`-l Name`	Specifies the device logical name, specified by the Name parameter, in the Customized Devices object class whose characteristics are to be changed.
`-a Attribute=Value`	Specifies the device attribute value pairs used for changing specific attribute values.

3.3.5 The lsattr command

The **lsattr** command displays attribute characteristics and possible values of attributes for devices in the system. The command has the following syntax:

```
lsattr -E -l Name [ -a Attribute ] ...
```

The commonly used flags are provided in Table 3-9.

Table 3-9 Commonly used flags of the lsattr command

Flag	Description
-E	Displays the attribute names, current values, descriptions, and user-settable flag values for a specific device.
-l *Name*	Specifies the device logical name in the Customized Devices object class whose attribute names or values are to be displayed.
-a *Attribute*	Displays information for the specified attributes of a specific device or kind of device.

3.4 Quiz

The following assessment questions help verify your understanding of the topics discussed in this chapter.

1. Which one of the following network protocols can alter an otherwise static routing table?

   A. RPC

   B. TCP

   C. UDP

   D. ICMP

2. Which one of the following commands is needed to add an alias IP address onto an interface?

   A. **alias**

   B. **route**

   C. **netstat**

   D. **ifconfig**

3. Which one of the following commands can store routes in the ODM?

   A. **gated**

   B. **chdev**

   C. **route**

   D. **ifconfig**

4. Which one of the following address classes applies to 127.0.0.1?

   A. Class A

   B. Class B

   C. Class C

   D. Class D

5. Which one of the following network masks allows room for exactly 510 hosts?

   A. 255.128.0.0

   B. 255.254.0.0

   C. 255.255.254.0

   D. 255.255.255.128

6. A default gateway has already been configured. However, it begins to point to a different address, although it was not manually changed. Which one of the following is the most probable cause of the change in addresses?

   A. **arp**

   B. **netstat**

   C. NIS, DNS, NFS

   D. **gated** or **routed**

7. On a newly installed AIX Version 4 machine, which one of the following actions will enable the machine to act as a gateway?

   A. Enable gated

   B. Enable routed

   C. Enable ipforwarding

   D. Configure a default gateway

8. Which one of the following commands can show statistics for each interface?

   A. **no**

   B. **lsattr**

   C. **vmstat**

   D. **netstat**

9. Which one of the following commands verifies that round-trip connectivity is functional between the local host and another machine?

   A. **ping**

   B. **lsdev**

   C. **netstat**

   D. **ifconfig**

10. A local subnet can be pinged as well as the default gateway. However, the hosts that are beyond the default gateway cannot be pinged. Which one of the following is the most probable cause?

    A. ARP is not functioning

    B. IP forwarding is not on for the local system

    C. The default gateway is not routing traffic for the host

    D. Something has been configured incorrectly on the local machine

11. Which one of the following commands will successfully add a route to the routing table?

    A. **`route add 128.66.12.3 0`**

    B. **`route add 0 128.66.12.3`**

    C. **`route -n add 128.88.12.1`**

    D. **`route add 128.66 128.88.12.1`**

12. Which one of the following parameters can prevent packet fragmentation on routers when connecting to remote networks?

    A. mtu

    B. sb_max

    C. rfc1323

    D. tcp_mssdflt

13. If AIX is configured to forward IP packets from one network to another network, which one of the following is true?

    A. AIX has no IP forwarding capabilities.

    B. The two network addresses must be placed in the forward file.

    C. Two network adapter cards would require the ipforwarding flag to be set to "1".

    D. The broadcast flag does not need to be set on the network adapter cards using the **`ifconfig`** command.

14. An ATM backbone has been subnetted so that each subnet contains 14 hosts. Which one of the following is the subnet mask?

    A. 255.255.255.14

    B. 255.255.255.16

    C. 255.255.255.240

    D. 255.255.255.248

15. Which one of the following **`no`** options should be set to “1” before the machine can act as a gateway?

    A. ipforwarding

    B. multi_homed

    C. ipsrcrouteforward

    D. subnetsarelocal

16. In AIX Version 4, which one of the following options is an invalid source IP address?

    A. 1.0.0.1

    B. 1.1.1.1

    C. 127.0.0.1

    D. 240.240.240.240

17. Which one of the following statements correctly describes the following address: 132.120.107.11 using a default network for the address class?

    A. 132.120.107.11 is a multicast address

    B. 132 is the network address and 120.107.11 is the local host address

    C. 132.120 is the network address and 107.11 is the local host address

    D. 132.120.107 is the network address and 11 is the local host address

18. Which one of the following commands is used to alter the cable type used for an installed adapter?

    A. **lscfg**

    B. **chdev**

    C. **cfgmgr**

    D. **ifconfig**

19. Which one of the following commands will permanently change the IP address of a network interface?

    A. **no**

    B. **alias**

    C. **chdev**

    D. **confsetcntrl**

20. Which one of the following options is *not* affected by the smit tcpip minimum configuration screen?

    A. /etc/hosts

    B. /etc/resolv.conf

    C. routing tables

    D. tunable network options

21. An entry in an ARP table reads incomplete. Which one of the following statements has most likely occurred?

    A. The hardware address at the host has changed.

B. The machine can still be pinged but not telneted.

C. An ARP request and reply was processed at the same time.

D. An ARP request was transmitted looking for that machine, but no reply was received.

22. Which one of the following procedures would set the hop count for a route to 5 for a specific Ethernet interface, provided that the destination IP address is B.C.D.E and the gateway is A.B.C.D?

A. **net B.C.D.E gateway A.B.C.D metric 5 passive**

B. **ifconfig en0 B.C.D.E netmask 255.255.255.0 metric 5**

C. `route add B.C.D.E A.B.C.D -hopcount 5`

D. **no -o ip6_defttl=5**

23. The routing term hop count is given another name on the SMIT route menus. What is this alternate name?

A. time-to-live

B. cost

C. time exceeded

D. metric

24. Which address range is routable and requires coordination with IANA?

A. 10.0.0.0 - 10.255.255.255

B. 172.16.0.0 - 172.31.255.255

C. 192.168.0.0 - 192.168.255.255

D. 193.0.0.0 - 255.255.255.255

3.4.1 Answers

The following are the preferred answers to the questions provided in this section:

1. D
2. D
3. B
4. A
5. C
6. D
7. C
8. D
9. A
10. C
11. B
12. A
13. C
14. C
15. A
16. D
17. C
18. B
19. C
20. D
21. D
22. C
23. B
24. D

3.5 Exercises

The following exercises provide sample topics for self study. They will help ensure comprehension of this chapter.

1. Calculate how many hosts and networks are within network class B with subnetmask 255.255.255.192.
2. For given host address 153.19.177.201 with subnet mask of 255.255.225.224, determine the network address and broadcast address.
3. Check the routing table for your system and find out what the default gateway is for.
4. Add a second address to the network card in your machine. Use the **`ifconfig`** command.
5. Check the routing table for your system. Do you have another routing entry now?
6. Which protocol will modify routes?
7. On which port does telnet listen and on which port does FTP listen?

4

Basic network administration

Basic network administration including name address configuration is discussed in this chapter. For a discussion on DNS, see Chapter 8, “Domain Name System” on page 193.

4.1 Network administration using SMIT

The following sections discuss how to perform basic network administration using the SMIT interface.

4.1.1 Minimum configuration

The minimum configuration of TCP/IP is typically done at initial installation or when an adapter and corresponding interface need to be installed. Issue the SMIT command: **`smitty tcpip`** and the screen shown in Figure 4-1 appears.

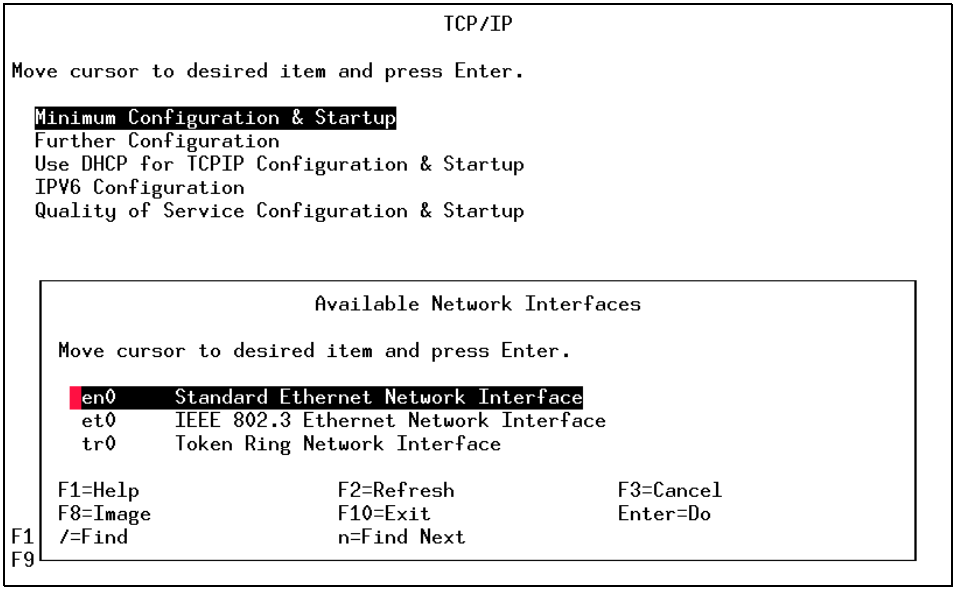


Figure 4-1 SMIT TCP/IP configuration screen

Select the **Minimum Configuration & Startup** menu and select the interface that needs to be configured from the list that is presented (Figure 4-2 on page 81).

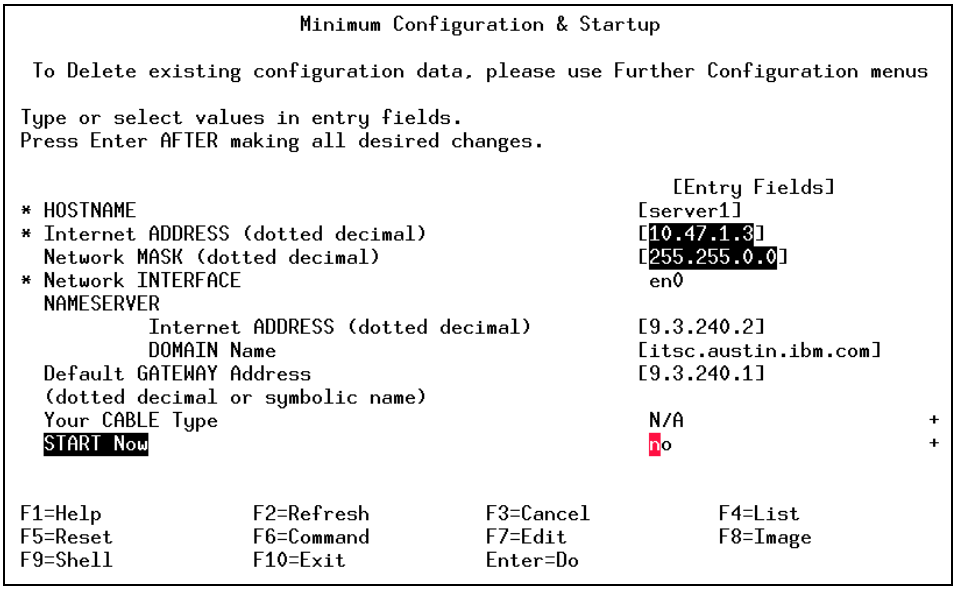


Figure 4-2 SMIT TCP/IP minimum configuration parameters screen

Figure 4-2 shows the SMIT screen that is used to enter the minimum configuration values. The SMIT menus use the program **mktcpip** to perform the actual TCP/IP configuration. The functions performed by the **mktcpip** command are:

- Setting the host name in both the configuration database and the running machine.
- Setting the IP address of the interface in the configuration database.
- Making entries in the /etc/hosts file for the host name and IP address.
- Setting the subnetwork mask, if specified.
- Setting the domain name and IP address of the name server, if specified.
- Adding a static route to both the configuration database and the running machine, if applicable.
- Starting or restarting the default TCP/IP daemons.

4.1.2 Further TCP/IP configuration

When performing a more detailed TCP/IP system administration, use the **smitty configtcp** command.

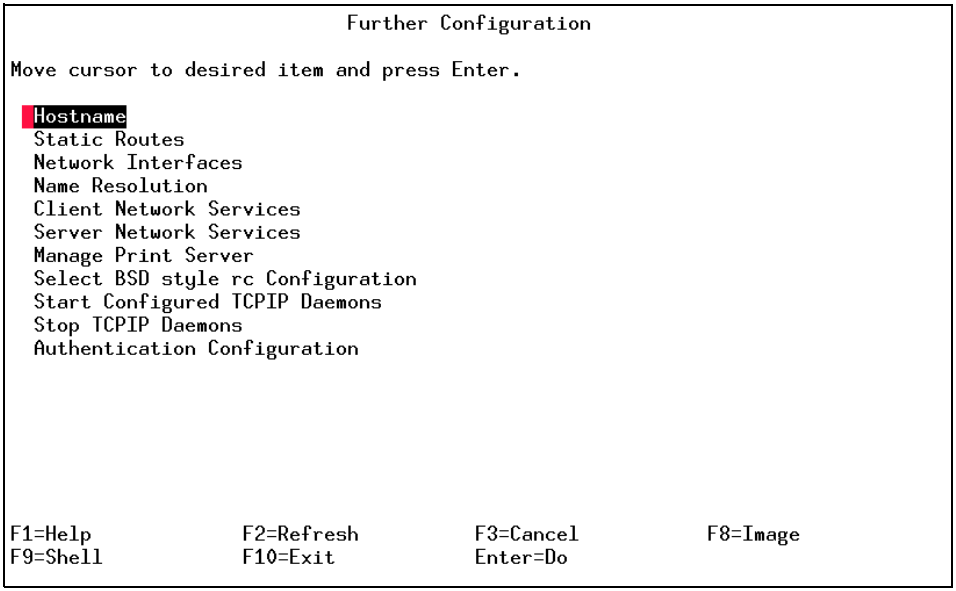


Figure 4-3 SMIT TCP/IP Further Configuration screen

The SMIT menu for further TCP/IP configuration (**`smitty configtcp`**) assists you in the administration of the following topics:

Hostname	Show and set the system host name.
Static Routes	List, add, delete routes, flush routing table.
Network Interfaces	List, add, change and remove network interfaces.
Name Resolution	List and edit contents of /etc/hosts and /etc/resolv.conf.
Client Network Services	Edit /etc/services file.
Server Network Services	Start/stop network daemons and network services. Menus to SRC commands.
Manage Print Server	List, add, remove network printer daemons.
Select BSD style rc Configuration	Modify TCP/IP bootup procedure to use /etc/rc.bsdnet instead of the default /etc/rc.net.
Start Configured TCPIP Daemons	Start all configured TCP/IP daemons.
Stop TCPIP Daemons	Stop all running TCP/IP daemons.
Authentication Configuration	Configure Kerberos 4 or Kerberos 5 authentication. Default is standard AIX.

4.1.3 Setting the host name

After your machine has an IP address you have to name it. The **hostname** command sets and displays the name of the current host system. Only users with root user authority can set the host name. The **chdev** command will also set the host name, but it does it permanently. You can also use the **smit mkhostname** fast path to run this command.

To check the host name, enter:

```
# hostname
server3
```

You can do the same job using the **chdev** command:

```
# chdev -l inet0 -a hostname=server3
inet0 changed
```

This will change the host name permanently. Now you can check the host name:

```
# lsattr -El inet0 -a hostname -F value
server3
```

4.1.4 Host name resolution

In simple TCP/IP networks, all machines on the network are defined with a name that has a corresponding IP address. The mapping of names to IP addresses is stored in the /etc/hosts file, acting as a simple lookup database. As most TCP/IP networks are very large and might be connected to the Internet, a different name resolution scheme is needed. These TCP/IP networks use the domain name system (DNS/BIND) having DNS server daemons (named) acting as databases responding to host name lookup. For more information on DNS, see Chapter 8, “Domain Name System” on page 193.

Note that TCP/IP host name lookup is also referred to as host name resolving. This resolution is done by all programs that want to communicate over a TCP/IP network (see man page on gethostbyname library call).

By default, the resolver routines first attempt to resolve names using the following priority scheme:

- DNS/BIND using the /etc/resolv.conf
- NIS (see Chapter 10, “NIS” on page 223)
- Look up in the /etc/hosts file

The default order can be changed by creating the configuration file /etc/netsvc.conf and specifying a different search order.

The environment variable NSORDER overrides both the /etc/netsvc.conf file and the default ordering. Services are ordered as hosts = value, value, value in the /etc/netsvc.conf file, where at least one value must be specified from the list bind, nis, local. NSORDER specifies a list of values.

Example of changing the NSORDER:

```
# ping -c 1 server2
PING server2.itsc.austin.ibm.com: (9.3.240.57): 56 data bytes
64 bytes from 9.3.240.57: icmp_seq=0 ttl=255 time=0 ms

----server2.itsc.austin.ibm.com PING Statistics----
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0/0/0 ms
# export NSORDER=local,bind,nis
# ping -c 1 server2
PING server2: (9.3.240.57): 56 data bytes
64 bytes from 9.3.240.57: icmp_seq=0 ttl=255 time=0 ms

----server2 PING Statistics----
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0/0/0 ms
```

Notice the missing domain name in the second **ping** command.

Changing the resolver priority scheme must be used with caution, but may be necessary in cases where the DNS servers are not responding.

resolv.conf

The /etc/resolv.conf file defines the DNS name server information for local resolver routines. If the /etc/resolv.conf file does not exist, the DNS is not available and the system will attempt name resolution using the default paths, the /etc/netsvc.conf file (if it exists), or the NSORDER environment variable (if it exists).

When a DNS server is specified during TCP/IP configuration, a /etc/resolv.conf file is generated. Further configuration of the resolv.conf file can be done using the SMIT command **smit resolv.conf (**Figure 4-4 on page 85).

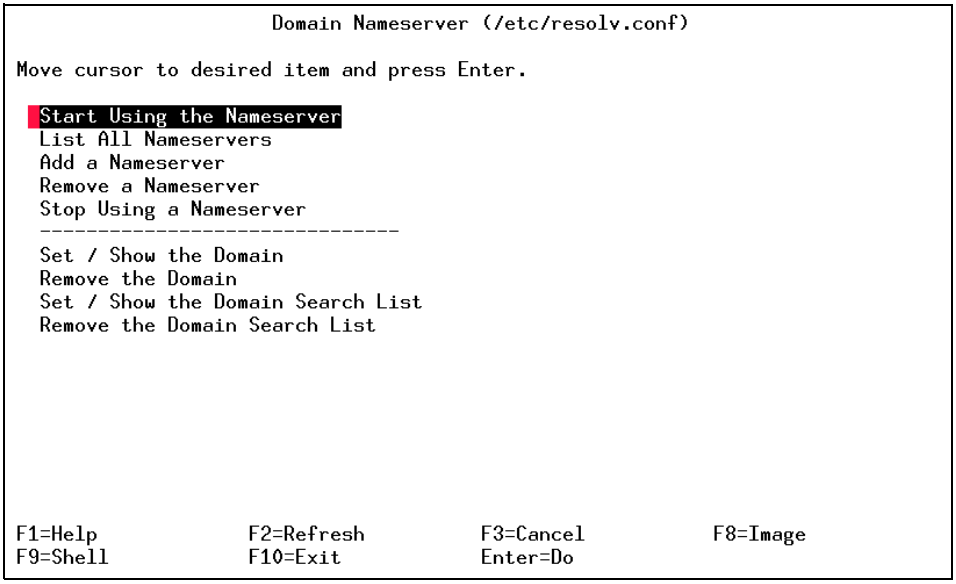


Figure 4-4 SMIT menu for resolv.conf

Following is an example of a resolv.conf file:

```
# cat /etc/resolv.conf
nameserver      9.3.240.2
nameserver      9.53.248.2
nameserver      9.53.183.2
domain          itsc.austin.ibm.com
```

Following are the valid entry format in the resolv.conf file:

- A domain entry tells the resolver routines which default domain name to append to names that do not end with a . (period). There can be only one domain entry. This entry is of the form:

  `domain` *`DomainName`*

- A search entry defines the list of domains to search when resolving a name. Only one domain entry or search entry can be used. If the domain entry is used, the default search list is the default domain. A search entry should be used when a search list other than the default is required. The entry is of the form:

  `search` *`DomainName`* `...`

  The search entry can have from one to six DomainName variables.

- A nameserver entry defines the Internet address of a remote DOMAIN name server to the resolver routines on the local domain. This entry is of the form:

  `nameserver` *`Address`*

- The options entry specifies miscellaneous behaviors of the resolver. The entry is of the form:

  `options` *`OptionName`*

  The OptionName variable can have one of the following values:

  `debug` Turns on the RES_DEBUG resolver option, which enables resolver debugging.

  `ndots:n` Specifies that for a domain name with n or more periods (.) in it, the resolver should try to look up the domain name "as is" before applying the search list.

Each nameserver entry specifies the IP address of the DNS name server to use. In this example, three name servers are defined. The local resolver routines will query each domain name server for name resolution. When multiple name servers are specified, if the first name server does not respond, then the next name server in the list is queried.

The entry domain is used for the default domain name. The local resolver appends the default domain to names that do not end with a . (period).

Instead of domain you can use the entry *search*. The search entry defines the list of domains to search when resolving a name. The first domain entry is interpreted as the default domain. Note that the usage of domain or search is complementary.

4.1.5 Network interface configuration

If you get an IP address and netmask of your machine from a network administrator, you have enough information to set up a network interface. Though SMIT allows you a shortcut to this method, many programmers wish to learn how to configure the interfaces directly.

First, list all your network interfaces:

```
# lsdev -Cc if
en0 Available  Standard Ethernet Network Interface
et0 Defined    IEEE 802.3 Ethernet Network Interface
lo0 Available  Loopback Network Interface
tr0 Available  Token Ring Network Interface
```

As shown, there are three interfaces that you could use: `en0`, `et0`, and `tr0`. To configure one of them, use **`smitty chinet`** as shown in Figure 4-5 on page 87.

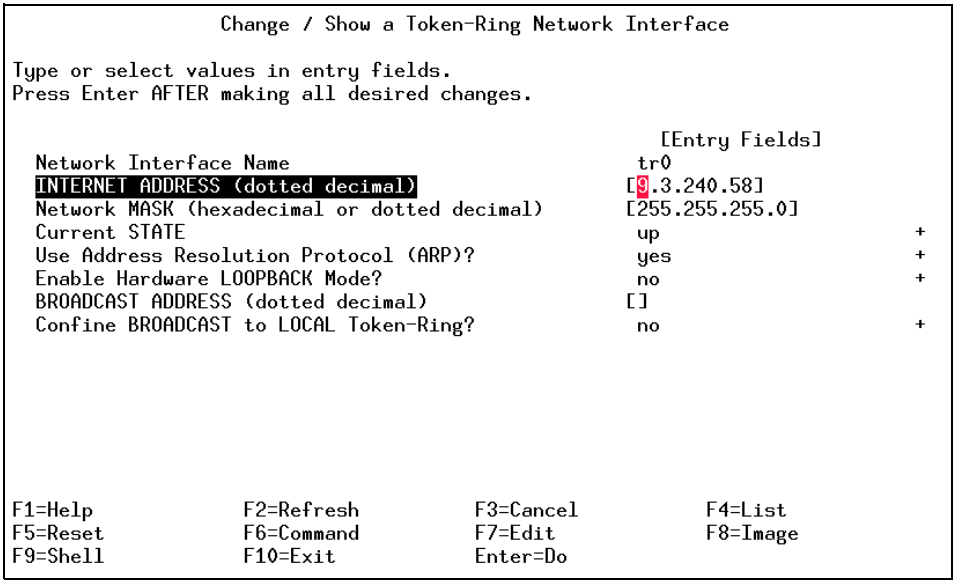


Figure 4-5 smitty chinet screen

You can do the same job using the **chdev** command for the appropriate interface:

```
# chdev -l en0 -a netaddr='9.3.240.58' -a netmask=255.255.255.0'
en0 changed
```

smitty chinet and **chdev** update the ODM database and the change will be permanent. Another way to change network interface characteristics is by using the **ifconfig** command, but this does not update the ODM database. The **ifconfig** command can assign an address to a network interface and can configure or display the current network interface configuration information. The network interface configuration is held on the running system and must be reset after each system restart.

To query the status of an en0 interface, enter the command in the following format:

```
# ifconfig en0
en0:
flags=e080863<UP,BROADCAST,NOTRAILERS,RUNNING,SIMPLEX,MULTICAST,GROUPRT,64BIT>
        inet 195.116.119.2 netmask 0xffffff00 broadcast 195.116.119.255
```

To mark the local Ethernet interface en0 as down, enter:

```
ifconfig  en0 inet  down
```

Finally to set up the IP address 195.116.119.2 with a netmask of 255.255.255.0 for interface en0, enter the command in the following format:

```
# ifconfig en0 195.116.119.2 netmask 255.255.255.0 up
```

You can also use **iptrace** command to record packets exchanged on an interface from a specific remote host. The **iptrace** daemon, invoked by **iptrace** command, records Internet packets received from configured interfaces. Command flags provide a filter so that the daemon traces only packets meeting specific criteria. Packets are traced only between the local host on which the **iptrace** daemon is invoked and the remote host. The LogFile parameter specifies the name of a file to which the results of the **iptrace** command are sent. See 7.3.2, “Controlling server daemons” on page 160 and 7.7.5, “The iptrace command” on page 183 for more details.

4.1.6 The prtconf command

The **prtconf** command located in /usr/sbin/prtconf displays system configuration information. The output includes the total amount of memory, and the configuration of system peripherals formatted as a device tree. The **prtconf** command is useful to show slot placements. The **prtconf** command has been included in AIX 5L and later versions. Running **prtconf** on an IBM RS/6000 Model F80 produces the following sample output:

```
# prtconf
System Model: IBM,7025-F80
Processor Type:  PowerPC_RS64-III
Number Of Processors:        4
Memory Size: 1024MB
Good Memory Size: 1024MB
Firmware Version: IBM,M2P01072
Console Login: enable
Auto Restart: false
Full Core: false

Network Information
        Host Name: server4.itsc.austin.ibm.com
        IP Address: 9.3.4.100
        Sub Netmask: 255.255.254.0
        Gateway: 9.3.4.41
        Name Server: 9.3.4.2
        Domain Name: itsc.austin.ibm.com

Paging Space Information
        Total Paging Space: 512MB
        Percent Used: 1%

Volume Groups Information
```

```
================================================================================
rootvg:
PV_NAME           PV STATE          TOTAL PPs   FREE PPs    FREE DISTRIBUTION
hdisk0            active            542         120         92..00..00..00..28
hdisk1            active            542         539
109..108..105..108..
109
================================================================================

testvg:
PV_NAME           PV STATE          TOTAL PPs   FREE PPs    FREE DISTRIBUTION
hdisk2            active            542         533
109..99..108..108..1
09
================================================================================

0516-010 : Volume group must be varied on; use varyonvg command.
================================================================================

INSTALLED RESOURCE LIST

The following resources are installed on the machine.
+/- = Added or deleted from Resource List.
*   = Diagnostic support not available.

  Model Architecture: chrp
  Model Implementation: Multiple Processor, PCI bus

+ sys0              00-00             System Object
+ sysplanar0        00-00             System Planar
+ mem0              00-00             Memory
+ proc0             00-00             Processor
+ L2cache0          00-00             L2 Cache
* pmc0              00-00             n/a
+ proc2             00-02             Processor
+ proc4             00-04             Processor
+ proc6             00-06             Processor
* pci0              00-fff7f09000     PCI Bus
* isa0              10-80             ISA Bus
+ fda0              01-D1             Standard I/O Diskette Adapter
+ fd0               01-D1-00-00       Diskette Drive
* siokma0           01-K1             Keyboard/Mouse Adapter
+ sioka0            01-K1-00          Keyboard Adapter
+ kbd0              01-K1-00-00       PS/2 keyboard
+ sioma0            01-K1-01          Mouse Adapter
+ mouse0            01-K1-01-00       3 button mouse
+ ppa0              01-R1             CHRP IEEE1284 (ECP) Parallel Port
                                      Adapter
+ sa0               01-S1             Standard I/O Serial Port
```

```
+ lp1              01-S1-00-00       IBM 4029 LaserPrinter
+ sa1              01-S2             Standard I/O Serial Port
+ lp0              01-S2-00-00       Other serial printer
+ sa2              01-S3             Standard I/O Serial Port
+ sa3              01-S4             Standard I/O Serial Port
* pci2             10-58             PCI Bus
+ scsi0            11-08             Wide/Ultra-2 SCSI I/O Controller
+ rmt0             11-08-00-0,0      SCSI 8mm Tape Drive (20000 MB)
+ cd0              11-08-00-1,0      16 Bit SCSI Multimedia CD-ROM Drive
                                     (650 MB)
+ hdisk0           11-08-00-2,0      16 Bit LVD SCSI Disk Drive (9100 MB)
+ hdisk1           11-08-00-4,0      16 Bit LVD SCSI Disk Drive (9100 MB)
+ scsi1            11-09             Wide/Ultra-2 SCSI I/O Controller
* pci3             10-5a             PCI Bus
* pci4             10-5c             PCI Bus
* pci5             10-5e             PCI Bus
+ tok0             1A-08             IBM PCI Tokenring Adapter (14103e00)
* pci1             00-fff7f0a000     PCI Bus
* pci6             20-58             PCI Bus
+ ent0             21-08             IBM 10/100 Mbps Ethernet PCI Adapter
                                     (23100020)
* pci7             20-5a             PCI Bus
+ scsi2            27-08             Wide/Ultra-2 SCSI I/O Controller
+ hdisk2           27-08-00-8,0      16 Bit LVD SCSI Disk Drive (9100 MB)
+ hdisk3           27-08-00-9,0      16 Bit LVD SCSI Disk Drive (9100 MB)
+ ses0             27-08-00-15,0     SCSI Enclosure Services Device
+ scsi3            27-09             Wide/Ultra-2 SCSI I/O Controller
+ hdisk4           27-09-00-8,0      16 Bit LVD SCSI Disk Drive (9100 MB)
+ hdisk5           27-09-00-9,0      16 Bit LVD SCSI Disk Drive (9100 MB)
+ ses1             27-09-00-15,0     SCSI Enclosure Services Device
* pci8             20-5c             PCI Bus
* pci9             20-5e             PCI Bus
+ mg20             2D-08             GXT130P Graphics Adapter
* pci10            20-60             PCI Bus
* pci11            20-62             PCI Bus
* pci12            20-64             PCI Bus
* pci13            20-66             PCI Bus
```

4.1.7 The TTY configuration

AIX is a multiuser operating system that allows user access from local or remote attached devices. The communication layer that supports this function is the TTY subsystem. The communication between terminal devices and the programs that read and write to them is controlled by the TTY interface. Examples of TTY devices are:

- Modems
- ASCII terminals

- System console
- Serial printer
- System console
- Xterm or aixterm under X-Windows

Following is an example of how to configure a TTY console on serial port 0 with login enable (if the TTY is used as a remote console, failure to enable the login will result in a CF1 error and a message to select the console after boot):

```
# mkdev -c tty  -t tty -s rs232 -p sa0 -w 0 -a login=enable
tty0 Available
```

To validate that the TTY has been added to the customized VPD object class, enter:

```
# lscfg -vp|grep tty
  tty0             01-S1-00-00       Asynchronous Terminal
```

To display the full path name of the system console effective on the next startup of the system, enter:

```
# lscons -b
/dev/tty0
```

To remove a TTY, first disable the login. For example, to disable login for tty0, enter:

```
# chdev  -l tty0 -a login=disable
tty0 changed
```

To remove tty0, enter:

```
# rmdev  -l tty0 -d
tty0  deleted
```

4.1.8 Asynchronous Terminal Emulation

The **ate** command starts the Asynchronous Terminal Emulation (ATE) program. ATE establishes a connection between a workstation and a remote computer. A workstation acts as a terminal connected to the remote computer. Using ATE you can connect to, and exchange data with, remote databases and other systems.

Note: Users must be members of the UNIX-to-UNIX Copy Program (uucp) group in order to use ATE. A user with root authority uses the SMIT to install individual users in groups.

ATE establishes the connection and allows users to record and control the session. After logging in to the remote system, a user executes programs, issues

commands, and uses files on the remote system as a local user. ATE also enables a workstation to emulate a VT100 terminal.

ATE uses menus and subcommands. From the menus, users issue subcommands to connect to a remote system, receive and transfer files, and execute commands. The Unconnected Main Menu displays any time users issue the **ate** command. The Connected Main Menu displays when users press the MAINMENU_KEY (usually the Ctrl+V key sequence) while connected to another system. The **connect** subcommand makes the connection.

The ATE program supports three control key sequences: the CAPTURE_KEY (usually Ctrl+B), PREVIOUS_KEY (usually CTRL+-R), and MAINMENU_KEY (usually CTRL+V). These control keys do not function until the ATE program is started. The control keys and other ATE defaults can be changed by editing the ate.def file format.

To start ATE, enter:

```
# ate
```

Table 4-1 lists the common ATE subcommands.

Table 4-1 The ATE program subcommands

Subcommand	Description
alter	Temporarily changes data transmission characteristics in the ATE program.
break	Interrupts current activity on a remote system.
connect	Connects to a remote computer.
directory	Displays the ATE dialing directory.
help	Provides help information for the ATE subcommands.
modify	Temporarily modifies local settings used for terminal emulation.
perform	Allows the user to issue workstation operating system commands while using ATE.
quit	Exits the Asynchronous Terminal Emulation (ATE) program.
receive	Receives a file from a remote system.
send	Sends a file to a remote system.
terminate	Terminates an ATE connection to a remote system.

4.1.9 EtherChannel

EtherChannel is a network aggregation technology that allows you to produce a single large pipe by combining the bandwidth of multiple Ethernet adapters. In AIX 5L Version 5.1, the EtherChannel feature has been enhanced to support the detection of interface failures. This is called network interface backup.

EtherChannel is a trademark registered by Cisco Systems and is generally called *multi-port trunking or link aggregation*. If your Ethernet switch device has this function, you can exploit the support provided in AIX 5L Version 5.1. In this case, you must configure your Ethernet switch to create a channel by aggregating a series of Ethernet ports.

Network interface backup mode

In the network interface backup mode, the channel will only activate one adapter at a time. The intention is that the adapters are plugged into different Ethernet switches, each of which is capable of getting to any other machine on the subnet/network. When a problem is detected, either with the direct connection, or through inability to ping a machine, the channel will deactivate the current adapter, and activate a backup adapter.

Note: The network interface backup feature is currently supported by 10/100 Ethernet and gigabit Ethernet PCI cards (devices.pci.23100020.rte and devices.pci.14100401.rte). If you are using other devices, you may receive unexpected results.

Configuring EtherChannel for network interface backup

Use SMIT either by choosing the SMIT fastpath EtherChannel or by clicking **Devices -> Communication -> EtherChannel**, as shown in Figure 4-6 on page 94.

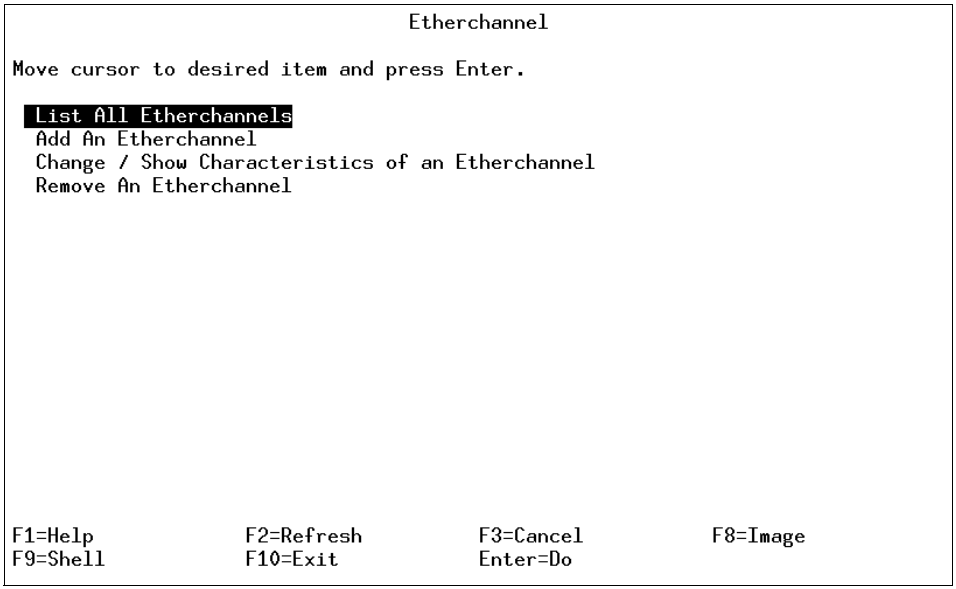


Figure 4-6 SMIT screen to add new EtherChannel

Choose **Add an EtherChannel** to add a new definition to your system, as shown in Figure 4-7.

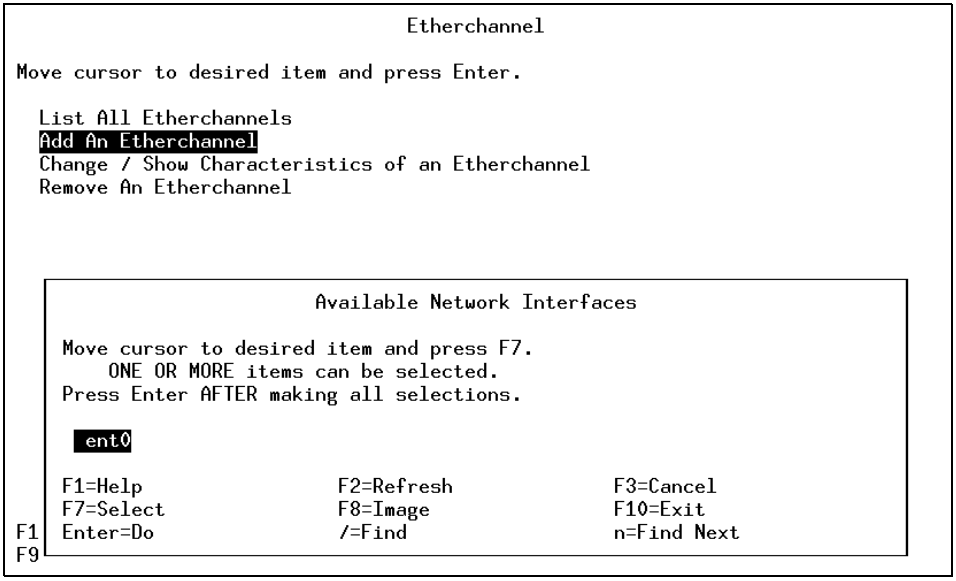


Figure 4-7 SMIT screen for choosing the adapters that belong to the channel

To create a new EtherChannel, you have to select the network interfaces that will be a part of the channel. If you select an interface that is in use or already part of another EtherChannel, you will receive an error similar to the following:

```
Method error (/usr/lib/methods/cfgech):
         0514-001 System error:
Method error (/usr/lib/methods/chgent):
         0514-062 can not perform the requested function because the
         specified device is busy.
```

```
                                Add An Etherchannel

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                        [Entry Fields]
  Etherchannel Adapters                                 ent0                 +
  Enable ALTERNATE ETHERCHANNEL address                 no                   +
  ALTERNATE ETHERCHANNEL address                       [0x1234deadbeef]      +
  Mode                                                  standard             +
  Enable GIGABIT ETHERNET JUMBO frames                  no                   +
  Internet Address to Ping                             [10.0.0.3]
  Number of Retries                                    []                    #
  Retry Timeout (sec)                                  []                    #

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure 4-8 SMIT screen for configuring the EtherChannel

Choose a valid alternate hardware address for the new EtherChannel, as shown in Figure 4-8. Change the EtherChannel mode to netif_backup to enable the network interface backup feature. In that mode, the channel will poll the adapter for Link Status. If the Link Status is not up (either due to a cable being unplugged, switch down, or device driver problem), the channel will switch to another adapter. This mode is the only one that makes use of the Internet Address to Ping, Number of Retries, and Retry Time-out fields. The following list provides the meaning of the fields:

- Internet Address to Ping

  The address will be pinged if the address field has a non-zero address and the mode is set to netif_backup. If the channel is unable to ping the address for the Number of Retries times in Retry Time-out intervals, the channel will switch adapters.

- Number of Retries

  The number of retries is the number of ping response failures before the channel switches adapters. The default is three times.

- Retry Timeout

  The retry timeout is the interval in seconds between the times when the channel will send out a ping packet and poll the adapter's Link Status. The default is one-second intervals.

  Once the EtherChannel has been configured, the new adapter and interfaces are made available.

Configuring IP on the EtherChannel interface

The new interface can be configured as any other network interface. Use SMIT to define an IP address on the interface:

```
# ifconfig en0
en0:
flags=e080863<UP,BROADCAST,NOTRAILERS,RUNNING,SIMPLEX,MULTICAST,GROUPRT,64
BIT>
       inet 10.0.0.4 netmask 0xffffff00 broadcast 10.0.0.255
```

Use the **ping** command to test the new IP connection:

```
# ping 10.0.0.3
PING 10.0.0.3: (10.0.0.3): 56 data bytes
64 bytes from 10.0.0.3: icmp_seq=0 ttl=255 time=0 ms
64 bytes from 10.0.0.3: icmp_seq=1 ttl=255 time=0 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=255 time=0 ms
```

4.2 Configuring network attributes

The **no** command is used to configure network attributes. The **no** command sets or displays current network attributes in the kernel. This command only operates on the currently running kernel. The command must be run again after each startup or after the network has been configured. Whether the command sets or displays an attribute is determined by the accompanying flag. The -o flag performs both actions. It can either display the value of an attribute or set a new value for an attribute. When the **no** command is used to modify a network option, it will log a message to the syslog using the LOG_KERN facility.

Be careful when using this command. The **no** command performs no range checking; therefore it accepts all values for the variables. If used incorrectly, the **no** command can cause your system to become inoperable.

Some network attributes are runtime attributes that can be changed at any time. Others are loadtime attributes that must be set before the netinet kernel extension is loaded and must be placed near the top of /etc/rc.net file.

The following section shows some examples of using network attribute configuration.

To change the maximum size of the mbuf pool to 3MB, enter:

```
# no -o thewall=3072
```

To change the default socket buffer sizes on your system, add the following lines to the end of the /etc/rc.net file:

```
/usr/sbin/no -o tcp_sendspace=16384
/usr/sbin/no -o udp_recvspace=16384
```

Table 4-2 is a partial listing of configurable network attributes.

Table 4-2 Configurable network attributes

Attribute	Description
directed_broadcast	Specifies whether or not to allow a directed broadcast to a gateway.
ipforwarding	Specifies whether the kernel should forward packets.
thewall	Specifies the maximum amount of memory, in kilobytes, that is allocated to the memory pool.
ipsendredirects	Specifies whether the kernel should send redirect signals.
net_malloc_police	Specifies the size of the net_malloc/net_free trace buffer. This includes checks for freeing a buffer, alignment and buffer overwrite.
route_expire	Specifies whether the route expires.
routervalidate	Specifies that each connection's cached route should be revalidated each time a new route is added to the routing table.
sb_max	Specifies the maximum buffer size allowed for a socket.
tcp_sendspace	Specifies the system default socket buffer size for sending data.
udp_recvspace	Specifies the system default socket buffer size for receiving UDP data.

4.3 Securing network services

This section provides a basic understanding about several standard network services on AIX, including:

- The r-commands
- The telnet service
- The ftp service

Although, these services can be configured with the Kerberos authentication method for additional security, using separate purchaseable software products on AIX. Other additional auditing packages include Computer Oracle and Password System (COPS), Security Administration Tool for Analyzing Networks (SATAN), and Security Administrator's Integrated Network Tool (SAINT).

Kerberos

Kerberos is a network authentication protocol. It is designed to provide strong authentication for client/server applications by using secret-key cryptography. Some sites attempt to use firewalls to solve their network security problems. Unfortunately, firewalls assume that intruders are on the outside, which is often an incorrect assumption. Several damaging incidents of computer crime are carried out by insiders. Firewalls also have a significant disadvantage in that they restrict how your users can use the Internet. Kerberos was created as a solution to these network security problems. The Kerberos protocol uses strong cryptography so that a client can prove its identity to a server (and vice versa) across an insecure network connection. After a client and server have used Kerberos to prove their identity, they can also encrypt all of their communications to assure privacy and data integrity as they go about their business.

Computer Oracle and Password System

Computer Oracle and Password System (COPS) is a collection of security tools that are designed specifically to aid the typical UNIX systems administrator, programmer, operator, or consultant in the often neglected area of computer security. The package can be broken down into three key parts. The first is the actual set of programs that attempt to automate security checks that are often performed manually (or perhaps with user-written short shell scripts or programs) by a system administrator. The second part is the documentation, which details how to set up, operate, and interpret any results given by the programs. The third part includes a list of possible extensions that might appear in future releases. COPS is a collection of programs that each attempt to tackle a different problem area of UNIX security. A few examples are listed below:

- File, directory, and device permissions/modes.
- Poor passwords.

- Content, format, and security of password and group files.
- The programs and files run in /etc/rc* and cron(tab) files.
- Existence of root-SUID files, their writeability, and whether or not they are shell scripts.
- A CRC check against important binaries or key files to report any changes therein.
- Writability of users' home directories and startup files (.profile, .cshrc, for example).
- Anonymous FTP setup.

Security Administration Tool for Analyzing Networks (SATAN)

SATAN is an older tool that was written to help systems administrators. It recognizes several common networking-related security problems, and reports the problems without actually exploiting them. For each type or problem found, SATAN offers a tutorial that explains the problem and what its impact could be. The tutorial also explains what can be done about the problem: correct an error in a configuration file, install a fix from the vendor, use other means to restrict access, or simply disable service. A few examples are listed below:

- NFS file systems exported to arbitrary hosts
- NFS file systems exported to unprivileged programs
- NFS file systems exported via the portmapper
- NIS password file access from arbitrary hosts
- Old (prior to 8.6.10) Sendmail versions
- REXD access from arbitrary hosts
- X server access control disabled
- Arbitrary files accessible via TFTP
- Remote shell access from arbitrary hosts
- Writable anonymous FTP home directory

Security Administrator's Integrated Network Tool (SAINT)

SAINT is the Security Administrator's Integrated Network Tool. In its simplest mode, it gathers as much information about remote hosts and networks as possible by examining such network services as finger, NFS, NIS, ftp and tftp, rexd, statd, and other services. The information gathered includes the presence of various network information services as well as potential security flaws, usually in the form of incorrectly set-up or configured network services, well-known bugs in system or network utilities, or poor or policy decisions. It can then either report on this data or use a simple rule-based system to investigate any potential security problems. Users can then examine, query, and analyze the output with an HTML browser, such as Mosaic, Netscape, or Lynx. While the program is primarily designed to analyze the security implications of the results,

a great deal of general network information can be gained when using the tool, such as network topology, network services running, types of hardware, and software being used on the network.

When run in exploratory mode, SAINT can be very beneficial. Based on the initial data collection and a user-configurable ruleset, it will examine the avenues of trust and dependency and iterate further data collection runs over secondary hosts. This not only allows you to analyze your own network or hosts, but also to examine the real implications inherent in network trust and services and help them make reasonably educated decisions about the security level of the systems involved.

SAINT has a target acquisition program that normally uses **fping** to determine whether or not a host or set of hosts in a subnet are alive. When a host is behind a firewall, however, tcp_scan is used to probe common ports to test for an alive host. It then passes this target list to an engine that drives the data collection and the main feedback loop. Each host is examined to see if it has been seen before, and, if not, a list of tests and probes is run against it (the set of tests depends on the distance the host is from the initial target and what probe level has been set.) The tests emit a data record that has the host name, the test run, and any results found from the probe; this data is saved in files for analysis. The user interface uses HTML to link the often vast amounts of data to more coherent and palatable results that the user can readily digest and understand.

Note: We have purposely excluded these advanced configurations from this redbook in order to simply explain the basic concept of these services.

4.3.1 The r-commands

The r-commands, where r stands for remote, is a so-called generic name for the **rcp** , **rlogin** and **rsh** commands. Table 4-3 lists the remote commands and describes their purpose:

Table 4-3 The r-commands

R-command	Description
rcp	Transfers files between a local and a remote host, or between two remote hosts.
rlogin	Connects a local hosts with a remote host.
rsh	Executes the specified command at the remote host or logs into the remote host.

These commands are installed in the /usr/bin directory and included in the bos.net.tcp.clients fileset, as shown in the following example:

```
# ls -l /usr/bin/rcp /usr/bin/remsh /usr/bin/rlogin /usr/bin/rsh
-r-sr-xr-x   1 root     system       319972 Feb 10 2002  /usr/bin/rcp
-r-sr-xr-x   2 root     system       303506 Feb 10 2002  /usr/bin/remsh
-r-sr-xr-x   1 root     bin          306328 Feb 10 2002  /usr/bin/rlogin
-r-sr-xr-x   2 root     system       303506 Feb 10 2002  /usr/bin/rsh
# lslpp -w /usr/bin/rcp /usr/bin/remsh /usr/bin/rlogin /usr/bin/rsh
  File                                        Fileset               Type
  ----------------------------------------------------------------------------
  /usr/bin/rcp                                bos.net.tcp.client    File
  /usr/bin/remsh                              bos.net.tcp.client    Hardlink
  /usr/bin/rlogin                             bos.net.tcp.client    File
  /usr/bin/rsh                                bos.net.tcp.client    File
```

As an example of the usage of these commands, Figure 4-9 illustrates the basic execution process flow of the **rsh** command.

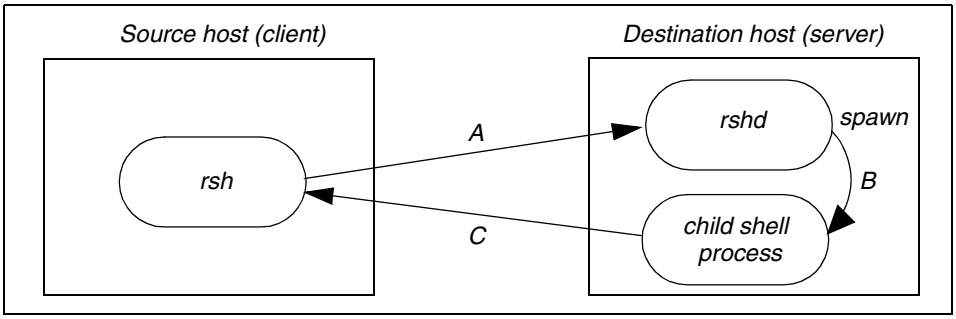


Figure 4-9 Execution process flow of rsh

The process flow is explained in the following list:

1. On the source host (client), the **rsh** command is invoked to connect to the destination host (server), as shown in A.
2. The rshd daemon attempts to validate the specified user using the following steps:
   a. The rshd daemon looks up the configured name service to be used for the user name and password, for example, the /etc/password file or NIS password map.
   b. If the user ID is not 0, rshd searches the /etc/hosts.equiv file to verify that the client name is listed; then the rshd daemon validates the user.
   c. If the $HOME/.rhosts file exists, rshd tries to authenticate the user by checking the $HOME/.rhosts file.

d. If either of the previous attempts failed, rshd shows you a password prompt of the user for the authentication.

3. Once rshd validates the user, it spawns a child shell of the user, as shown in B. The shell inherits the network connections established by the rshd daemon and it passes the command specified on the **rsh** command line. The shell sends the output to the client using the inherited network connection, as shown in C. When the remote command terminates, the local **rsh** process exits.

The /etc/host.equiv file is a system-wide configuration file for r-commands. The $HOME/.rhosts file is a user-basis configuration file for each user. To facilitate r-commands service, both files must reside on the server and have the same format as follows:

```
hostname [username]
```

The first field of this format expresses the host name, which is allowed to access the server. If a special character '+' is specified, any host is allowed to access the server. The optional second field expresses the user name to which access to the server is granted. If a special character '+' is specified, any user is granted the access; in other words, no authentication is attempted.

Although the usage of r-commands is quite simple and convenient, it may provide a security hole in your system. We strongly recommend you disable these services on your system.

The white paper *rlogin(1): The Untold Story*, provided by CERT, gives detailed information about security vulnerability for r-commands. It can be found at:

http://www.cert.org/archive/pdf/rlogin1_98tr017.pdf

Note: The r-commands service transmits all the data between client and server in clear text.

4.3.2 The telnet service

The telnet service is based on a client/server architecture, as follows:

telnet The **telnet** command, installed as /usr/bin/telnet, is an application client that supports the telnet service.

telnetd The telnetd daemon, installed as /usr/sbin/telnetd, is a server daemon process that supports the telnet service. The telnetd daemon process listens at port 23 by default, as specified in the /etc/services file. The telnetd daemon is invoked from inetd (an Internet super daemon process) upon receiving the connection request to the telnet service.

Note: The telnet service transmits all the data between client and server in clear text.

4.3.3 The FTP service

The ftp service is based on a client/server architecture, as follows:

ftp The **ftp** command, installed as /usr/bin/ftp, is an application client that supports the ftp service to be used for transferring files between a local and a remote host.

ftpd The ftpd daemon, installed as /usr/sbin/ftpd, is a server daemon process that supports the ftp service. The ftpd daemon is invoked from inetd upon receiving the connection request to the ftp service.

Note: The ftp service transmits all the data between client and server using a non-encrypted format.

4.4 Command summary

The following section provides a list of the key commands discussed in this chapter. For a complete reference of the following commands, consult the AIX product documentation.

4.4.1 The lsattr command

The **lsattr** command displays attribute characteristics and possible values of attributes for devices in the system. The command has the following syntax:

```
lsattr { -D [ -O ] | -E [ -O ] | -F Format } -l Name [ -a Attribute ] ...
[ -f File ] [ -h ] [ -H ]

lsattr { -D [ -O ] | -F Format } { [ -c Class ] [ -s Subclass ] [ -t Type ] }
[ -a Attribute ] ... [ -f File ] [ -h ] [ -H ]

lsattr -R { -l Name | [ -c Class ] [ -s Subclass ] [ -t Type ] } -a Attribute
[ -f File ] [ -h ] [ -H ]
```

The commonly used flags are provided in Table 4-4 on page 104.

Table 4-4 Commonly used flags of the lsattr command

Flag	Description
-a *Attribute*	Displays information for the specified attributes of a specific device or kind of device. You can use one -a flag for each attribute name or multiple attribute names. If you use one -a flag for multiple attribute names, the list of attribute names must be enclosed in quotes with spaces between the names. Using the -R flag, you must specify only one -a flag with only one attribute name. If you do not specify either the -a or -R flag, the **lsattr** command displays all information for all attributes of the specified device.
-E	Displays the attribute names, current values, descriptions, and user-settable flag values for a specific device when not used with the -O flag. The -E flag displays only the attribute name and current value in colon format when used with the -O flag. This flag cannot be used with the -c, -D, -F, -R, -s, or -t flag.
-l *Name*	Specifies the device logical name in the Customized Devices object class whose attribute names or values are to be displayed.

4.4.2 The chdev command

The **chdev** command changes the characteristics of a device. The command has the following syntax:

```
chdev -l Name [ -a Attribute=Value ... ] [ -f File ] [ -h ] [ -p ParentName ]
[-P | -T ] [ -q ] [ -w ConnectionLocation ]
```

The commonly used flags are provided in Table 4-5.

Table 4-5 Commonly used flags of the chdev command

Flag	Description
-l device	The name of the device that is being changed.
-a	The device attribute and the new value. Use **lsattr** to see the attributes that can be changed.

4.5 Quiz

The following assessment questions help verify your understanding of the topics discussed in this chapter.

1. All of the following are times when the minimum configuration smit screen for TCP/IP should be used *except*:

   A. When setting a default route

   B. When reconfiguring TCP/IP from scratch

   C. When changing the IP address of one adapter in the system

   D. When configuring the first adapter in a newly installed machine

2. Scenario: A network administrator has been asked to integrate a new RS/6000 to be used as a corporate mail server into the network. There are five nodes on the Ethernet II network, with a network address of 193.3.7.0 and a subnet mask of 255.255.255.0. The machine contains ATM, token-ring and integrated Ethernet adapters.

   Which one of the following files should be modified in order to enable this node to use DNS for host name resolution?

   A. /etc/hosts

   B. /etc/inetd.conf

   C. /etc/resolv.conf

   D. /etc/named.boot

3. Scenario: A network administrator has been asked to integrate a new RS/6000 to be used as a corporate mail server into the network. There are five nodes on the Ethernet II network, with a network address of 193.3.7.0 and a subnet mask of 255.255.255.0. The machine contains ATM, token-ring and integrated Ethernet adapters.

   The Internet Service Provider has set up a gateway for the administrator to access the Internet. The IP address of this gateway is 193.3.7.99. Which one of the following actions must occur for this new machine to reach the Internet?

   A. Create a network route to 193.3.7.99 for 0.0.0.0

   B. Assign 193.3.7.99 as an alias to the Ethernet adapter

   C. Add the address 193.3.7.99 to the /etc/resolv.conf file

   D. Use **no** to set the ipforwarding attribute to 193.3.7.99

4. If the local machine is configured as a primary name server, which one of the following statements is true?
   A. /etc/resolv.conf does not exist
   B. /etc/resolv.conf must be an empty file
   C. /etc/resolv.conf contains the local loopback address
   D. /etc/resolv.conf is either an empty file or contains “nameserver 127.0.0.1”
5. Which one of the following will result from an adapter configuration of **`ifconfig en0 129.35.22.8 network 255.0.0.0`**?
   A. Destination Gateway 129.0.0.99 129.35.22.8
   B. Destination Gateway 129.255.0.1 129.35.22.8
   C. Destination Gateway 224.0.0.1 129.35.22.8
   D. Destination Gateway 254.0.0.1 129.35.22.8
6. Which one of the following procedures must be performed to add multiple nameservers or multiple domains when searching for a DNS lookup?
   A. Use a smit panel.
   B. Edit the /etc/resolv.conf.
   C. Start the named daemon.
   D. Refresh the named daemon.
7. All of the following are used to audit or evaluate system security *except*:
   A. COPS
   B. Kerberos
   C. SATAN
   D. SAINT
8. Which one of the following statements best describes Kerberos?
   A. Kerberos is a system designed to provide host security.
   B. Kerberos is a password encryption system that replaces login in a Trusted Computing Base.
   C. Kerberos is a network authentication protocol.
   D. Kerberos is a public-key cryptography system used in AIX IPSec.

9. Which one of the following commands does *not* definitively show whether there is an Asynchronous Terminal installed on a system?

   A. **lscfg -vp**

   B. **lsattr -El tty0**

   C. **lsdev -C**

   D. **lscons**

10. All of the following will show all TTY ports *except*?

   A. **lscfg -vp**

   B. **lscons**

   C. **lsdev -C**

   D. **lsattr -El tty0**

11. Which command is used to adjust tcp_sendspace?

   A. **netstat**

   B. **ifconfig**

   C. **no**

   D. **chdev**

12. Which command is used to modify network options?

   A. **no -o**

   B. **ifconfig -a**

   C. **chdev -l**

   D. **netstat -m**

13. After installing and configuring a modem for a remote console tty0 on the S1 port, the system is restarted, an error code CF1 is received, and a prompt to identify the console is displayed. Which one of the following options is the best recovery solution?

   A. **pdisable**

   B. **chcons -a login=enable /dev/tty0**

   C. **chdev -l tty0 -a login=enable**

   D. **swcons tty0**

4.5.1 Answers

The following are the preferred answers to the questions provided in this section:

1. C
2. C
3. A
4. D
5. A
6. B
7. B
8. C
9. D
10. B
11. C
12. A
13. C

4.6 Exercises

The following exercises provide sample topics for self study. They will help ensure comprehension of this chapter.

1. Determine the location of your network adapters.
2. On a test system, change the subnet mask of en0 using the **chdev** command.

5

Network daemons

This chapter discusses the following topics:

- TCP/IP network startup
- Network daemons
- Network services, specifically BOOTP and DHCP
- General network configuration and the tools provided
- Administration of network adapters and interfaces

Several common services that a system administrator has to manage are discussed in this chapter.

5.1 Network startup

When the system is powered on, the network startup is initiated by **cfgmgr** as part of the second boot phase. The network startup script that starts the network is determined by the ODM configuration rules. The AIX default is /etc/rc.net script, which uses the ODM data to define, load, and configure network interfaces.

Figure 5-1 illustrates the complete startup process.

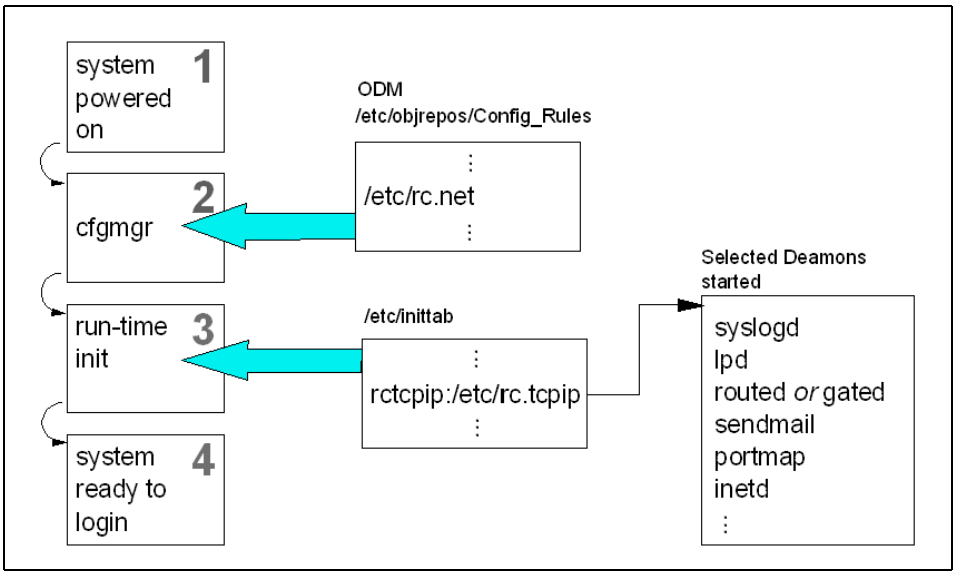


Figure 5-1 TCP/IP network startup procedure

Another possible network startup is the BSD-style network configuration using /etc/rc.bsdnet. This script uses the traditional **ifconfig** command to configure the networking interface.

The next phase of networking startup is running the /etc/rc.tcpip script that is started by the init program. At network installation time, an entry is made in the /etc/inittab automatically inserting the rc.tcpip script. The rc.tcpip script starts selected network daemons using the System Resource Controller (SRC).

In AIX, the name subsystem and subserver have specific meanings:

subsystem A daemon or server that is controlled by SRC.

subserver A daemon that is controlled by a subsystem. Since the only TCP/IP subsystem that controls subservers is inetd, all TCP/IP daemons controlled by inetd are subservers.

5.1.1 System Resource Controller

The SRC is an AIX-specific subsystem controller used to manage and control subsystem processes (also known as server daemons). The SRC helps system administrators control system server processes/daemons by providing utilities for start, stop, trace, list, and refresh of daemons.

The following utilities are provided for managing the SRC:

`startsrc`	Starts the TCP/IP subsystems and TCP/IP subservers.
`stopsrc`	Stops all TCP/IP subsystems and TCP/IP subservers.
`refresh`	Refreshes the subsystems and subservers (that is, it forces the re-initialization).
`lssrc`	Provides the status of subsystems and subservers.

For more information on the SRC, refer to the *IBM @server Certification Study Guide - pSeries AIX System Administration*, SG24-6191.

5.2 Network subsystems

The /etc/rc.tcpip file is a shell script that, when executed on system bootup, uses **`startsrc`** to start up selected daemons. The rc.tcpip script can also be executed at any time from the command line.

The following TCP/IP subsystems, listed in file order, can be started with rc.tcpip:

syslogd	Log server for standard UNIX error logs.
portmap	Port lookup facility used for remote procedure call (RPC).
inetd	Internet daemon that starts other services such as telnet or ftp.
named	Domain name server in a domain network.
lpd	Print server daemon.
routed or **gated**	Dynamic routing. Note that you cannot have both running simultaneously.
sendmail	Mail transfer agent.
timed, **xntpd**	Time synchronization daemons.
rwhod	Remote uptime and users.
snmpd, **dpid2**	SNMP daemons.
dhcpcd, **dhcprd**, **dhrcpsd**	DHCP daemons.

autoconf6, **ndpd-host**	IPv6 daemons.
mrouted	Multicast routing.

Note: The rc.tcpip starts syslogd, portmap, inetd, lpd, and sendmail daemons automatically. All the other daemons listed above must be uncommented in rc.tcpip. This is usually done when the network daemons are individually configured.

To list of all network daemons, use the **lssrc** command:

```
# lssrc -g tcpip
Subsystem         Group            PID     Status
 inetd            tcpip            7484    active
 snmpd            tcpip            7740    active
 dpid2            tcpip            7998    active
 tftpd            tcpip            14494   active
 rwhod            tcpip            15466   active
 gated            tcpip                    inoperative
 named            tcpip                    inoperative
 routed           tcpip                    inoperative
 iptrace          tcpip                    inoperative
 xntpd            tcpip                    inoperative
 timed            tcpip                    inoperative
 dhcpcd           tcpip                    inoperative
 dhcpsd           tcpip                    inoperative
 dhcprd           tcpip                    inoperative
 ndpd-host        tcpip                    inoperative
 ndpd-router      tcpip                    inoperative
 mrouted          tcpip                    inoperative
```

This lists all the server daemons in the group tcpip. Alternatively, for controlling TCP/IP subsystems you can use **smitty subsys**, as shown in Figure 5-2 on page 113.

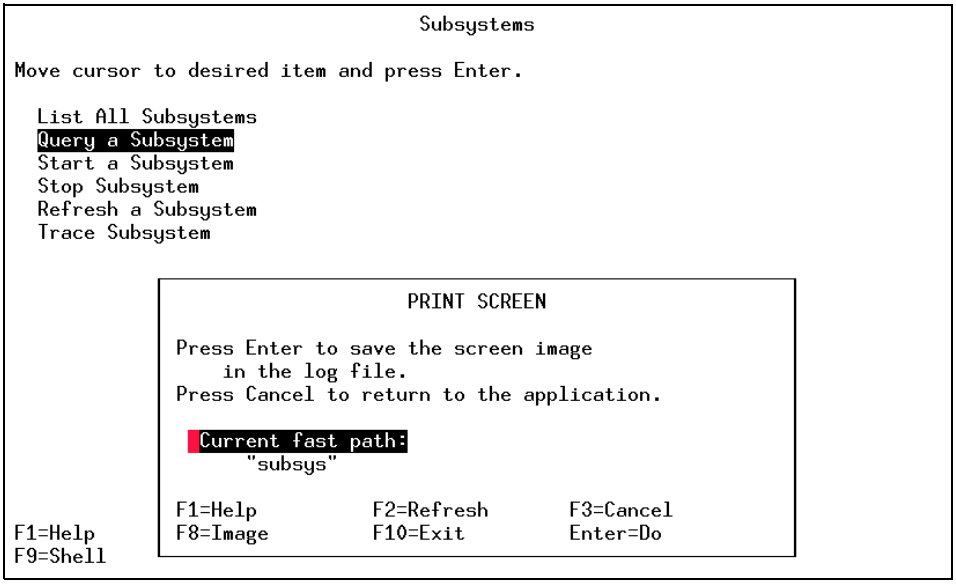


Figure 5-2 SMIT screen for controlling SRC subsystems

These SMIT menus assist you in using the SRC features such as status, starting, stopping, refreshing, and tracing the subsystem server daemons.

5.3 Stopping network subsystems

All TCP/IP subsystems started with rc.tcpip can be stopped with the SRC **stopsrc** command. The subsystems can be stopped individually using the -s flag.

```
# stopsrc -s dhcpsd
0513-044 The dhcpsd Subsystem was requested to stop.
```

Or the subsystems can be stopped collectively using the TCP/IP group -g flag for **stopsrc**:

```
# stopsrc -g tcpip
```

Note: Use this command at the system console only.

Additionally, for convenience, the script /etc/tcp.clean can be used for stopping the daemons.

5.4 Internet daemon - inetd

The Internet daemon inetd is the *super* server daemon that manages the other Internet subservers and starts up the other server daemons upon request. The inetd both simplifies the management and reduces system load by invoking other daemons only when they are needed. The inetd is started from the rc.tcpip script using the SRC. At startup, the inetd reads its configuration file /etc/inetd.conf, which specifies what Internet services to provide on the system. The inetd will listen to each port that the corresponding Internet service is using, for example, telnet (port 23). If a client request is made on the specific port, inetd starts up the program specified in the inetd.conf, which in the example of telnet is the telnetd daemon.

5.4.1 The /etc/inetd.conf file

The /etc/inetd.conf file is the default configuration file for the inetd daemon. This file enables you to specify which daemons to start by default and supply the arguments that correspond to the desired style of functioning for each daemon. If you change the /etc/inetd.conf file, run the **`refresh -s inetd`** or **`kill -1 InetdPID`** command to inform the inetd daemon of the changes to its configuration file.

The inetd configuration file located in /etc/inetd.conf is a simple ASCII file containing an entry for each supported Internet service. Each entry consists of:

ServiceName The name of the Internet service as it is listed in /etc/services. The name must be identical to the first entry of the /etc/services line that matches the name.

SocketType Contains the name for the type of socket used for the service.
stream - specifies a stream socket.
dgram - specifies a datagram socket.
sunrpc_tcp - specifies a RPC stream socket.
sunrpc_udp - specifies a RPC datagram socket.

ProtocolName The name of the Internet protocol used by the service as defined in the /etc/protocols file.
tcp - specifies TCP/IP protocol.
udp - specifies the UDP protocol.

wait/nowait/SRC Wait is for dgram, nowait is for stream. Determines whether inetd waits for a datagram server to release the socket before continuing listening to the socket. The SRC instruction works like wait, but uses **`startsrc`** on the subsystem and stores information about the starting of the service.

User Name Specifies the user name the inetd starts the server with. This allows control of the permissions of the server process.

Server Path Full path to the server program. For services that the inetd daemon provides internally, this field should be internal.

Program Arguments Optional command-line arguments the server program is started with. The maximum number of arguments is five.

The following shows an extract from the /etc/inetd.conf file:

```
## service   socket    protocol  wait/     user   server          server program
## name      type                nowait           program             arguments
##
   ftp       stream    tcp6      nowait    root   /usr/sbin/ftpd      ftpd
   telnet    stream    tcp6      nowait    root   /usr/sbin/telnetd   telnetd -a
   shell     stream    tcp6      nowait    root   /usr/sbin/rshd      rshd
   kshell    stream    tcp       nowait    root   /usr/sbin/krshd     krshd
   login     stream    tcp6      nowait    root   /usr/sbin/rlogind   rlogind

klogin    stream    tcp    nowait    root   /usr/sbin/krlogind     krlogind
exec      stream    tcp6   nowait    root   /usr/sbin/rexecd       rexecd
#comsat   dgram     udp    wait      root   /usr/sbin/comsat       comsat
#uucp     stream    tcp    nowait    root   /usr/sbin/uucpd        uucpd
#bootps   dgram     udp    wait      root   /usr/sbin/bootpd       bootpd
/etc/bootp tab
....
```

The following is a list of the Internet subservers supported by inetd on a basic AIX installation. A starting hash (#) sign indicates that these subservers by default are not configured (commented out) in /etc/inetd.conf.

The following daemons are controlled by the inetd daemon:

bootpd The bootpd daemon is the server that receives all bootp requests. It uses the /etc/bootptab file to read its configuration information.

comsat The comsat daemon is the server that receives reports of incoming mail and notifies users if they have enabled this service with the **biff** command.

ftpd The /usr/sbin/ftpd daemon is the DARPA Internet File Transfer Protocol (FTP) server process. The ftpd daemon uses the TCP to listen at the port specified with the **ftp** command service specification in the /etc/services file.

telnetd The /usr/sbin/telnetd daemon is a server that supports the Defense Advanced Research Product Agency (DARPA) standard Telnet Protocol (TELNET). When a **telnet** session is

started, the telnetd daemon sends TELNET options to the client (remote) host to indicate an ability to perform options.

rshd The /usr/sbin/rshd daemon is the server for the **rcp** and **rsh** commands. The rshd daemon provides remote execution of shell commands. These commands are based on requests from privileged sockets on trusted hosts.

rlogind The /usr/sbin/rlogind daemon is the server for the **rlogin** remote login command. The server provides a remote login facility.

rexecd The /usr/sbin/rexecd daemon is the server for the **rexec** command.

fingerd The /usr/sbin/fingerd daemon is a simple protocol that provides an interface to the **finger** command at several network sites.

tftpd The /usr/sbin/tftpd daemon runs the Trivial File Transfer Protocol (TFTP) server. Files sent using TFTP can be found in the directory specified by the full path name given on the **tftp** or **utftp** command line.

talkd The /usr/sbin/talkd daemon is the server that notifies a user (the recipient) that another user (the caller) wants to initiate a conversation. The daemon sets up the conversation if the recipient accepts the invitation. The caller initiates the conversation by executing the **talk** command specifying the recipient. The recipient accepts the invitation by executing the **talk** command specifying the caller.

uucpd The uucpd daemon is a subserver of the inetd daemon. The uucpd daemon must be running as a background process on all the networked systems before the BNU program can use TCP/IP system to communicate. If the uucpd daemon is not running, reconfigure the inetd daemon to start the uucpd daemon. Use the **netstat** command to find out if the uucpd daemon is running.

The ftpd, rlogind, rexecd, rshd, talkd, telnetd, and uucpd daemons are started by default. The tftpd, fingerd, and comsat daemons are not started by default unless they are uncommented in the /etc/inetd.conf file.

Additional software products might use the inetd features to start up their network services. Typically this is done by inserting entries in the /etc/inetd.conf; for example, WebSphere MQ places the listener daemon in care of the inetd daemon.

5.4.2 The /etc/services file

The /etc/services file contains information about the known services used in the DARPA Internet network as well as other entries that may be added by third-party vendors. Each service is listed on a single line corresponding to the form:

ServiceName PortNumber/ProtocolName Aliases

These fields contain the following information:

ServiceName Specifies an official Internet service name.

PortNumber Specifies the socket port number used for the service.

ProtocolName Specifies the transport protocol used for the service.

Aliases Specifies a list of unofficial service names.

Items on a line are separated by spaces or tabs. Comments begin with a # (pound sign) and continue until the end of the line.

If you edit the /etc/services file, run the **refresh -s inetd** or **kill -1** ***InetdPID*** command to inform the inetd daemon of the changes.

An example of the /etc/services file is as follows:

```
# Network services, Internet style
#
tcpmux          1/tcp                           # TCP Port Service Multiplexer
tcpmux          1/udp                           # TCP Port Service Multiplexer
compressnet     2/tcp                           # Management Utility
compressnet     2/udp                           # Management Utility
...
telnet          23/tcp
smtp            25/tcp          mail
nsw-fe          27/tcp                          # NSW User System FE
nsw-fe          27/udp                          # NSW User System E
...
man             9535/tcp
man             9535/udp
isode-dua       17007/tcp
isode-dua       17007/udp
dtspc           6112/tcp
fontserver      7100/tcp                xfs     # X11R6 font server
```

5.4.3 The ports assigned to network services

Table 5-1 provides a quick reference to some of the more common daemons that are controlled in /etc/inetd.conf or /etc/sendmail.cf and what they do.

Table 5-1 Command and port quick reference guide

Daemon	Port	Description
ftp	21	Transfers files between a local and a remote host.
tftp	69	Trivial File Transfer Protocol. Transfers files between hosts using minimal protocol.
login	513	The **rlogin** command connects the local terminal to the remote host specified by the HostName parameter.
telnet	23	Connects the local host with a remote host, using the Telnet interface.
bootps	67	Sets up the Internet Boot Protocol server.
timed	525	Time server daemon. Synchronizes clock with other machines running timed on the local area network.
shell	514	At login, the shell defines the user environment after reading the shell startup files.
snmp	161	SNMP is used by network hosts to exchange information in the management of networks.
smtp	25	A protocol, typically used over a network, in which the objective is to transfer mail. SMTP is used by the **sendmail** command to accept and receive mail.

Every network service is performed over a port. Below is a list of some of the more common ports and their respective network services as extracted from the /etc/services file:

```
ftp             21/tcp
telnet          23/tcp
shell           514/tcp         cmd             # no passwords used
kshell          544/tcp         krcmd
login           513/tcp
klogin          543/tcp
exec            512/tcp
uucp            540/tcp         uucpd           # uucp daemon
bootps          67/udp                          # bootp server port
finger          79/tcp
systat          11/tcp          users
netstat         15/tcp
tftp            69/udp
talk            517/udp
ntalk           518/udp
snmp            161/tcp                         # snmp request port
snmp            161/udp                         # snmp request port
snmp-trap       162/tcp                         # snmp monitor trap port
```

```
snmp-trap       162/udp                         # snmp monitor trap port
smtp            25/tcp          mail
re-mail-ck      50/tcp                          # Remote Mail Checking Protocol
re-mail-ck      50/udp                          # Remote Mail Checking Protocol
xns-mail        58/tcp                          # XNS Mail
xns-mail        58/udp                          # XNS Mail
ni-mail         61/tcp                          # NI MAIL
ni-mail         61/udp                          # NI MAIL
imap2           143/tcp                         # Interim Mail Access Pro. v2
imap2           143/udp                         # Interim Mail Access Pro. v2
pcmail-srv      158/tcp                         # PCMail Server
pcmail-srv      158/udp                         # PCMail Server
mailq           174/tcp                         # MAILQ
mailq           174/udp                         # MAILQ
tam             209/tcp                         # Trivial Auth. Mail Protocol
tam             209/udp                         # Trivial Auth. Mail Protocol
imap3           220/tcp                         # Interactive Mail Acces Pro.
imap3           220/udp                         # Interactive Mail Acces Pro.
mailbox         2004/tcp
```

The list is not exhaustive, but does show the main daemons needed for the networking environment.

5.4.4 Inetd subsystem control

Any change to the /etc/inetd.conf file requires a refresh of the inetd daemon in order to re-read the configuration and apply the change.

```
# refresh -s inetd
0513-095 The request for subsystem refresh was completed successfully.
```

An alternate way of controlling the inetd daemon and the /etc/inetd.conf is using the Web-based System Manager (wsm) menus for networking. Figure 5-3 on page 120 shows the wsm networking support window for controlling inetd.

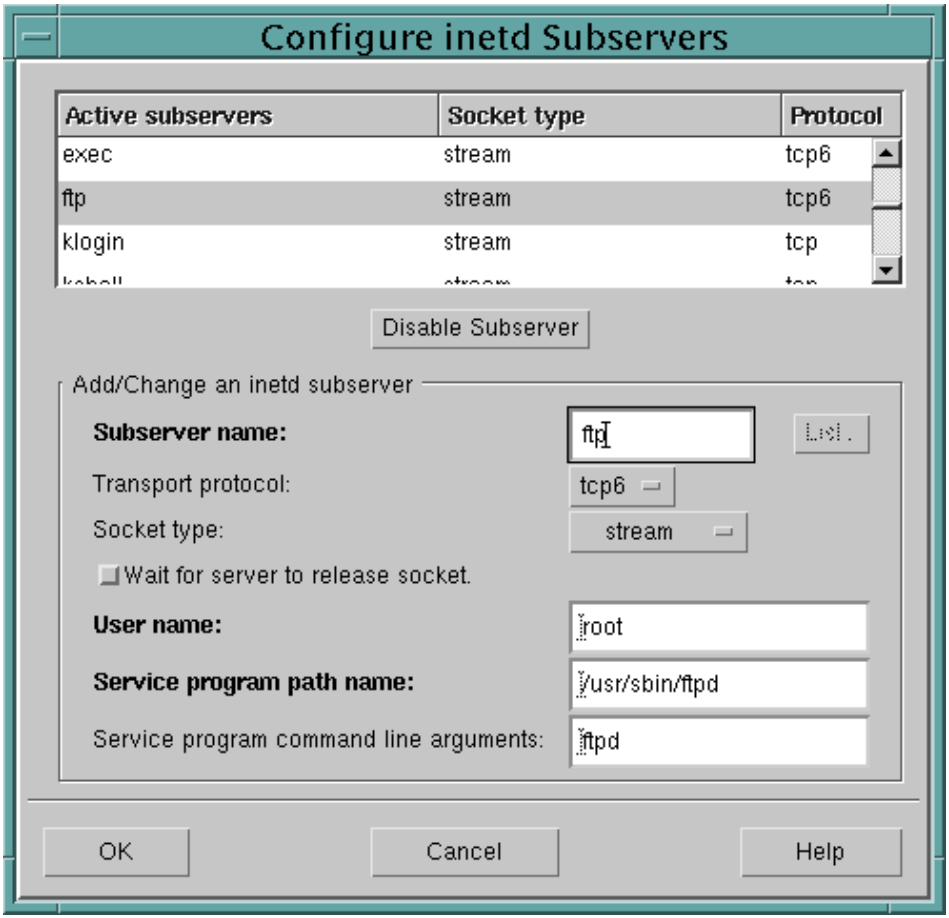


Figure 5-3 Inetd configuration support in wsm

5.5 Network subservers

The following section discusses the network subservers and how to perform basic administration on them.

5.5.1 Controlling subservers

The SRC can be used to activate the individual inetd subservers by using **startsrc** with the - t flag.

```
# startsrc -t time
0513-124 The time subserver has been started.
```

Alternatively, use the dedicated SMIT command **smitty subserver**.

Stopping individual inetd subservers can be done by using the **stopsrc -t** command.

For example:

```
# stopsrc -t ftp
0513-127 The ftp subserver was stopped successfully.
# hostname
server2
# ftp server2
ftp: connect: A remote host refused an attempted connect operation.
ftp> quit
# startsrc -t ftp
0513-124 The ftp subserver has been started.
# ftp server2
Connected to server2.itsc.austin.ibm.com.
220 server2 FTP server (Version 4.1 Fri Nov 19 18:18:48 CST 1999) ready.
Name (server2:root):
```

5.5.2 File Transfer Protocol (FTP)

The File Transfer Protocol (FTP) is used for copying files between machines using TCP. The FTP client logs into the other system with an FTP server (ftpd) and is authenticated with a user ID and password. After the login, the FTP client can perform a set of operations. The following are the most frequent operations:

prompt	Toggle interactive prompting.
cd	Select a directory.
lcd	Change the working directory on the local host. If you do not specify a directory, the **ftp** command uses your home directory.
ls, **dir**	List files available for transfer.
ascii, **binary**	Define the transfer type: ascii (default) sets the file-transfer type to network ASCII, and binary sets the file-transfer type to binary image. Must always be used when transferring programs.
get, **mget**	Copy file or files from the remote server.
put, **mput**	Copy file or files to the remote server.
help	List and help on all FTP commands.

The FTP server ftpd is an inetd subserver and it is by default activated in the /etc/inetd.conf configuration.

5.5.3 Anonymous FTP

Anonymous FTP allows public access to some file directories on your system. The remote user only needs to use the login name anonymous and password guest or some other common password conventions (typically the user's Internet e-mail ID).

To set up anonymous FTP on AIX, use the script /usr/samples/tcpip/anon.ftp.This will create the appropriate users and directories for using anonymous FTP. Setting up anonymous FTP will allow guest users to write binaries that could contain computer viruses if they are not carefully monitored.

5.5.4 RCP file transfer

The **rcp** command copies one or more files between a local host and a remote host, between two separate remote hosts, or between files at the same remote host. This command is similar to the **cp** command except that it works only for remote file operations and the attributes of a file are maintained. If extra security is needed for your network, this command may be disabled by the system administrator.

5.5.5 Trivial File Transfer Protocol

Trivial File Transfer Protocol (TFTP) is a simple protocol to transfer files implemented on top of UDP (User Datagram Protocol). TFTP is used, for example, by network stations to download boot images. TFTP is a small subset of FTP, providing only read/write of files from/to a server.

Note: TFTP has no means of user authentication and is considered an unsecure protocol.

The TFTP server tftpd is an inetd subserver, so /etc/inetd.conf must be configured to activate TFTP. The file /etc/tftpaccess.ctl file is used for configuring remote access to the directories on the system, by allowing (keyword allow) or denying (keyword deny) access to directories. A sample file is provided in /usr/samples/tcpip/tftpaccess.ctl.

5.5.6 Security consideration with inetd subservers

The following sections discuss various security considerations with regards to inetd subservers.

Note: Setting up services of FTP, remote login (rlogind), or remote execution (rexec) will have security implications for your system. In the following, configuration files for automatic access are discussed, but be aware of the possible danger of these configurations.

The $HOME/.netrc file

The $HOME/.netrc file contains information used by the automatic login feature of the **rexec** and **ftp** commands. It is a hidden file in a user's home directory and must be owned either by the user executing the command or by the root user. If the .netrc file contains a login password, the file's permissions must be set to 600 (read and write by owner only). The login password is in plain text. Even with permissions set to 600, passwords for remote systems are vulnerable to being revealed to any user with root authority.

The $HOME/.forward file

When mail is sent to a local user, the **sendmail** command checks for the $HOME/.forward file. The $HOME/.forward file can contain one or more addresses or aliases. If the file exists, the message is not sent to the user. The message is sent to the addresses or aliases in the $HOME/.forward file. All messages, including confidential ones, will never reach the user if this is implemented.

The /etc/hosts.equiv file

The /etc/hosts.equiv file, along with any local $HOME/.rhosts files, defines the hosts (computers on a network) and user accounts that can invoke remote commands on a local host without supplying a password. The $HOME/.rhosts file is similar to the /etc/hosts.equiv file, except that it is maintained for individual users.

The $HOME/.rhosts file

The $HOME/.rhosts file defines which remote hosts (computers on a network) can invoke certain commands on the local host without supplying a password. This file is a hidden file in the local user's home directory and must be owned by the local user. It is recommended that the permissions of the .rhosts file be set to 600 (read and write by owner only). Bypassing the need for a password may be a security concern, especially if you allow all users on a particular system access without needing a password.

The permissions and the entries in the $HOME/.rhosts file will affect whether a user on a remote host can successfully establish an rsh session. Both files, hosts.equiv and .rhosts, must have permissions denying write access to group and other. If either group or other have write access to a file, that file is ignored.

The securetcpip command

The **securetcpip** command provides enhanced security for the network. This command performs the following:

1. Runs the **tcbck -a** command, which disables the nontrusted commands and daemons: **rcp**, **rlogin**, **rlogind**, **rsh**, **rshd**, **tftp**, and **tftpd**. The disabled commands and daemons are not deleted; instead, they are changed to mode 0000. You can enable a particular command or daemon by re-establishing a valid mode.
2. Adds a TCP/IP security stanza to the /etc/security/config file. The stanza is in the following format:

```
tcpip: netrc = ftp,rexec     /* functions disabling netrc */
```

Before running the **securetcpip** command, quiesce the system by logging in as root user and executing the **killall** command to stop all network daemons.

Note: The **killall** command kills all processes except the calling process. If logged in or applications are running, exit or finish before executing the **killall** command.

After issuing the **securetcpip** command, shut down and restart your system. All of your TCP/IP commands and network interfaces should be properly configured after the system restarts.

Some examples are shown in Table 5-2.

Table 5-2 $HOME/.rhosts definitions

Local Host (sv1050a) User itsouser	Remote Host (aix4xdev) User itsouser
$ cat > $HOME/.rhosts aix4xdev $ chmod 600 $HOME/.rhosts $	$ rsh sv050a -l itsouser ls -a rshd: 0826-813 Permission is denied. $
$ cat > $HOME/.rhosts aix4xdev itsouser $ chmod 600 $HOME/.rhosts $	$ rsh sv050a -l itsouser ls -a . .. .profile .rhosts .sh_history $

Local Host (sv1050a) User itsouser	Remote Host (aix4xdev) User itsouser
$ cat > $HOME/.rhosts aix4xdev + $ chmod 600 $HOME/.rhosts $	$ rsh sv050a -l itsouser ls -a . .. .profile .rhosts .sh_history $
$ chmod 644 $HOME/.rhosts $	$ rsh sv050a -l itsouser ls -a . .. .profile .rhosts .sh_history $
$ chmod 666 $HOME/.rhosts $	$ rsh sv050a -l itsouser ls -a rshd: 0826-813 Permission is denied. $
$ chmod 777 $HOME/.rhosts $	$ rsh sv050a -l itsouser ls -a rshd: 0826-813 Permission is denied. $

5.6 Command summary

The following sections include descriptions of the key commands discussed in this chapter. For a complete reference of the following commands, consult the AIX product documentation.

5.6.1 The startsrc command

The **startsrc** command starts a subsystem, a group of subsystems, or a subserver. The command has the following syntax:

For subsystem:

```
startsrc [-a Argument] [-e Environment] [-h Host] {-s Subsystem |-g Group}
```

For subserver:

```
startsrc [-h Host] -t Type [-o Object] [-p SubsystemPID]
```

The commonly used flags are provided in Table 5-3 on page 126.

Table 5-3 Commonly used flags of the startsrc command

Flag	Description
-s Subsystem	Specifies a subsystem to be started. The Subsystem variable can be the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the subsystem is not contained in the subsystem object class.
-t Type	Specifies that a subserver is to be started. The command is unsuccessful if the Type variable is not contained in the subserver object class.

5.6.2 The stopsrc command

The **stopsrc** command stops a subsystem, a group of subsystems, or a subserver. The command has the following syntax:

For Subsystem:

```
stopsrc [-h Host] [-f | -c] {-a | -g Group | -p SubsystemPID | -s Subsystem }
```

For Subserver:

```
stopsrc [-h Host] [-f] -t Type [-p SubsystemPID] [-P SubserverPID | -o Object]
```

The commonly used flags are provided in Table 5-4.

Table 5-4 Commonly used flags of the stopsrc command

Flag	Description
-g Group	Specifies that a group of subservers are to be stopped. The command is unsuccessful if the Group name is not contained in the subsystem object class.
-s Subsystem	Specifies a subsystem to be stopped. The Subsystem parameter can be the actual subsystem name or the synonym name for the subsystem. The **stopsrc** command stops all currently active instances of the subsystem. The command is unsuccessful if the subsystem name is not contained in the subsystem object class.
-t Type	Specifies that a subserver is to be stopped. The **stopsrc** command is unsuccessful if the Type specified is not contained in the subserver object class.

5.6.3 The refresh command

The **refresh** command requests a refresh of a subsystem or group of subsystems. The command has the following syntax:

refresh [-h Host] {-g Group|-p SubsystemPID|-s Subsystem}

The commonly used flags are provided in Table 5-5.

Table 5-5 Commonly used flags of the refresh command

Flag	Description
-g Group	Specifies a group of subsystems to refresh. The **refresh** command is unsuccessful if the group name is not contained in the subsystem object class.
-s Subsystem	Specifies a subsystem to refresh. The Subsystem name can be the actual subsystem name or the synonym name for the subsystem. The **refresh** command is unsuccessful if subsystem name is not contained in the subsystem object class.

5.6.4 The lssrc command

The **lssrc** command gets the status of a subsystem, a group of subsystems, or a subserver. The command has the following syntax:

Subsystem status:

```
lssrc [ -h Host ] { -a | -g GroupName | [ -l ] -s Subsystem | [ -l ] -p
SubsystemPID }
```

Subserver status:

```
lssrc [ -h Host ] [ -l ] -t Type [ -p SubsystemPID ] [ -o Object ] [ -P
SubserverPID ]
```

The commonly used flags are provided in Table 5-6.

Table 5-6 Commonly used flags of the lssrc command

Flag	Description
-a	Lists the current status of all defined subsystems.
-g Group	Specifies a group of subsystems to get status for. The command is unsuccessful if the GroupName variable is not contained in the subsystem object class.
-s Subsystem	Specifies a subsystem to get status for. The Subsystem variable can be the actual subsystem name or the synonym name for the subsystem. The command is unsuccessful if the Subsystem variable is not contained in the subsystem object class.

Flag	Description
-t Type	Requests that a subsystem send the current status of a subserver. The command is unsuccessful if the subserver Type variable is not contained in the subserver object class.

5.7 Quiz

The following assessment questions help verify your understanding of the topics discussed in this chapter.

1. Which one of the following methods will allow a file to be copied from a remote host and retain the attributes of the file?

   A. **ftp**

   B. **rcp**

   C. **tcopy**

   D. No protocol provides such function

2. After uncommenting a line in /etc/inetd.conf to enable tftpd, which one of the following will allow a remote machine to access a file through **tftp**?

   A. **/usr/sbin/tftpd**

   B. **refresh -s inetd**

   C. Uncomment the line for tftpd in /etc/services

   D. No action is required

3. Since the use of /etc/hosts.equiv or ~/.rhosts allows remote access without using a password, which one of the following procedures is most appropriate to disable the use of these files?

   A. Use the -l flag on **rsh** and **rlogin**

   B. Put an entry in the /etc/nologin file

   C. Delete the /etc/hosts.equiv and ~/.rhosts files and tell your users not to create new ones

   D. Change permissions of /etc/hosts.equiv and ~/.hosts to anything other than 600

4. Scenario: A network administrator has been asked to integrate a new RS/6000 to be used as a corporate mail server into the network. There are five nodes on the Ethernet II network, with a network address of 193.3.7.0 and a subnet mask of 255.255.255.0. The machine contains ATM, token-ring and integrated Ethernet adapters.

   Which one of the following files must be edited to allow the machine to be a TFTP server?

   A. /etc/bootptab

   B. /etc/rc.tcpip

   C. /etc/inetd.conf

   D. /etc/netsvc.conf

5. Which one of the following files on the local machine should be edited in order to enable user *Fred* to perform **rexec** commands on a remote machine without being prompted for a password?

   A. ~fred/.login

   B. ~fred/.netrc

   C. ~fred/.rhosts

   D. /etc/hosts.equiv

6. Once a file has been edited, which one of the following actions must occur before clients can use the bootp server?

   A. **telinit q**

   B. **refresh -s inetd**

   C. **startsrc -s bootpd**

   D. No action is required

7. All of the following services are controlled by inetd *except*:

   A. nfsd

   B. ftpd

   C. telnetd

   D. pop3d

5.7.1 Answers

The following are the preferred answers to the questions provided in this section:

1. B
2. B
3. C is the most secure, but D will work
4. C
5. B
6. B
7. A

5.8 Exercises

The following exercises provide sample topics for self study. They will help ensure comprehension of this chapter.

1. Verify, on your system, which network subsystems and subservers are running.
2. On a dedicated test system, try to disable the FTP facility. What file would you need to edit? Make a backup of the corresponding file before you edit it. Test it to see if it works! Once you have done the test, re-enable FTP by restoring the original file.

6

Network services administration

The AIX TCP/IP supports a large set of network services. The main network services are the following:

DNS	Domain Name System
NFS	Network File System
NIS	Network Information Services
BOOTP	BOOTstrap Protocol
DHCP	Dynamic Host Configuration Protocol
DDNS	Dynamic Domain Name System
SNMP	Simple Network Management Protocol

Some of the network services are covered in separate chapters. For DNS, refer to Chapter 8, “Domain Name System” on page 193. For NFS, refer to Chapter 7, “NFS” on page 149. For NIS, refer to Chapter 10, “NIS” on page 223.

The network services BOOTP, DHCP, and DDNS are described in the following sections.

6.1 Bootstrap protocol BOOTP

The Bootstrap Protocol (BOOTP) is used for providing IP addresses and IP parameters to systems on a TCP/IP network that are not configured. The system types could be network computers, X-terminals, network printers, and other machines that only have a minimal startup program in ROM.

Once BOOTP has provided the boot parameters, the actual downloading of image software is typically done with Trivial File Transfer Protocol (TFTP) or NFS.

The BOOTP uses UDP to bootstrap systems that request the IP address and additional information such as boot file from a BOOTP server. BOOTP is a draft standard protocol and its specifications can be found in RFC 951 Bootstrap Protocol.

The BOOTP client uses a broadcast on the local network, as it does not yet have an IP address.

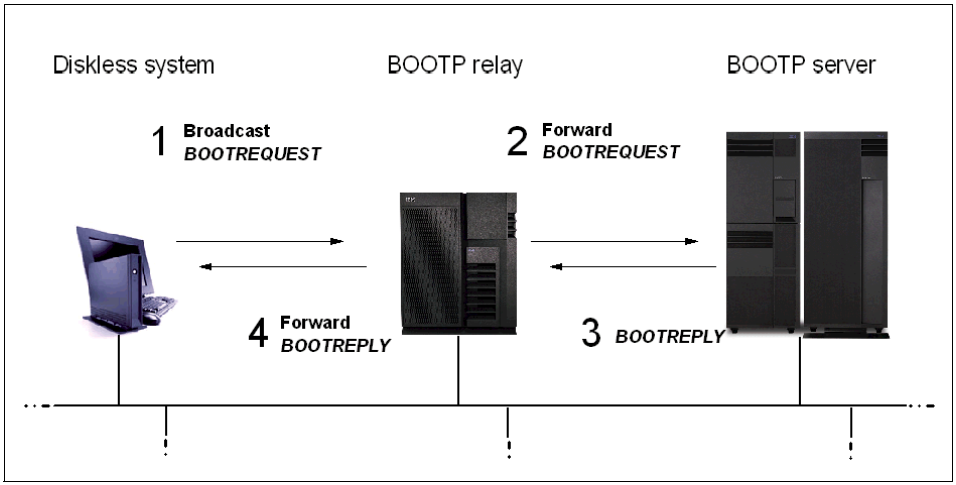


Figure 6-1 The BOOTP client/server message flow

The server replies to the broadcast with either a broadcast or unicast back to the client. The BOOTP request and replies contain a vendor-specific area that allows transmission of system information such as subnet mask, host name, domain name, default gateway, name servers, and other information.

By using a BOOTP server, the management of network machines can be centralized and administration becomes easier.

In situations where a lot of network clients requesting BOOTP are residing on smaller subnetworks without a BOOTP server, a router known as the BOOTP relay agent is required. This server forwards the BOOTP requests from the clients to the BOOTP server and similar are the BOOTP replies forwarded back to the requestor. This scheme of having one or multiple BOOTP relay agents allows consolidation of multiple networks with a central BOOTP server, thus reducing the overall network administration.

The BOOTP message flow between the client and the BOOTP server is illustrated in Figure 6-1 on page 132:

1. The client broadcasts a BOOTREQUEST datagram to the bootps service (port 67), which contains the hardware address of the client.
2. The datagram is picked up and forwarded by the BOOTP relay agent that listens to the same port 67. Note this might only happen in complex network scenarios.
3. The BOOTP server replies with a BOOTREPLY datagram message to the bootpc service (port 68). If the request came directly from the client, then the server might broadcast the request to 255.255.255.255. If the BOOTREQUEST came from a relay, the server can unicast the datagram to the relay.
4. The relay (if involved) will broadcast or unicast the BOOTREPLY to the client.

6.1.1 Configuring BOOTP

In AIX, the BOOTP is implemented in the server daemon bootpd, which is started by inetd (/etc/inetd.conf). Alternatively, the bootpd can be started in stand-alone mode using the flag -s. The bootpd daemon reads at startup a configuration file which, by default, is the /etc/bootptab. This file contains an entry for each client using the BOOTP service.

The following is an extract from a /etc/bootptab file:

```
...
# Legend:
#
#         first  -- hostname
#         field     (may be full domain name and probably should be)
#
#         hd     -- home directory
#         bf     -- bootfile
#         sa     -- server IP address to tftp bootfile from
#         gw     -- gateways
#         ha     -- hardware address
#         ht     -- hardware type
#         ip     -- host IP address
```

```
#         sm    -- subnet mask
#         tc    -- template host (points to similar host entry)
#         hn    -- name switch
#         bs    -- boot image size
#         dt    -- old style boot switch
#         T170  -- (xstation only) -- server port number
#         T175  -- (xstation only) -- primary/secondary boot host indicator
#         T176  -- (xstation only) -- enable tablet
#         T177  -- (xstation only) -- xstation 130 hard file usage
#         T178  -- (xstation only) -- enable XDMCP
#         T179  -- (xstation only) -- XDMCP host
#         T180  -- (xstation only) -- enable virtual screen
aixnc1:ht=token-ring:ha=0000E5740839:ip=9.55.43.28:sa=9.55.33.48:bf=kernel:hd=/
usr/netstation:ds=9.55.15.1:gw=9.55.15.53:sm=255.255.0.0
...
```

The entry shown in the bootptab file specifies a network station `aixnc1` and all the necessary information to boot it using TFTP.

The first entry is the client name.

`ht` Specifies the host hardware type, in this case token-ring.

`ha` Specifies the host hardware address.

`ip` Specifies the client's IP address.

`sa` Specifies the IP address of the TFTP server, where the client's boot file resides.

`bf` Specifies the name of the boot file (in this case, kernel).

`hd` Specifies the home directory on the TFTP server.

`ds` Specifies the domain name server address list.

`gw` Specifies the gateway address list. If this tag is defined, the `sm` (subnet mask) tag must also be defined.

6.2 Dynamic Host Configuration Protocol (DHCP)

The Dynamic Host Configuration Protocol (DHCP) provides a mechanism for dynamic allocation of IP addresses and configuration parameters on a TCP/IP network. DHCP is used extensively for client PCs (stationary PCs and laptops) or other network computing devices to relieve the network administration of manual configuration. The ability to move from network to network and automatically obtain a valid configuration is especially important for mobile users.

DHCP is based on the BOOTP protocol with the additional capability of automatic allocation of reusable network addresses and additional configuration options. The DHCP specifications can be found in RFC 2131 and RFC 2132.

DHCP messages use the same UDP port 67 for requests to servers and UDP port 68 for clients. A DHCP setup can coexist with BOOTP provided it is configured to do so (see more on this issue later in 6.2.3, “BOOTP and DCHP interoperation” on page 139).

DHCP consists of two components:

- A protocol that delivers host-specific configuration parameters from a DHCP server to a network host.
- A mechanism for the allocation of temporary or permanent network addresses to network host.

DHCP supports three mechanisms for IP address allocation:

Dynamic allocation	DHCP assigns an IP address for a limited period of time. This network address, called a *lease,* allows automatic reuse of addresses that no longer are in use.
Automatic allocation	DHCP assigns a permanent IP address to the host.
Manual allocation	The network address is assigned manually by a network administrator.

The following is a description of the DHCP interaction sequence between client and server to obtain a DHCP network address. Figure 6-2 illustrates the message flow.

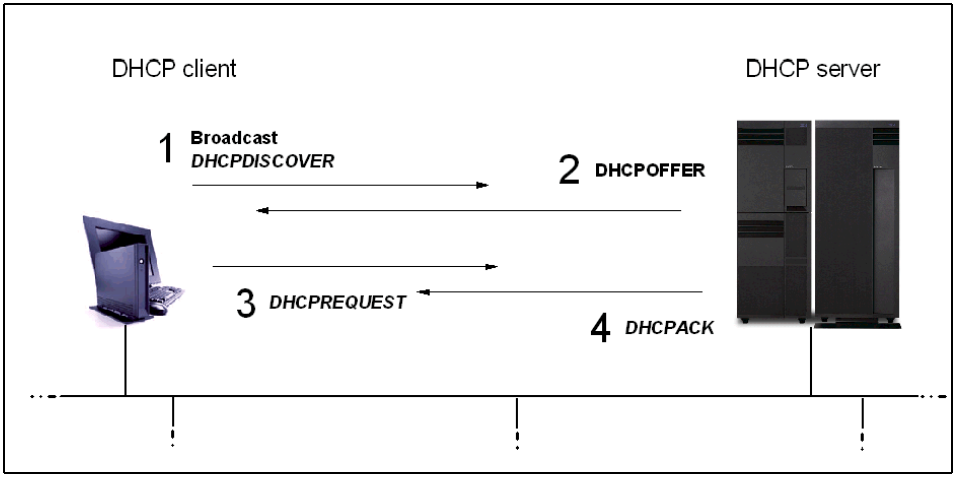


Figure 6-2 The DHCP client/server simple request message flow

1. The client broadcasts a DHCPDISCOVER message on the local subnet. This message may include some options such as network address suggestion or lease duration.
2. The DHCP server responds with a DHCPOFFER message that includes an available network address and other configuration options. The address offered to the client is reserved in order to prevent it from being used by other requesting clients. Multiple DHCP servers might react on the client broadcast, so multiple DHCPOFFERs might be sent.
3. The client chooses the configuration parameters offered and sends a DHCPREQUEST message back indicating which server it has selected and the requested IP address option. The DHCP server receives the DHCPREQUEST broadcast from the client. In case of multiple offers, the DHCP servers not selected by the DHCPREQUEST message use this message to drop out of the transaction.
4. The DHCP server selected in the DHCPREQUEST message commits itself to the client with a DHCPACK message containing the configuration parameters for the client.

After step 4 the client is fully configured. This simple scenario illustrates only a successful request scenario. The following parts of the DHCP client initialization are not shown:

- If the client is not satisfied with the parameters offered, it may send a DHCPDECLINE and restart the request process again.
- A client renews its lease prior to expiration by issuing another DHCPREQUEST.
- If no DHCPACK is received, the client times out and retries from the beginning (step 1.).
- When shutting down, a client makes a DHCPRELEASE and releases its parameters.

6.2.1 DHCP server configuration

The DHCP server program dhcpsd implements the DHCP service described above. At startup the DHCP server is configured by reading the /etc/dhcpsd.cnf file, which specifies the server's initial database of options and addresses.

The dhcpsd server is started in the /etc/rc.tcpip file, or it can be started from Web-based System Manager, from SMIT, or through SRC commands.

```
# startsrc -s dhcpsd
0513-059 The dhcpsd Subsystem has been started. Subsystem PID is 17744.
```

Configuring a good DHCP server environment on your network is not a trivial thing to do. Many considerations must be taken into account, such as what subnets in your networks have DHCP clients, which pool of addresses are available for each network, which gateways need to be set up, and so on.

Following is a simple example of a DHCP server configuration file /etc/dhcpsd.cnf file:

```
...
#
#   dhcpsd.cnf -- DHCP Server Configuration File
#
#
#   This file contains directives that can be specified by the
#   server's administrator to configure the server and enforce
#   policies.
...
numLogFiles 6
logFileSize 1000
logFileName /usr/tmp/dhcpsd.log
logItem SYSERR
logItem OBJERR
logItem PROTERR
logItem WARNING
leaseTimeDefault 1 day
leaseExpireInterval 6  hour
#
network 10.0.0.0 24
{
        subnet 10.47.1.0 10.47.1.55-10.47.1.100
        {
                option 1  255.255.255.0
                option 3  10.47.1.1
                option 6  10.47.1.2
                option 15 itsc.austin.ibm.com
        }
}
```

The numLogFiles, logFileSize, logFileName, and logItem parameters are for the logging configuration. The parameter `leaseTimeDefault` specifies the default lease duration. The default is 1 hour, while in this example it is specified to 1 day. The leaseExpireInterval parameter specifies the time a lease expiration condition is examined.

This example shows a DHCP configuration for the subnet 10.47.1.0. The DHCP server assigns IP addresses ranging from 10.47.1.55 to 10.47.1.100. Each

DHCP client will receive the following settings: subnet mask (option 1) is set to 255.255.255.0, the default gateway (option 3) is set to 10.47.1.1, the domain name server (option 6) is set to 10.47.1.2, and finally the domain name (option 15) is set to itsc.austin.ibm.com.

A large set of options can be configured in DHCP. The description of the DHCP configuration option numbers are located in the file /etc/option.file.

To assist the administration of an DHCP server in AIX 4.3, the system utility dadmin is provided. The **dadmin** command lets the DHCP administrator query and modify the state of the DHCP server database. Both local and remote DHCP servers can be queried for a pool of IP addresses or IP address status. Other possible administration commands delete an IP address mapping, alter the tracing level, and refresh the server.

For example:

```
# dadmin -h server4 -v -s
Connecting to the DHCP server: server4
Got a socket, attempting to connect.

Connected to server4 successfully.
Send of header completed.

PLEASE WAIT....Gathering Information From the Server....PLEASE WAIT

Receive of header completed.

IP Address      Status  Lease Time Start Time  Last Leased Proxy ClientID

10.47.1.55      Leased     6:00:00 06/27 12:11 06/27 12:11 FALSE 1-00062995ec27
...
```

6.2.2 DHCP/BOOTP relay agent configuration

The dhcprd daemon is the DHCP relay agent for forwarding both BOOTP and DHCP requests. The UDP broadcasts sent by a BOOTP or DHCP client are not allowed to be passed through network gateways and routers; thus, a BOOTP/DHCP relay agent, the dhcprd daemon, has to send these packets to the appropriate servers.

The dhcprd is started using SRC, either in /etc/rc.tcpip (by uncommenting the corresponding entry) or by interactively using the **startsrc** command.

The dhcprd daemon reads the configuration file /etc/dhcprd.cnf at startup.

An example of an /etc/dhcprd.cnf file is as follows:

```
numLogFiles 4
logFileSize 100
logFileName /usr/tmp/dhcprd.log
logItem SYSERR
logItem OBJERR
server 10.47.1.1
```

The numLogFile, logFileSize, logFileName, and logItem have the same parameter format as used in the DHCP server configuration file, namely logging parameters. The server parameter specifies the IP address of the server to which a DHCP relay agent should forward BOOTP or DHCP datagram. Multiple servers can be specified; all will receive a datagram message.

Since the dhcprd uses the same port as the bootpd daemon (port 67), you can only have one (either dhcprd or bootpd) daemon running. If you choose the dhcprd daemon, you will need to uncomment bootp from the /etc/inetd.conf file, then enter **refresh -s inetd** on the command line. If bootpd is running, this program needs to be stopped before starting the daemons.

6.2.3 BOOTP and DCHP interoperation

The format of DHCP messages is based on the format of BOOTP messages, which allows BOOTP and DHCP clients to coexist. Every DHCP message contains an IP Address Lease Time (DHCP message type option 51). Any message without this option is assumed to be from a BOOTP client.

The DHCP server responds to BOOTPREQUEST messages with BOOTPREPLY. A DHCP server may offer static addresses or automatic addresses to a BOOTP client (although not all BOOTP implementations will understand automatic addresses). If an automatic address is offered to a BOOTP client, then that address must have an infinite lease time, as the client will not understand the DHCP lease mechanism.

To support BOOTP clients from a DHCP server, the dhcpsd flag supportBOOTP must be set.

Add the following line to your dhcpsd configuration file /etc/dhcpsd.cnf:

```
...
supportBOOTP Yes
...
```

To support BOOTP clients from a DHCP server, the /etc/bootptab configuration must be migrated to DHCP configuration. The bootptodhcp utility is provided in order to support this migration.

6.2.4 DHCP client configuration

The DHCP client daemon is implemented in the dhcpcd daemon. It requests IP address and parameters from a DHCP server. When an AIX system is configured to run with a DHCP client, the dhcpcd entry in the /etc/rc.tcpip startup script needs to be uncommented. Notice that the dhcpcd is, obviously, the first network daemon to be started.

At startup, the dhcpcd reads its configuration file /etc/dhcpcd.ini.

An example of the /etc/dhcpcd.ini is as follows:

```
#
#   dhcpcd.ini -- DHCP Client configuration file
#
#
#   This file contains directives that can be specified
#   to configure the client.
numLogFiles      4
logFileSize      100
logFileName     /usr/tmp/dhcpcd.log
logItem SYSERR
updateDNS "/usr/sbin/dhcpaction '%s' '%s' '%s' '%s' A NONIM >> /tmp/updns.out
2>&1 "
```

The numLogFiles, logFileSize, logFileName and logItem entries have the same parameter format used in the DHCP server configuration file, namely logging parameters. The updateDNS parameter is a quoted string used for executing a program, in this case dhcpaction, to update the DNS server with the inverse mapping of the IP address provided by DHCP and host name of the machine. For more information about DNS updates, see 6.3, “Dynamic Domain Name System (DDNS)” on page 140.

Instead of editing /etc/rc.tcpip and /etc/dhcpcd.ini manually, a preferred way to configure the DHCP client is using the **smit usedhcp** fast path.

6.3 Dynamic Domain Name System (DDNS)

The Domain Name System is a static implementation of naming network units, providing host names for statically allocated IP addresses. In order to take advantage of DHCP and dynamically assigned IP addresses and still be able to allocate meaningful host names, the Dynamically Domain Name System (DDNS) was specified. In a DDNS environment, when the client receives its address from a DHCP server, it automatically updates its A record on the DNS server with the new address. In AIX Version 4.3, the program nsupdate is used to update information on a DDNS server.

For more information on DDNS, refer to subsection on DHCP and the Dynamic Domain Name System (DDNS) in the *AIX Version 4.3 System Management Guide: Communications and Networks* and look in the man page for nsupdate.

6.4 Simple Network Management Protocol (SNMP)

SNMP is used by network hosts to exchange information on the management of networks. SNMP network management is based on the familiar client/server model that is widely used in TCP/IP-based network applications. Each host that is to be managed runs a process called an agent. The agent is a server process that maintains the Management Information Base (MIB) database for the host. Hosts that are involved in network management decision-making may run a process called a manager. A manager is a client application that generates requests for MIB information and processes responses. In addition, a manager may send requests to agent servers to modify MIB information.

The SNMP daemon is started using the **snmpd** command. This command may be issued only by a user with root privileges or by a member of the system group.

6.4.1 Files and file formats

The following files and formats are used with the SNMP.

mib.defs Defines the MIB variables the SNMP agent should recognize and handle. The **snmpinfo** command requires a set format to be followed for the /etc/mib.defs file.

mibII.my Defines the ASN.1 definitions for the MIB variables as defined in RFC 1213.

smi.my Defines the ASN.1 definitions by which the SMI is defined as in RFC 1155.

snmpd.conf Defines the configuration file for the snmpd agent.

ethernet.my Defines the ASN.1 definitions for the MIB variables defined in RFC 1398.

fddi.my Defines the ASN.1 definitions for the MIB variables defined in RFC 1512.

generic.my Defines the ASN.1 definitions for the MIB variables defined in RFC 1229.

ibm.my Defines the ASN.1 definitions for the IBM enterprise section of the MIB tree.

token-ring.my Defines the ASN.1 definitions for the MIB variables defined in RFC 1231.

unix.my	Defines the ASN.1 definitions for a set of MIB variables for memory buffer (mbuf) statistics, SNMP multiplexing (SMUX) peer information, and various other information.
view.my	Defines the ASN.1 definitions for the SNMP access list and view tables.
snmpd.peers	Defines a sample peers file for the snmpd agent.

6.4.2 SNMP Requests for Comments (RFCs)

SNMP is defined in several Requests for Comments (RFCs), which are available from the Network Information Center at SRI International, Menlo Park, California.

The following RFCs define SNMP:

RFC 1155	Defines the structure of management information.
RFC 1157	Defines the SNMP to create requests for Management Information Base (MIB) information and formatting responses.
RFC 1213	Defines the MIB for network management.
RFC 1227	Defines the SNMP multiplexing (SMUX) protocol for extending base SNMP agents.
RFC 1228	Defines the Distributed Protocol Interface (DPI) for extending base SNMP agents.
RFC 1229	Defines an extension to the interfaces table as defined in RFC 1213.
RFC 1231	Defines an extension to the interfaces table for token-ring devices.
RFC 1398	Defines an extension to the interfaces table as Ethernet devices.
RFC 1512	Defines an extension to the interfaces table for Fiber Distributed Data Interface (FDDI) devices.

The snmpd.conf file

The snmpd.conf file provides the configuration information for the snmpd agent. This file can be changed while the snmpd agent is running. If the **`refresh`** or **`kill -1`** command is issued, the snmpd agent will reread this configuration file. The snmpd agent must be under System Resource Control (SRC) for the **`refresh`** command to force the reread.

This configuration file contains:

- Entries for community names. The community entry specifies the communities, associated access privileges and MIB views the snmpd agent allows.
- Access privileges and view definitions for incoming Simple Network Management Protocol (SNMP) request packets. The view entry specifies the MIB subtrees to which a particular community has access.
- Entries for host destinations for trap notification. The trap entry specifies the hosts the snmpd agent notifies in the event a trap is generated.
- Entries for log file characteristics. The logging entry specifies the characteristics for the snmpd agent logging activities if logging is not directed from the **snmpd** command with the -f option.
- Entries for snmpd-specific parameters. The snmpd entry specifies configuration parameters for the snmpd agent.
- Entries for SNMP Multiplexing Protocol (SMUX) association configurations. The SMUX entry specifies configuration information for SMUX associations between the snmpd agent and SMUX peer clients.
- Entries for the sysLocation and sysContact variables. The sysLocation and sysContact entries specify the values of the sysLocation and sysContact variables.

The snmpd.conf file must be owned by the root user. If the snmpd.conf file is not owned by root, or if the snmpd daemon cannot open the configuration file, the snmpd daemon issues a FATAL message to the log file if logging is enabled and snmpd terminates.

Certain rules apply for specifying particular parameters in entries in the snmpd.conf configuration file. Some entries require the specification of object identifiers, object names, or both. The following rules apply:

- An object identifier is specified in dotted numeric notation and must consist of at least three elements. The maximum number of elements in the object identifier is 50. Elements are separated by a . (period). The first element must be a single digit in the range of 0 to 2. The second element must be an integer in the range of 1 to 40. The third and subsequent elements must be integers in the range of 1 to the size of an unsigned integer.
- An object name consists of a textual name with an optional numeric instance. The object name must be known to the snmpd agent. Object names typically are names of nodes in the Management Information Base (MIB) tree. If the root of the MIB tree, iso, is specified as an object name, the numeric instance is absolutely required. A . (period) separates the textual name from the numeric instance.

Following is an example of the last lines of the /etc/snmpd.conf file:

```
#
# NOTE:  Comments are indicated by # and continue to the end of the line.
#        There are no restrictions on the order in which the configuration
#        entries are specified in this file.
#
##############################################################################

logging         file=/usr/tmp/snmpd.log         enabled
logging         size=0                          level=0

community       public
community       private 127.0.0.1 255.255.255.255 readWrite
community       system  127.0.0.1 255.255.255.255 readWrite 1.17.2

view            1.17.2          system enterprises view

trap            public          127.0.0.1       1.2.3   fe      # loopback

#snmpd          maxpacket=1024 querytimeout=120 smuxtimeout=60

smux            1.3.6.1.4.1.2.3.1.2.1.2         gated_password  # gated
smux            1.3.6.1.4.1.2.3.1.2.2.1.1.2     dpid_password   #dpid
```

The snmpd.peers file

The snmpd.peers file defines a sample peers file for the snmpd agent.

In the following example, the file layout is explained in the /etc/snmpd.peers file:

```
##############################################################################
# Syntax:
#
#       <name>  <object id>     <password>      <priority>
#
#    where <name> is the name of the process acting as an SMUX peer and
#    <object id> is the unique object identifier in dotted decimal
#    notation of that SMUX peer.  <password> specifies the password that the
#    snmpd daemon requires from the SMUX peer client to authenticate
#    the SMUX association.  The highest priority is 0 (zero).  The lowest
#    priority is (2^31)-1.  The default password is the null string.  The
#    default priority is 0 (zero).  Fields to the right of <object id> are
#    optional, with the limitation that no fields to the left of a specified
#    field are omitted.
#
#    Each token is separated by white space, though double-quotes may be
#    used to prevent separation.
#
##############################################################################
```

```
"gated"     1.3.6.1.4.1.2.3.1.2.1.2     "gated_password"
```

"dpid2" 1.3.6.1.4.1.2.3.1.2.2.1.1.2 "dpid_password"

6.5 Command summary

The following sections provide descriptions of the key commands discussed in this chapter. For a complete reference of the following commands, consult the AIX product documentation.

6.5.1 The dadmin command

The **dadmin** command is used to query and modify the status of the DHCP server. The command has the following syntax:

```
dadmin [-?] [-v] [-h Hostname] [-f] -d IpAddress | [-x] -i | [-x] -s | -t
on|off|Value | -q IpAddress | -p IpAddress | -c Clientld
```

The commonly used flags are provided in Table 6-1.

Table 6-1 Commonly used flags of the dadmin command

Flag	Description
-v	Toggle the verbose mode.
-h host	The host name of the DHCP server.
-s	Displays the status of each address in the DHCP server's configured pools.

6.6 Quiz

The following assessment questions help verify your understanding of the topics discussed in this chapter.

1. A company has a network in which hosts are frequently added to and removed from the network or are reconfigured. Which one of the following methods should be used for host name resolution?

   A. DDNS

   B. DHCP

   C. inetd

   D. /etc/hosts

2. Which one of the following services allows complete remote administration of the local network?

   A. FTP

   B. NIS

   C. SNMP

   D. telnet

3. Which one of the following protocols utilizes MIBs on a client system to remotely monitor control functions on that client?

   A. NTP

   B. IPNG

   C. SMTP

   D. SNMP

4. Which one of the following should be disabled if a host is to act as a DHCP server?

   A. tftpd

   B. gated

   C. snmpd

   D. bootpd

5. Which one of the following files must be edited to enable bootpd?

   A. /etc/inittab

   B. /etc/inetd.conf

   C. /etc/bootptab

   D. /etc/netsvc.conf

6. A machine with the hardware address 0XAB213BAFEE0B is on the subnet 9.67.112.0. It has the following attributes:

```
Lease Time Default30 minutes
Lease Expire Interval3 minutes
Support Bootpyes
Support Unlisted Clientsyes

Network 9.0.0.0  24
Subnet 9.2.218.09.2.218.1-9.2.218.128
Subnet 9.67.112.09.67.112.1-9.67.112.64
Client   6 Oxab213bafee0b0
```

Which one of the following address ranges will the DHCP server assign to the client?

A. 9.2.218.1 to 9.2.218.218

B. 9.67.112.1 to 9.67.112.64

C. 9.67.112.65 to 9.67.112.128

D. An address will not be assigned to the client

6.6.1 Answers

The following are the preferred answers to the questions provided in this section:

1. A
2. C
3. D
4. D
5. B
6. D

6.7 Exercises

The following exercises provide sample topics for self study. They will help ensure comprehension of this chapter.

1. Set up DHCP on an isolated test network, using the example in 6.2.1, “DHCP server configuration” on page 136 as input for a dhcpsd configuration file.
2. On a system running DHCP server, use the **dadmin** command to query the current status.
3. What is the configuration parameter for allowing BOOTP and DHCP to interoperate? View other configuration options in the /etc/options.file.
4. How is the snmpd.conf file used?

7

NFS

In this chapter, the following topics are discussed:

- NFS protocols and daemons
- NFS server considerations
- NFS client considerations
- Automount

NFS is an acronym for Network File System, a product developed by Sun Microsystems. This is a distributed file system implementation providing remote, transparent access to files and directories. AIX supports the latest NFS protocol update, NFS Version 3. AIX also provides an NFS Version 2 client and server and is therefore providing backward compatibility with existing install bases of NFS clients and servers. Negotiation will occur to check what is the highest version of NFS supported by both involved systems.

NFS operates on a client/server basis. An NFS server has files on a local disk, which are accessed through NFS on a client machine. To handle this operation, NFS consists of:

- Networking protocols
- Client and server daemons
- Kernel extensions

The kernel extensions are outside the scope of this chapter, but the protocols and the daemons will be covered. The following sections discuss the protocols involved.

7.1 Protocols

The NFS specific protocols are Remote Procedure Call protocol (RPC) and eXternal Data Representation (XDR) protocol. Figure 7-1 shows the relationships between the protocols:

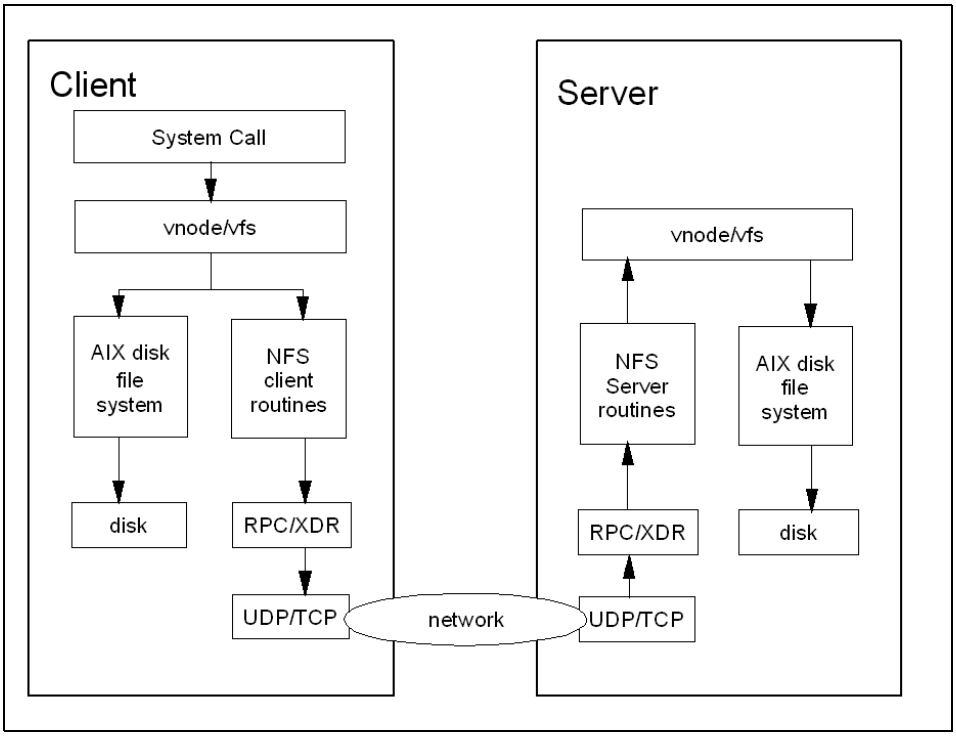


Figure 7-1 NFS protocol flowchart

7.1.1 UDP or TCP

As all traffic on the Internet is more or less defined by the use of IP at the network layer, so is NFS. On the next layer, the Transport Layer, the choice of UDP or TCP is optional on AIX.

The AIX version decides the default set of the NFS version and transport protocol. Table 7-1 on page 151 defines the order of default mount options and the *fallback* order if the default options are not available.

Table 7-1 NFS protocols

AIX Version	NFS Version 3		NFS Version 2	
4.2.1	UDP	TCP	UDP	TCP
4.3.x	TCP	UDP	TCP	UDP
5.1.x	TCP	UDP	TCP	UDP

There are many differences in the behavior, especially in timeout handling, between NFS using TCP and NFS using UDP. More on this subject is found in 7.4.2, “Client mount options” on page 171.

7.1.2 RPC

RPC is a library of procedures. The procedures allow one process (the client process) to direct another process (the server process) to execute procedure calls as though the client process had executed the calls in its own address space. Because the client and the server are two separate processes, they are not required to be on the same physical system, although they can.

The RPC call used is based on the file system action taken by the user. For example, when issuing an **`ls -la`** command on an NFS mounted directory, the long listing will be done through a RPC named NFSPROC3_FSINFO, which will initiate the long listing on the server, which in turn will send the output from the command through RPC back to the client. To the user, this transaction is totally transparent.

The /etc/rpc file contains a list of server names and their corresponding RPC program numbers and aliases. For example:

```
# more /etc/rpc
portmapper      100000  portmap sunrpc
nfs             100003  nfsprog
ypserv          100004  ypprog
mountd          100005  mount showmount
ypbind          100007
yppasswdd       100009  yppasswd
statmon         100023
status          100024
bootparam       100026
ypupdated       100028  ypupdate
ypxfrd          100069  ypxfr
pcnfsd          150001
autofs          100099  automount       #209812
```

Because the server and client processes can reside on two different physical systems, which may have completely different architectures, RPC must address the possibility that the two systems may not represent data in the same manner. Therefore, RPC uses data types defined by the eXternal Data Representation (XDR) protocol.

7.1.3 XDR

XDR is the specification for a standard representation of various data types. By using a standard data type representation, data can be interpreted correctly, even if the source of the data is a machine with a completely different architecture.

XDR is used when the vnode points out that the file or directory accessed is not a local file or directory, but resides on a remote system. A conversion of data into XDR format is needed before sending the data. Conversely, when it receives data, it converts the data from XDR format into its own specific data type representation.

7.2 NFS daemons

Depending on the task, some of the NFS-related daemons are started on a system. Servers need the following daemons in an active state:

- portmap
- nfsd
- rpc.mountd

The client only needs the following daemons to be able to mount a remote directory:

- portmap
- biod

As default, the startup of NFS services is handled by /etc/rc.nfs called by init from /etc/inittab. When looking at these scripts, you can see that the default startup also include the following daemons on both a server and a client system:

- rpc.statd
- rpc.lockd

It is important to remember that the portmap must be started before starting the NFS daemons.

The relationship between NFS daemons on the server side and the client side is shown in Figure 7-2 on page 153.

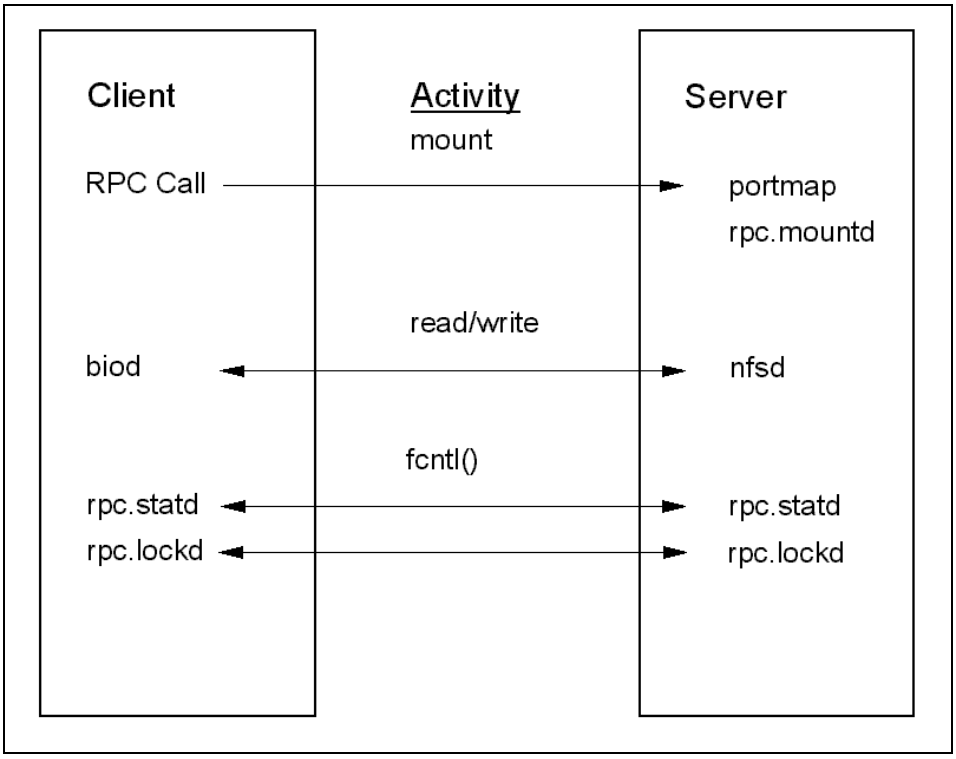


Figure 7-2 NFS daemon activity

In the following sections are overviews of the different tasks the daemons handle.

7.2.1 The portmap daemon

The portmap daemon converts RPC program numbers into Internet port numbers. When an RPC server starts up, it registers with the portmap daemon. The server tells the daemon which port number it is listening to and which RPC program numbers it serves. By this process, the portmap daemon knows the location of every registered port used by RPC servers on the host, and which programs are available on each of these ports. When mounting, the mount request starts with an RPC call named GETPORT that calls the portmap which, in turn, will inform the client of the port number that the called RPC server listens to. After this, the port number is used as reference for further communication. This is why the NFS daemons need to be registered with the portmap daemon. See Figure 7-3 on page 155.

A client consults the portmap daemon only once for each program the client tries to call. The portmap daemon tells the client which port to send the call to. The client stores this information for future reference.

Since standard RPC servers are normally started by the inetd daemon, the portmap daemon must be started before the inetd daemon is invoked.

Note: If the portmap daemon is stopped or comes to an abnormal end, all RPC servers on the host must be restarted.

7.2.2 The rpc.mountd daemon

rpc.mountd handles the actual mount service needed when a client sends a mount request with an RPC procedure named MOUNTPROC3_MNT to the server. The mountd daemon finds out which file systems are available for export by reading the /etc/xtab file. In addition, the mountd daemon provides a list of currently mounted file systems and the clients on which they are mounted. This list can be shown by the **showmount** command.

For example:

```
# showmount -a
server4f.itsc.austin.ibm.com:/home
server4f.itsc.austin.ibm.com:/tmp/thomasc/testfs
```

The output shows that server4 has mounted /tmp/thomasc/testfs and /home.

The mount services is provided on the server from the /usr/sbin/rpc.mountd daemon, and the **/usr/sbin/mount** command on the client. Figure 7-3 on page 155 has a flowchart of a mount.

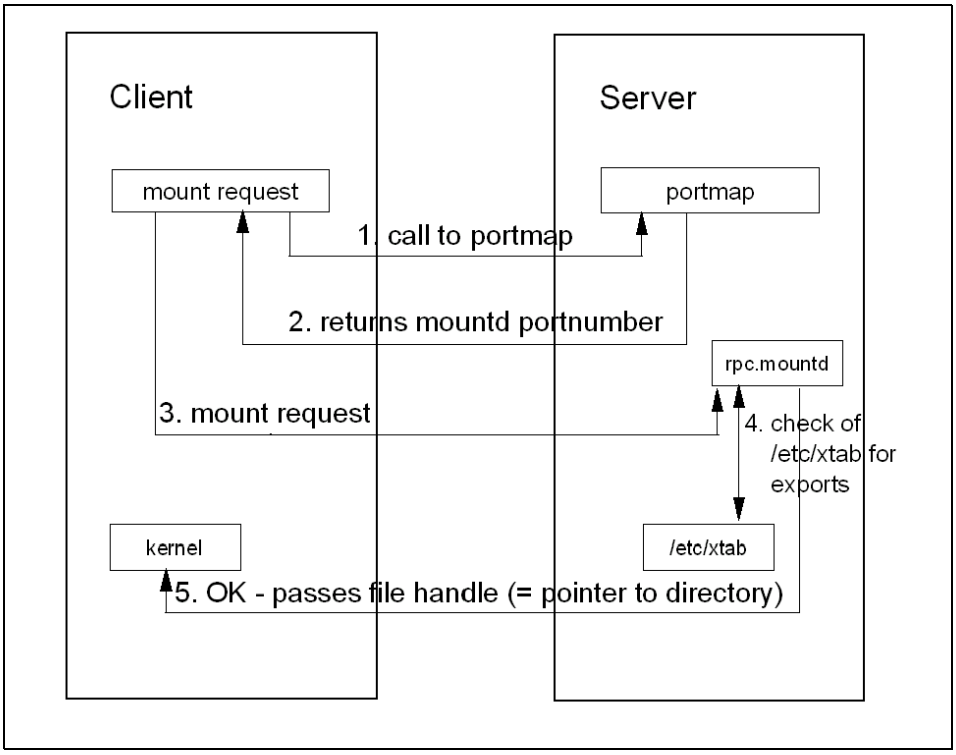


Figure 7-3 NFS mount

7.2.3 The nfsd daemon

The nfsd daemon runs on a server and handles client requests for file system operations. Each daemon handles one request at a time. This means that on the server side, the receipt of any one NFS protocol request from a client requires the dedicated attention of an nfsd daemon until that request is satisfied, and the results of the request processing are sent back to the client. The nfsd daemons are the active agents providing NFS services. The default number of nfsd started from /etc/rc.nfs is eight.

The NFS daemons are inactive if there is no NFS requests to handle. When the NFS server receives RPC calls on the nfsd's receive socket, nfsds are awakened to pick the packet of the socket and invoke the requested operations. As mentioned earlier, the nfsd taking a packet is dedicated to that one operation until its completion. This is regardless of the type of operation.

7.2.4 The biod daemon

The block I/O daemon (biod) runs on all NFS client systems. When a user on a client wants to read or write to a file on a server, the biod daemon sends this request to the server. For each read or write request, one biod is requested. The biod daemon is activated during system startup and runs continuously.

The number of biods are limited on a per-mount-point basis. Up to six biods can work on any one remote mounted file system at any time. But the default number of started biods are six for NFS Version 2 and four for NFS Version 3. The reason to set a limit on biods per mount is that a unregulated number of biods may overload a server.

7.2.5 The rpc.lockd daemon

When mounting file systems that could be accessed both locally and remotely, the system need some kind of file locking mechanism to maintain file system integrity. This is handled by the rpc.lockd and the rpc.statd. These daemons also cooperate to reestablish locks on files after a server crash.

The lockd processes lock requests. The lockd forwards lock requests for remote data to the server lock daemon through the RPC package. The lockd then asks statd (status monitor) for monitor service. The reply to the lock request is not sent to the kernel until both the statd and the server lockd reply. The statd should always be started before lockd.

If either the status monitor (rpc.statd, covered in 7.2.6, “The rpc.statd daemon” on page 156) or the server lock daemon is unavailable, the reply to a lock request for remote data is delayed until all daemons (that is, rpc.lockd and rpc.statd on both sides) become available.

When a server recovers, it waits for a grace period for all client lockds to submit reclaim requests. The client lockd are notified of the server recovery by statd. At this stage, the daemons resubmit previously granted lock requests.

7.2.6 The rpc.statd daemon

The statd daemon interacts with the lockd to provide crash and recovery functions for the locking services on NFS. The statd should always be started before lockd.

The status monitor maintains information on the location and status of connections in the /etc/sm directory, the /etc/sm.bak file, and the /etc/state file. When restarted, the status monitor daemon queries these files and tries to re-establish the connection it had prior to the server crash. If you need to start

these daemons and release existing locks, delete these files before restarting the statd daemon. After this, start the lockd daemon. The communication occurring at file locking is shown in Figure 7-4.

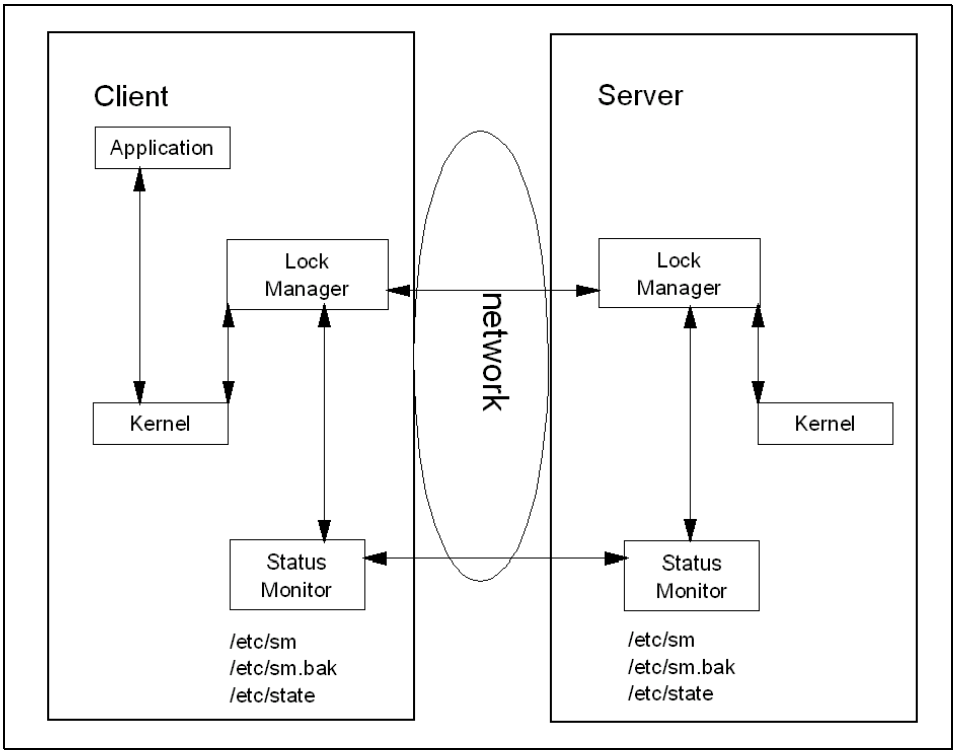


Figure 7-4 NFS file locking request

7.3 NFS server considerations

Because the NFS protocol is designed to be operating system independent, the connection between the server and the client is stateless. Statelessness means that the server does not have to maintain state of its clients to be able to function correctly. Statelessness does *not* mean that the server is not allowed to maintain the state of its clients. In the NFS configuration, the server is *dumb* and the client is *smart*, which means that the client has to convert the file access method provided by the server into an access method understood by its applications.

Considering this, there is really not much to do at the server side but export the file system, directory or file chosen, start the daemons, and control performance. In the following sections, these issues will be covered in more detail.

7.3.1 Exporting file systems from a server

The fileset needed for the NFS server function is named bos.net.nfs.server and is part of the default definition of the Server bundle.

The connection between /etc/exports, exportfs, and /etc/xtab

There are two files used for export on a server. The first one, the one that is actually edited, is /etc/exports. This is a simple text file that can be directly edited with your favorite editor or edited through **smitty nfs** submenus. A simple example of this file follows:

```
# more /etc/exports
/tmp/thomasc -root=server4,access=server1:server2:server4
/tmp/thomasc/testfs -ro
```

This /etc/exports file defines, with `access=,` that a mount of /tmp/thomasc can be made from server1, server2, and server4. The statement `-root=server4` allows root access only to the root users from server4. The default is for no hosts to be granted root access. As mentioned earlier, the **showmount** command is helpful in checking what is exported from a specified server, but the **showmount** command will not show whether some system is granted root access or not, as shown in the following output:

```
# showmount -e server3
export list for server3:
/tmp/thomasc server1,server2,server4
```

As shown in the output from **showmount**, there is no export done of /tmp/thomasc/testfs (-ro in the /etc/exports file shows that the intent was to do a read-only export). The reason is that the actual NFS subsystem does not use the /etc/exports file, but the /etc/xtab file. This file is updated at execution of the **exportfs** command as shown in the example:

```
# exportfs -a
```

The **exportfs -a** command will read all entries in the /etc/exports file and update the /etc/xtab with these entries. Now, the output from **showmount -e server3** appears as follows:

```
# showmount -e server3
export list for server3:
/tmp/thomasc        server1,server2,server4
/tmp/thomasc/testfs (everyone)
```

Again, there is no entry in the **showmount** command whether the file system exported is read-only or read-write. But when trying to create a file in the directory, the following error message appears:

```
# touch testfile
```

```
touch: 0652-046 Cannot create testfile.
```

When using smitty, **smitty mknfsexp** does both these steps: updates the /etc/exports and executes the **exportfs** command, as shown in Figure 7-5.

```
                        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                    [/tmp/thomasc/testfs]     /
* MODE to export directory                            read-write               +
  HOSTS & NETGROUPS allowed client access            []
  Anonymous UID                                      [-2]
  HOSTS allowed root access                          [server4]
  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
```

Figure 7-5 smitty mknfsexp screen

The /etc/rmtab file

When mountd accepts a mount request from a client, it notes the directory name passed in the mount request and the client host name in /etc/rmtab. Entries in /etc/rmtab are long-lived; they remain in the file until the client performs an explicit unmount of the file system. It is this file that is read to generate the **showmount -a** output.

The information in /etc/rmtab can become stale if the server goes down abruptly, or if clients are physically removed without unmounting the file system. In this case, you would remove all locks and the rmtab file. For example:

```
# stopsrc -g nfs
# stopsrc -s portmap
# cd /etc
# rm -fr sm sm.bak state xtab rmtab
# startsrc -s portmap
# startsrc -g nfs
# exportfs -a
```

7.3.2 Controlling server daemons

As discussed in previous sections, the daemons to control on the server side are portmap, rpc.mountd, nfsd, and the lock-handling daemons rpc.statd and rpc.lockd. You do not need to have rpc.statd and rpc.lockd running to be able to mount, although it is recommended and they are started as the defaults from /etc/rc.nfs. In the following sections, a couple of scenarios are covered describing what happens when some of these daemons are inactive.

Portmap problem determination

In the following scenario the portmap daemon is stopped.

```
# showmount -a
server4f.itsc.austin.ibm.com:/tmp/thomasc/testfs
# stopsrc -s portmap
0513-044 The portmap Subsystem was requested to stop.
```

Existing mounts (server4) are still accessible because the biod/nfsd interaction is not dependent on portmap after the initial client contact. For example:

```
# mount
  node    mounted      mounted over    vfs    date            options
-------- ---------- -------------     ----   ------          ------------
         /dev/hd4     /               jfs    Jun 11 16:46    rw,log=/dev/hd8
         /dev/hd2     /usr            jfs    Jun 11 16:46    rw,log=/dev/hd8
         /dev/hd9var  /var            jfs    Jun 11 16:47    rw,log=/dev/hd8
         /dev/hd3     /tmp            jfs    Jun 11 16:47    rw,log=/dev/hd8
         /dev/hd1     /home           jfs    Jun 11 16:47    rw,log=/dev/hd8
         /dev/cd0     /exinfocd       cdrfs  Jun 20 08:27    ro
server3  /tmp/thomasc/testfs /tmp/server3ro   nfs3   Jun 20 19:04
# cd /tmp/server3ro
# touch tesfile2
# ls -la testfile2
-rw-r--r--   1 root     staff          0 Jun 21 14:02 testfile2
```

When trying to use **showmount -e server3** from server1 (which does not have any active mount) to see the exported directories, the command will hang. The **showmount -e** command communicates with the rpc.mountd daemon, which is pointed out by the portmap daemon.

When trying to mount the directory, an iptrace of the event will show that the portmap port, 111, is unreachable:

```
# startsrc -s iptrace -a " -a -s server3 -b /tmp/iptrace2.bin"
0513-059 The iptrace Subsystem has been started. Subsystem PID is 16526.
```

The command example starts the iptrace through SRC with some useful flags (the -a outside the quotation marks is an attribute flag for the **startsrc** command):

-a (within the quotation marks) suppresses ARP requests

-s defines host to trace

-b bidirectional traffic

In the next step, the mount is initiated. The command will eventually hang:

```
# mount server3:/tmp/thomasc/testfs /tmp/thomasc
```

The event to trace was the mount try. The **iptrace** command can now be stopped with:

```
# stopsrc -s iptrace
```

Use the **ipreport** command to convert the binary iptrace file to ASCII format:

```
# ipreport -srn /tmp/iptrace2.bin > /tmp/thomasc/ipreport2.out

# more /tmp/thomasc/ipreport2.out

IPTRACE version: 2.0
Packet Number 1
TOK: ====( 106 bytes transmitted on interface tr0 )==== 14:30:59.084118759
TOK: 802.5 packet
TOK: 802.5 MAC header:
TOK: access control field = 0, frame control field = 40
TOK: [ src = 00:06:29:be:b1:dc, dst = 00:06:29:be:d2:a2]
TOK: 802.2 LLC header:
TOK: dsap aa, ssap aa, ctrl 3, proto 0:0:0, type 800 (IP)
IP:     < SRC =      9.3.240.56 >  (server1.itsc.austin.ibm.com)
IP:     < DST =      9.3.240.58 >  (server3.itsc.austin.ibm.com)
IP:     ip_v=4, ip_hl=20, ip_tos=0, ip_len=84, ip_id=6898, ip_off=0
IP:     ip_ttl=30, ip_sum=8f2e, ip_p = 17 (UDP)
UDP:    <source port=830, <destination port=111(sunrpc) >
UDP:    [ udp length = 64 | udp checksum = 54ca ]
RPC: **CALL**   XID=961637844
RPC: Program=100000 (PMAPPROG) Version=2 Procedure=3 (PMAPPROC_GETPORT)
RPC: AUTH_NULL Opaque Authorization Base 0 Opaque Authorization Length 0
PMP: Prog=100005 Vers=3 Prot=6 Port=0

Packet Number 2
TOK: ====( 78 bytes received on interface tr0 )==== 14:30:59.084636275
TOK: 802.5 packet
TOK: 802.5 MAC header:
TOK: access control field = 18, frame control field = 40
TOK: [ src = 00:06:29:be:d2:a2, dst = 00:06:29:be:b1:dc]
TOK: 802.2 LLC header:
TOK: dsap aa, ssap aa, ctrl 3, proto 0:0:0, type 800 (IP)
IP:     < SRC =      9.3.240.58 >  (server3.itsc.austin.ibm.com)
IP:     < DST =      9.3.240.56 >  (server1.itsc.austin.ibm.com)
```

```
IP:     ip_v=4, ip_hl=20, ip_tos=0, ip_len=56, ip_id=42767, ip_off=0
IP:     ip_ttl=255, ip_sum=223c, ip_p = 1 (ICMP)
ICMP:   icmp_type=3 (DEST UNREACH)
ICMP:   icmp_code=3 (9.3.240.58: UDP PORT 111 unreachable, src=830)
```

To fix this problem, the right order of starting services should be followed:

1. Stop the NFS daemons on server.

   This might result in a situation where rpc.lockd and nfsd stay in a STOPPING status. If this happens, restart the statd daemon, stop the lockd daemons, and finally stop the statd daemon. Check the status with the **`lssrc -g nfs`** command. This should also take care of the unresponsive nfsd. If this did not help, unmount all clients and repeat the procedure.

2. Start portmap.
3. Start NFS daemons on server.

nfsd problem determination

In the next scenario, the nfsd daemon is stopped at the NFS server. When trying to mount the test file system from the server, the **`mount`** command hangs with the following error message:

```
# mount server3:/tmp/thomasc/testfs /tmp/server3mnt
mount: 1831-010 server server3 not respondingmount: retrying
server3:/tmp/thomasc/testfs
```

When looking at the iptrace output of this event, the client uses the RPC PMAPPROC_GETPORT to connect to 100003, which, as earlier mentioned, is nfsd. The output shows PMP returning a value of 0. This RPC is defined at the following URL:

http://www.opengroup.org/onlinepubs/9629799/toc.htm

The description tells you that if the port value is zero, as in this example, the program called is not registered. Again the importance of portmap is shown.

```
Packet Number 24
TOK: ====( 106 bytes transmitted on interface tr0 )==== 15:30:00.808241654
TOK: 802.5 packet
TOK: 802.5 MAC header:
TOK: access control field = 0, frame control field = 40
TOK: [ src = 00:06:29:be:b1:dc, dst = 00:06:29:be:d2:a2]
TOK: 802.2 LLC header:
TOK: dsap aa, ssap aa, ctrl 3, proto 0:0:0, type 800 (IP)
IP:     < SRC =      9.3.240.56 >  (server1.itsc.austin.ibm.com)
IP:     < DST =      9.3.240.58 >  (server3.itsc.austin.ibm.com)
IP:     ip_v=4, ip_hl=20, ip_tos=0, ip_len=84, ip_id=8397, ip_off=0
IP:     ip_ttl=30, ip_sum=8953, ip_p = 17 (UDP)
UDP:    <source port=683, <destination port=111(sunrpc) >
```

```
UDP:    [ udp length = 64 | udp checksum = 6bda ]
RPC: **CALL**   XID=962484044
RPC: Program=100000 (PMAPPROG) Version=2 Procedure=3 (PMAPPROC_GETPORT)
RPC: AUTH_NULL Opaque Authorization Base 0 Opaque Authorization Length 0
PMP: Prog=100003 Vers=3 Prot=6 Port=0

Packet Number 25
TOK: ====( 78 bytes received on interface tr0 )==== 15:30:00.809164951
TOK: 802.5 packet
TOK: 802.5 MAC header:
TOK: access control field = 18, frame control field = 40
TOK: [ src = 00:06:29:be:d2:a2, dst = 00:06:29:be:b1:dc]
TOK: 802.2 LLC header:
TOK: dsap aa, ssap aa, ctrl 3, proto 0:0:0, type 800 (IP)
IP:     < SRC =      9.3.240.58 >  (server3.itsc.austin.ibm.com)
IP:     < DST =      9.3.240.56 >  (server1.itsc.austin.ibm.com)
IP:     ip_v=4, ip_hl=20, ip_tos=0, ip_len=56, ip_id=44783, ip_off=0
IP:     ip_ttl=30, ip_sum=fb4c, ip_p = 17 (UDP)
UDP:    <source port=111(sunrpc), <destination port=683 >
UDP:    [ udp length = 36 | udp checksum = 7967 ]
RPC: **REPLY**  XID=962484044
RPC: 100000(PMAPPROG) 3(PMAPPROC_GETPORT)
RPC: Reply Stat: MSG_ACCEPTED
RPC: Accepted Reply Stat:  SUCCESS
PMP: Returning 0
```

That is what a mount attempt would look like if nfsd is down on the server. Take a look at how an unresponsive nfsd daemon influences a client with a mounted file system.

When issuing a long listing of an NFS mounted file system, a biod-to-nfsd interaction is requested. This will result in a command hang, with the following error message:

```
# pwd
/tmp/server3ro
# ls -la
NFS server server3 not responding still trying
```

This problem is solved by starting the nfsd. As long as the portmap daemon was active and the nfsd can register with the portmap daemon, no further actions need to be taken.

rpc.mountd problem determination

If the rpc.mountd at the server does not answer to mount requests, there are some points to remember.

When trying to mount a file system from the server, **iptrace** shows that the server responds with port unreachable, just as expected. More interesting is what happens when an unmount of an existing mount is issued from a client, which would be the normal scenario at a client reboot (as an example).

The iptrace from the client shows the portmap has a port registered for rpc.mountd, which portmap communicates to the client. The client calls program 100005 (rpc.mountd) on the assigned port, but receives a port unreachable message.

```
Packet Number 3
TOK: ====( 166 bytes transmitted on interface tr0 )==== 08:20:39.765724065
TOK: 802.5 packet
TOK: 802.5 MAC header:
TOK: access control field = 0, frame control field = 40
TOK: [ src = 00:04:ac:61:73:f7, dst = 00:06:29:be:d2:a2]
TOK: 802.2 LLC header:
TOK: dsap aa, ssap aa, ctrl 3, proto 0:0:0, type 800 (IP)
IP:     < SRC =      9.3.240.59 >  (server4f.itsc.austin.ibm.com)
IP:     < DST =      9.3.240.58 >  (server3.itsc.austin.ibm.com)
IP:     ip_v=4, ip_hl=20, ip_tos=0, ip_len=144, ip_id=40703, ip_off=0
IP:     ip_ttl=30, ip_sum=ae2, ip_p = 17 (UDP)
UDP:    <source port=946, <destination port=38637 >
UDP:    [ udp length = 124 | udp checksum = 6523 ]
RPC: **CALL**   XID=962260761
RPC: Program=100005 (MOUNTPROG) Version=1 Procedure=3 (MOUNTPROC_UMNT)
RPC: AUTH_UNIX
RPC: Cred:
RPC:    Time=0x395212a7 (Thu Jun 22 08:20:39 2000)
RPC:    Machine=server4 Uid=0 Gid=0 Group List Length=6
RPC:    Groups= ( 0 2 3 7 8 10 )
MNT: Path: /tmp/thomasc/testfs

Packet Number 4
TOK: ====( 78 bytes received on interface tr0 )==== 08:20:39.766378665
TOK: 802.5 packet
TOK: 802.5 MAC header:
TOK: access control field = 18, frame control field = 40
TOK: [ src = 00:06:29:be:d2:a2, dst = 00:04:ac:61:73:f7]
TOK: 802.2 LLC header:
TOK: dsap aa, ssap aa, ctrl 3, proto 0:0:0, type 800 (IP)
IP:     < SRC =      9.3.240.58 >  (server3.itsc.austin.ibm.com)
IP:     < DST =      9.3.240.59 >  (server4f.itsc.austin.ibm.com)
IP:     ip_v=4, ip_hl=20, ip_tos=0, ip_len=56, ip_id=58893, ip_off=0
IP:     ip_ttl=255, ip_sum=e33a, ip_p = 1 (ICMP)
ICMP:   icmp_type=3 (DEST UNREACH)
ICMP:   icmp_code=3 (9.3.240.58: UDP PORT 38637 unreachable, src=946)
```

At the client, the error messages `Warning: unmount:: RPC: 1832-008 Timed out` would appear.

```
# mount
  node    mounted      mounted over    vfs   date            options
-------- -----------  -------------   ----  ----------      ----------
         /dev/hd4     /               jfs   Jun 11 16:46    rw,log=/dev/hd8
         /dev/hd2     /usr            jfs   Jun 11 16:46    rw,log=/dev/hd8
         /dev/hd9var  /var            jfs   Jun 11 16:47    rw,log=/dev/hd8
         /dev/hd3     /tmp            jfs   Jun 11 16:47    rw,log=/dev/hd8
         /dev/hd1     /home           jfs   Jun 11 16:47    rw,log=/dev/hd8
server3  /tmp/thomasc/testfs /tmp/server3ro   nfs3   Jun 22 08:37
# umount /tmp/server3ro
Warning: unmount:: RPC: 1832-008 Timed out
```

When checking the mount points of the client, /tmp/thomasc/testfs is no longer mounted.

```
# mount
  node    mounted      mounted over    vfs   date            options
--------  ----------   ------------    ----  ------------    ------------
          /dev/hd4     /               jfs   Jun 11 16:46    rw,log=/dev/hd8
          /dev/hd2     /usr            jfs   Jun 11 16:46    rw,log=/dev/hd8
          /dev/hd9var  /var            jfs   Jun 11 16:47    rw,log=/dev/hd8
          /dev/hd3     /tmp            jfs   Jun 11 16:47    rw,log=/dev/hd8
          /dev/hd1     /home           jfs   Jun 11 16:47    rw,log=/dev/hd8
```

The unmount was successful from a client point of view, but at the server rpc.mountd keeps track of its clients in the /etc/rmtab file as mentioned earlier. This file will not be up to date after such a scenario occurs. It will still tell the server NFS subsystem that a file system is exported to server4.

Under normal circumstances, the unmount would communicate with rpc.mountd on the server, and the rpc.mountd would update the /etc/rmtab file by commenting out the entry for the export (exchanging the first letter with a #). For example:

```
# more /etc/rmtab
#erver4f.itsc.austin.ibm.com:/tmp/thomasc/testfs
```

7.3.3 Server performance

When narrowing down the performance discussion on servers to NFS specifics, the issue is often related to dropped packets. NFS servers may sometimes drop packets due to overload.

One common place where a server will drop packets is the UDP socket buffer. Remember that the default for data transfer for AIX Version 4.3 is TCP, but UDP is still used for mounting and GETPORT calls. Dropped packets here are counted by the UDP layer and the statistics can be seen by using the **netstat -p UDP** command. For example:

```
# netstat -p UDP
udp:
        89827 datagrams received
        0 incomplete headers
        0 bad data length fields
        0 bad checksums
        329 dropped due to no socket
        77515 broadcast/multicast datagrams dropped due to no socket
        0 socket buffer overflows
        11983 delivered
        11663 datagrams output
(At the testsystem the buffer size was sufficient)
```

NFS packets will usually be dropped at the socket buffer only when a server has a lot of NFS write traffic. The NFS server uses UDP and TCP sockets attached to the NFS port, and all incoming data is buffered on those ports. The default size of this buffer is 60000 bytes. Doing some quick math by dividing that number by the size of the default NFS Version 3 write packet (32765), you find that it will take only two simultaneous write packets to overflow that buffer. That could be done by just one NFS client (with the default configurations). Practically speaking, however, it is not as easy as it sounds to overflow the buffer. As soon as the first packet hits the socket, an nfsd will be awakened to start taking the data off.

One of two things has to happen. There is either high volume or high burst traffic on the socket. If there is high volume, a mixture of lots of writes, or other possibly non-write NFS traffic, there may not be enough nfsds to take the data off the socket fast enough to keep up with the volume (recall that it takes a dedicated nfsd to service each NFS call of any type). In the high burst case, there may be enough nfsds, but the speed at which packets arrive on the socket is such that the nfsd daemons cannot wake up fast enough to keep it from overflowing.

Each of the two situations has a different method to handle it. In the case of high volume, it may be sufficient to just increase the number of nfsds running on the system. Since there is no significant penalty for running extra nfsds on an AIX machine, this should be tried first.

This can be done with the following command:

```
# chnfs -n 16
```

This stops the currently running daemons, modifies the SRC database code to reflect the new number, and restarts the daemons indicated.

In the case of high burst traffic, the only solution is to make the socket bigger in the hope that some reasonable size will be sufficiently large enough to give the nfsds time catch up with the burst. Memory dedicated to this socket will not be available for any other use, so it must be noted that making the socket larger may result in that memory being underutilized the vast majority of the time. The cautious administrator will watch the socket buffer overflows statistic and correlate it with performance problems and make a determination of how big to make the socket buffer. To check the NFS kernel options, use the **nfso** command:

```
# nfso -a
portcheck= 0
udpchecksum= 1
nfs_socketsize= 60000
nfs_tcp_socketsize= 60000
nfs_setattr_error= 0
nfs_gather_threshold= 4096
nfs_repeat_messages= 0
nfs_udp_duplicate_cache_size= 0
nfs_tcp_duplicate_cache_size= 5000
nfs_server_base_priority= 0
nfs_dynamic_retrans= 1
nfs_iopace_pages= 0
nfs_max_connections= 0
nfs_max_threads= 8
nfs_use_reserved_ports= 0
nfs_device_specific_bufs= 1
nfs_server_clread= 1
nfs_rfc1323= 0
nfs_max_write_size= 0
nfs_max_read_size= 0
nfs_allow_all_signals= 0
```

If you change the nfsbuffer sizes, you must verify that the kernel variable sb_max is greater then the NFS buffer values chosen. The default value of sb_max is 1048576 on AIX Version 4.3.3 and later. If you need to increase the sb_max value. This can be done with the **no** command. Remember that everything changed with **no** or **nfso** is valid only until the next boot (if these changes have been added to some startup script, for example, /etc/rc.nfs).

7.4 NFS client considerations

There are a couple of things to consider when looking at the clients in an NFS environment. The first is mount problems, the second is what options should be used when mounting, and finally, performance issues.

7.4.1 Client-side mount problem determination

The first issue to be covered is the problems with mounting file systems, directories or files. Except for the problems discussed in “Portmap problem determination” on page 160, “nfsd problem determination” on page 162, and “rpc.mountd problem determination” on page 163, and the way to use **iptrace** shown in those examples, there is not really much to do on the client side. A simple checklist can help you with most problems:

- Check if the file system you try to mount is exported.

  When a mount request is sent to a server for an export that does not exist, the following error message appears:

  ```
  # mount server3:/usr/welcome /tmp/server3mnt
  mount: 1831-011 access denied for server3:/usr/welcome
  mount: 1831-008 giving up on:
  server3:/usr/welcome
  The file access permissions do not allow the specified action.
  ```

  To check what file systems, directories, or files are exported from a server, use the **showmount** command as follows:

  ```
  # showmount -e <server>
  ```

  The output from the command shows you the directories exported, and to whom they are exported, as discussed in “The connection between /etc/exports, exportfs, and /etc/xtab” on page 158.

- If the server does not answer to a **showmount -e** call (which communicates with rpc.mountd), check if the RPC servers are registered with the portmap daemon, as follows:

  ```
  # rpcinfo -p server3 (the command issued on server4; edited output)
  program   vers  proto  port     service
  100000    4     tcp    111      portmapper
  100000    3     tcp    111      portmapper
  100000    2     tcp    111      portmapper
  100000    4     udp    111      portmapper
  100000    3     udp    111      portmapper
  100000    2     udp    111      portmapper
  100024    1     udp    660      status
  100024    1     tcp    654      status
  100021    1     udp    38624    nlockmgr
  100021    2     udp    38624    nlockmgr
  100021    3     udp    38624    nlockmgr
  100021    4     udp    38624    nlockmgr
  100021    1     tcp    37693    nlockmgr
  100021    2     tcp    37693    nlockmgr
  100021    3     tcp    37693    nlockmgr
  100021    4     tcp    37693    nlockmgr
  ```

```
100003    2      udp    2049      nfs
100003    3      udp    2049      nfs
100003    2      tcp    2049      nfs
100003    3      tcp    2049      nfs
100005    1      udp    40212     mountd
100005    2      udp    40212     mountd
100005    3      udp    40212     mountd
100005    1      tcp    38422     mountd
100005    2      tcp    38422     mountd
100005    3      tcp    38422     mountd
```

The output shows that the portmap (program 100000) is available, so is statd (100024), lockd (100021), nfsd (100003), and mountd (100005).

If the RPC programs are up and running but you still do not have any answer on **showmount -e**, then you probably tried to mount a file system from a host that is not configured as a server.

- Check the syntax on the **mount** command. Also remember that only root can issue any **mount** command, and system group members can issue mounts, provided they have write access to the mount point.

  To mount the file system that has been used in the previous examples, from server3 on /tmp/thomasc/server3ro, issue the following command on server4:

  ```
  # mount server3:/tmp/thomasc/testfs /tmp/thomasc/server3ro
  ```

  You can also use **smitty mknfsmnt**, as shown in Figure 7-6:

```
                        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                             [/tmp/server3ro]          /
* PATHNAME of remote directory                        [/tmp/thomasc/testfs]
* HOST where remote directory resides                 [server3]
  Mount type NAME                                     []
* Use SECURE mount option?                            no                        +
* MOUNT now, add entry to /etc/filesystems or both?   both                      +
* /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                              []                        #
[MORE...26]

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure 7-6 smitty mknfsmnt screen

When using **smitty**, the option to edit /etc/filesystems is available (highlighted). By editing /etc/filesystems, the only thing to do when mounting an NFS file system is to issue the **mount** command with the local mount point as an argument. For example:

```
# mount /tmp/server3ro
# mount
  node    mounted         mounted over    vfs   options
-------- -------------    ------------    ----  ------------
          /dev/hd4        /               jfs   rw,log=/dev/hd8
          /dev/hd2        /usr            jfs   rw,log=/dev/hd8
          /dev/hd9var     /var            jfs   rw,log=/dev/hd8
          /dev/hd3        /tmp            jfs   rw,log=/dev/hd8
          /dev/hd1        /home           jfs   rw,log=/dev/hd8
server3  /tmp/thomasc/testfs /tmp/server3ro   nfs3 bg,hard,intr
(The output is edited to fit the screen)
```

The stanza format of /etc/filesystems is easy to comprehend. The entry for the file system in our examples appears as follows:

```
/tmp/server3ro:
        dev             = "/tmp/thomasc/testfs"
        vfs             = nfs
        nodename        = server3
        mount           = false
        options         = bg,hard,intr
        account         = false
        type            = thomasc
```

These options are covered in 7.4.2, “Client mount options” on page 171. In the stanza, you can see the **mount** has a value of false. The **mount** command uses the associated values. It recognizes five values for the mount attributes: automatic, true, false, removable, and readonly. Automatic means that the file system is to be mounted at boot; this is usually used for system-defined file systems. A value of true means that the **mount all** is allowed to mount this file system. Finally the value of false means that the mount will only occur when the file system is specified as an argument to the **mount** command, or the type is used for mount.

The type = value is a nice feature with the **mount** command. By defining type to a common value for several file systems, all these file systems can be mounted by giving the value as an argument to the -t flag. For example:

```
# mount -t thomasc
```

Note: It is recommended that you use empty directories as mount points.

If a file system is mounted on a directory in use, the file names and their i-node pointer will be hidden. Access is lost by using this method.

The only way to access these hidden files is to unmount the file system.

7.4.2 Client mount options

There are several useful options when considering and planning for an NFS mount. The one specific for smitty, update of /etc/filesystems, was covered in 7.4.1, “Client-side mount problem determination” on page 168.

The most common issue is whether to use a *hard* mount or a *soft* mount. A soft mount will try to re-transmit a number of times. This re-transmit value is defined by the retrans option. After the set number of retransmissions has been used, the soft mount gives up and returns an error.

A hard mount retries a request until a server responds. The hard option is the default value. On hard mounts, the intr option should be used to allow a user to interrupt a system call that is waiting on a crashed server.

Both hard mounts and soft mounts use the timeo option, to calculate the time between re-transmits. The default value is 0.7 seconds for the first timeout. After that, it increases the timeout exponentially until a maximum of 30 seconds, where it stabilizes until a reply is received. Depending on the value set for the retrans option, the soft mount has probably given up already at this stage. When discussing timeouts and hard mounts, you should choose between two other mount options, *proto* TCP or UDP.

When using UDP, it is important to understand that if a write or read packet is lost on the network or dropped at the server, the full timeout interval will expire before the packet is retransmitted from the client. Using UDP, there is no intermediate-ack mechanism that would inform the client, for example, that the server only received five of the expected six write fragment packets.

The reliable delivery mechanisms built into TCP will help maintain good performance in networks where the unreliable UDP transport fails. The reason is that TCP uses a packet-level delivery acknowledgment mechanism that keeps fragments from being lost. Recall that lost fragments using UDP require re-sending the entire read or write request after a timeout expires. TCP avoids this by guaranteeing delivery of the request.

Finally, there is the choice of mounting in the background (bg) or in the foreground (fg). If bg is defined and an NFS server does not answer a mount

request, then another mount process will start in the background and keep trying to establish the mount. By this method, the mount process is free to process another mount request. Define bg in the /etc/filesystems file when establishing a predefined mount that will be mounted during system startup. Mounts that are non-interruptible and running in the foreground can hang the client if the network or server is down when the client system starts up. If a client cannot access the network or server, the user must start the machine again in maintenance mode and edit the appropriate mount requests.

This applies to the default mount options, which are TCP, NFS Version 3, and hard mount in the background (on test system running 4.3.3, but the documentation at the time of publication states that fg is default).

7.4.3 Client performance considerations

A client performance discussion often concentrates on the number of biods used. For biod daemons, there is a default number of biods (six for a V2 mount, four for a V3 mount) that may operate on any one remote mounted file system at one time. The idea behind this limitation is that allowing more than a set number of biods to operate against the server at one time may overload the server. Since this is configurable on a per-mount basis on the client, adjustments can be made to configure client mounts by the server capabilities.

When evaluating how many biods to run, you should consider the server capabilities as well as the typical NFS usage on the client machine. If there are multiple users or multiple process on the client that will need to perform NFS operations to the same NFS mounted file systems, you have to be aware that contention for biod services can occur with just two simultaneous read or write operations.

Since up to six biods can be working on reading a file in one NFS file system, if another read starts in another NFS mounted file system, both reads will be attempting to use all six biods. In this case, presuming that the server(s) are not already overloaded, performance will likely improve by increasing the biod number to 12. This can be done using the **chnfs** command:

```
# chnfs -b 12
```

On the other hand, suppose both file systems are mounted from the same server and the server is already operating at peak capacity. Adding another six biods could actually decrease the response dramatically due to the server dropping packets and resulting in timeouts and retransmits.

Tuning the numbers of nfsd and biod daemons

After you have arrived at an initial number of biod and nfsd daemons, or have changed one or the other, the following steps will assist you in fine tuning your system.

First, recheck the affected systems for CPU or I/O saturation with the **vmstat** and **iostat** commands. If the server is now saturated, you must reduce its load or increase its power, or both.

Use the **netstat -s** command to determine if any system is experiencing UDP socket buffer overflows. If so, use the **no -a** command to verify the UDP settings have been implemented. If so, and the system is not saturated, increase the number of biod or nfsd daemons.

Examine the nullrecv column in the **nfsstat -s** command output. If the number starts to grow, it may mean there are too many nfsd daemons. However, this is less likely on this operating system's NFS servers than it is on other platforms. The reason for that is that all nfsd daemons are not awakened at the same time when an NFS request comes into the server. Instead, the first nfsd daemon wakes up, and if there is more work to do, this daemon wakes up the second nfsd daemon, and so on.

To change the number of nfsd daemons, you can use the **chnfs** command, or set the nfso nfs_max_threads parameter as mentioned earlier.

To change the number of nfsd daemons on a server to 10, both immediately and at each subsequent system boot, use the following:

```
# chnfs -n 10
```

To change the number of nfsd daemons on a system to 9, with the change delayed until the next system boot, run the following command:

```
# chnfs -I -n 9
```

To change the number of biod daemons per mount, use the biod mount option.

Increasing the number of biod daemons on the client worsens server performance because it allows the client to send more request at once, further loading the network and the server. In extreme cases of a client overrunning the server, it may be necessary to reduce the client to one biod daemon, as follows:

```
# stopsrc -s biod
```

This leaves the client with the kernel process biod still running.

There are also some mount options that may improve the performance on the client. The most useful options are used to set the read and write sizes to some value that changes the read/write packet size that is sent to the server.

For NFS Version 3 mounts, the read/write sizes can be both increased and decreased. The default read/write sizes are 32 KB. The maximum possible on AIX at the time of publication is 61440 bytes (60 x 1024). Using 60 KB read/write sizes may provide slight performance improvement in specialized environments. To increase the read/write sizes when both server and client are AIX machines requires modifying settings on both machines. On the client, the mount must be performed setting up the read/write sizes with the -o option (for example, `-o rsize=61440,wsize=61440`). On the server, the advertised maximum read/write size is configured through use of the **nfso** command using the nfs_max_write_size and nfs_max_read_size parameters. For example:

```
# nfso -o nfs_max_write_size=61440
```

The **nfsstat** command displays statistical information about the NFS and RPC interfaces to the kernel. You can also use this command to reinitialize this information. If no flags are given, the default is the **nfsstat -csnr** command. For example, to display statistics for each NFS mounted file system, enter:

```
# nfsstat -m
/mnt from /mnt:ut.austin.ibm.com
 Flags:
vers=3,proto=tcp,auth=unix,hard,intr,link,symlink,rsize=32768,wsize=32
768,retrans=5
 All:     srtt=0 (0ms), dev=0 (0ms), cur=0 (0ms)
```

7.5 Automount

Automount is used for automatic and transparent mounting and unmounting of file systems. Automount monitors specify directory mount points. When a file I/O operation is requested to that mount point, the automountd daemon performs the RPC call (or the system call) to complete the mount. Any directories that do not already exist on the client will be created. AIX Version 4.3.1 and earlier used a daemon called automount, and in AIX 4.3.2 the AutoFS is used for automount. AutoFS provides automatic mount of many types of file systems, for example CDRFS and JFS. The daemon in AutoFS is called automountd. In AIX Version 4.3.2 and later, **automount** is just a command, not a daemon.

As discussed, AutoFS allows file systems to be mounted as needed. With this method of mounting directories, all file systems do not need to be mounted all of the time; only those being used are mounted.

For example, to mount an NFS directory automatically, first check that the server has exported the directory by using the **showmount** command:

```
# showmount -e server3
export list for server3:
/tmp/thomasc        server1,server2,server4
/tmp/thomasc/testfs (everyone)
/home               (everyone)
```

Then create an AutoFS map file. Any file name can be used although it is a good idea to define if an indirect map or a direct map is used. The **automount** command is used as an administration tool for AutoFS. It installs AutoFS mount points and associates an automount map with each mount point. The AutoFS file system monitors attempts to access directories within it and notifies the automountd daemon. The daemon uses the map to locate a file system, which it then mounts at the point of reference within the AutoFS file system. The syntax for the **automount** command is:

```
/usr/sbin/automount [  -v ] [  -t Duration ] [  -i Interval ]
```

Some useful **automount** flags are provided in Table 7-2.

Table 7-2 Commonly used flags of the automount command

Flags	Description
-t *Duration*	Specify a duration, in seconds, that an AutoFS unmount thread sleeps before it starts to work again. The default timeout is five minutes.
-i *Interval*	Specifies an interval, in seconds, that an AutoFS automounted directory lives.
-v	Displays on standard output verbose status and warning messages. Supported for both implementation.

7.5.1 Indirect maps

In this section, how to use indirect maps is discussed.

Start by editing a file to look like the example file mount.indirect.map. Because this is a configuration file, it is usually placed in the /etc file system, but in the examples below the /tmp directory is used. Start with defining the mount point to be used by automountd. Then define the options (if such are needed) and finally enter the path to the server directory, just like a normal mount.

```
# more mount.indirect.map
S3testfs        -rw     server3:/tmp/thomasc/testfs
```

Then start the automountd with:

```
# startsrc -s automountd
0513-059 The automountd Subsystem has been started. Subsystem PID is 22574.
```

At this stage, you can see that the only thing that has been done is editing a file and starting a daemon. To make this work, you have to define for the **automount** command where the parent directory is for the AutoFS mount point directory (S3testfs), defined in the mount.indirect.map file. This is done in the following way:

```
# automount -m /tmp/thomasc /tmp/mount.indirect.map
```

NIS is sometimes used to propagate map files to NFS clients. The -m flag tells the automount facility not to use NIS. The automount daemon, by default, reads the /etc/auto.master map to find which directories to watch for mounts. The auto.master map has the following format:

DirectoryPath AutomountMapName

The AutomountMapName field specifies a file containing the automount map for the directory specified by the DirectoryPath field. For example, the contents of the /etc/auto.master file on the server might be as follows:

```
/home/home      /etc/auto.home
/usr/lpp        /etc/auto.direct
```

The auto.master file entries direct the automount daemon to use the /etc/auto.home automount map for the /home/home directory and the /etc/auto.direct automount map for the /usr/lpp directory.

In this example, the /etc/auto.home and /etc/auto.direct were local files on the client that contained all of the automount map needed. The contents of the automount maps can also be maintained by NIS. The files would still exist on the client, but the contents would be different. For example, the /etc/auto.home file would contain the following:

```
+auto.home
```

And the /etc/auto.direct file would contain the following:

```
+auto.direct
```

This directs the automount daemon to consult the NIS maps auto.home and auto.direct when it reads the local files. The NIS server would contain two new NIS maps. The maps would be auto.home and auto.direct. They would be added to the /var/yp/Makefile in the same way that the auto.master NIS map was added. This makes them available for use by the NIS clients running the automount daemon. See Chapter 10, “NIS” on page 223 for further details on NIS.

After the initiation of the automount facility, there is an entry in the mount table that tells us that automountd will look at the entries in /tmp/mount.indirect.map for reference when creating mount points under the parent directory /tmp/thomasc. (The mount point will be /tmp/thomasc/S3testfs).

```
# mount
  node    mounted      mounted   vfs    date            options
                       over
--------  ----------   -------   ----   --------        ------------
          /dev/hd4     /         jfs    Jun 11 16:46    rw,log=/dev/hd8
          /dev/hd2     /usr      jfs    Jun 11 16:46    rw,log=/dev/hd8
          /dev/hd9var  /var      jfs    Jun 11 16:47    rw,log=/dev/hd8
          /dev/hd3     /tmp      jfs    Jun 11 16:47    rw,log=/dev/hd8
          /dev/hd1     /home     jfs    Jun 11 16:47    rw,log=/dev/hd8
         /tmp/mount.indirect.map /tmp/thomasc     autofs rw,ignore
(the output has been edited to fit the screen - the timestamp is removed)
```

When issuing a long listing of the content of /tmp/thomasc there will, at this point, be no entries, because the mount point to monitor is S3testfs.

```
# ls -la
total 536873840
dr-xr-xr-x   2 root     system         2 Jun 22 14:33 .
drwxrwxrwt  18 bin      bin         1024 Jun 22 14:12 ..
```

When issuing a long listing of one of the mount points, the mount will occur, as well as the creation of the mount point.

```
# ls -la S3testfs
total 537196352
drwxr-sr-x  12 thomasc  staff        512 Jun 22 11:03 .
dr-xr-xr-x   2 root     system         3 Jun 22 14:34 ..
drwxr-xr-x   3 root     sys          512 Jun 19 15:53 dumpfmt
drwxr-xr-x   2 root     sys          512 Jun 19 15:53 findcore
```

The mount point will only exist as long as the mount is valid. As mentioned before, the automount facility also handles the unmount of the file systems. The activity in the file system defines when the unmount will occur. If nobody uses the file system (no process uses the directory as $PWD), two timeout values are used.

The first one, -tl (time to live), defines the time in seconds that the automountd should wait before attempting to unmount a quiescent file system. The default value is 300 seconds.

The other one, -tw (time to wait), defines the number of seconds to wait before the daemon retries to unmount the file system in the previous unmount attempt was unsuccessful. The default is 60 seconds.

To change these timeout values, use the flags with the **automount** command. For example:

```
# automount -m -tl 600 -tw 300 /tmp/mount.indirect.map /tmp/thomasc
```

In the mount table, the actual mount will appear:

```
# mount
  node    mounted      mounted   vfs    date            options
                       over
--------  ----------   --------  ----   ----------      ---------------
          /dev/hd4     /         jfs    Jun 11 16:46    rw,log=/dev/hd8
          /dev/hd2     /usr      jfs    Jun 11 16:46    rw,log=/dev/hd8
          /dev/hd9var  /var      jfs    Jun 11 16:47    rw,log=/dev/hd8
          /dev/hd3     /tmp      jfs    Jun 11 16:47    rw,log=/dev/hd8
          /dev/hd1     /home     jfs    Jun 11 16:47    rw,log=/dev/hd8
         /tmp/mount.indirect.map /tmp/thomasc     autofs rw,ignore
server3  /tmp/thomasc/testfs /tmp/thomasc/S3testfs nfs3   Jun 22 14:34 rw
```

In the preceding example, an indirect map file is used. As seen in the map file (/tmp/mount.indirect.map), the mount points are defined with relative paths. This provides the administrator the opportunity to use another parent directory.

7.5.2 Direct maps

The other map file used with automount is a direct map file. In the direct map file, the absolute path to the mount point is defined (in the following example, /tmp/thomasc and /home/remote):

```
# more /tmp/mount.direct.map
/home/remote    server3:/home
```

The initiation of the **mount** command differs from the indirect automount in the sense that you do not need to point out the parent directory that is specified in the direct map. This is defined by the use on /-. The mount point will also be created at this point, if it did not already exist. To initiate the automount with a direct map (auto.direct.map), use the following command:

```
# automount -m /- /tmp/mount.direct.map
```

When using direct maps, the mount table will appear slightly different. Instead of pointing out one file that has the mount points defined, one mount point definition is defined in the mount list for each entry in the direct map:

```
# mount
  node    mounted      mounted   vfs    date            options
                       over
-------- --------      -------   ----   ------------    ---------------
          /dev/hd4     /         jfs    Jun 11 16:46    rw,log=/dev/hd8
```

```
         /dev/hd2     /usr      jfs    Jun 11 16:46    rw,log=/dev/hd8
         /dev/hd9var  /var      jfs    Jun 11 16:47    rw,log=/dev/hd8
         /dev/hd3     /tmp      jfs    Jun 11 16:47    rw,log=/dev/hd8
         /dev/hd1     /home     jfs    Jun 11 16:47    rw,log=/dev/hd8
      /tmp/mount.direct.map /home/remote    autofs Jun 22 15:02 rw,ignore
```

This does not mean that the actual mount has occurred. The actual mount request will be sent to the server when the mount point is used. The output in the mount table will then also show a mount point for the actual mount point:

```
# mount
  node    mounted         mounted   vfs       date            options
                          over
-------- --------         --------  ----      ------------    ---------------
         /dev/hd4         /         jfs       Jun 11 16:46    rw,log=/dev/hd8
         /dev/hd2         /usr      jfs       Jun 11 16:46    rw,log=/dev/hd8
         /dev/hd9var      /var      jfs       Jun 11 16:47    rw,log=/dev/hd8
         /dev/hd3         /tmp      jfs       Jun 11 16:47    rw,log=/dev/hd8
         /dev/hd1         /home     jfs       Jun 11 16:47    rw,log=/dev/hd8
      /tmp/auto.direct.map /home/remote     autofs Jun 22 15:11 rw,ignore
server3  /home            /home/remote     nfs3   Jun 22 15:18 rw
```

7.5.3 Auto.master map

In the previous examples, the **automount** commands are used with arguments (the map files), but when initiated without arguments, automount consults the master map for a list of AutoFS mount points and their maps. This gives you an easy way to start several map files, both indirect and direct, at the same time. This file can be called /etc/auto.master or /etc/auto_master.

The syntax of the auto.master file is intuitive. Just point out the parent directory for indirect automounts, and point out that with special flag /- that the direct map includes the absolute path. The two mapfiles used in the previous examples will be used in the following example:

```
# more /etc/auto.master
/tmp/thomasc            /tmp/mount.indirect.map
/-                      /tmp/mount.direct.map
```

Because the syntax for the indirect and the direct maps are included in the auto.master, you only need to tell the **automount** command which file to read. For example:

```
# automount -m -f auto.master

# mount
  node       mounted        mounted over    vfs       date        options
-------- ---------------  ---------------  ------ ------------
         /dev/hd4         /                jfs    Jun 11 16:46 rw,log=/dev/hd8
```

```
  /dev/hd2         /usr            jfs    Jun 11 16:46 rw,log=/dev/hd8
  /dev/hd9var      /var            jfs    Jun 11 16:47 rw,log=/dev/hd8
  /dev/hd3         /tmp            jfs    Jun 11 16:47 rw,log=/dev/hd8
  /dev/hd1         /home           jfs    Jun 11 16:47 rw,log=/dev/hd8
/tmp/mount.indirect.map /tmp/thomasc    autofs Jun 22 16:21 ignore
  /tmp/mount.direct.map /home/remote     autofs Jun 22 16:21 ignore
```

The mount table appears as expected. The access of the mount point directories will next initiate the actual mount as defined in the indirect and the direct map file.

7.6 Summary

NFS is used for transparent mount of remote file systems.

7.6.1 Protocols

NFS can use UDP or TCP on the transport layer.

NFS uses XDR for interpreting data representation between different hardware architectures.

NFS uses RPC for transparent remote execution of calls.

7.6.2 Daemons

The portmap registers all NFS daemons.

- **rpcinfo -p** is used to check what programs are registered.

The rpc.mountd handles mount requests on the server.

- It uses /etc/xtab to verify exports.
- **showmount -a** shows exports.
- **showmount -e <server>** shows what file systems are exported.

Nfsd on the server answers all client requests, except mount requests.

- By default, eight nfs daemons are started from /etc/rc.nfs.

Biod handles all write and read requests at the client side.

- Up to six biods can work on one mount point.

Rc.lockd and rc.statd handles locking requests and information.

7.6.3 Files

The /etc/exports file is edited with file systems to be exported.

The /etc/xtab files is generated with the **exportfs** command and is used by the rpc.mountd at mount requests.

The /etc/rmtab file has records of active exports.

A list of server names and their corresponding RPC program number is in /etc/rpc.

7.7 Command summary

The following section provides a list of the key commands discussed in this chapter. For a complete reference of the following commands, consult the AIX product documentation.

7.7.1 The showmount command

The **showmount** command displays a list of all clients that have remotely mounted file systems.

The syntax of the **showmount** command is:

```
showmount [ -a ] [ -d ] [ -e ] [ Host ]
```

Some useful **showmount** flags are provided in Table 7-3.

Table 7-3 Commonly used flags of the showmount command

Flag	Description
-a	Shows active mounts.
-e <server>	Shows exported file systems.

7.7.2 The exportfs command

The **exportfs** command exports and unexports directories to NFS clients.

The syntax of the **exportfs** command is:

```
exportfs [ -a ] [ -v ] [ -u ] [ -i ] [ -fFile ] [ -oOption [ ,Option ... ] ]
[ Directory ]
```

Some useful **exportfs** flags are provided in Table 7-4 on page 182.

Table 7-4 Commonly used flags of the exportfs command

Flags	Description
-a	Exports all filesets defined in /etc/exports.
-u	Unexports the directories you specify; can be used with -a.
-o <option>	Specifies optional characteristics for the exported directory.

7.7.3 The mount command

The **mount** command makes a file system available for use.

The syntax of the **mount** command is:

```
mount [ -f ] [ -n Node ] [ -o Options ] [ -p ] [ -r ] [ -v VfsName ] [ -t Type
| [ Device | Node:Directory ] Directory | all | -a ]
[-V [generic_options] special_mount_points
```

Some useful **mount** flags are provided in Table 7-5.

Table 7-5 Commonly used flags of the mount command

Flags	Description
-[a | all]	Mounts all file systems in the /etc/filesystems file with stanzas that contain the true mount attribute.
-n <node>	Specifies the remote node that holds the directory to be mounted.
-o fg	Foreground mount attempt.
-o bg	Background mount attempts.
-o proto=[tcp|udp]	Protocol to use.
-o vers=[2|3]	NFS version to use.
-o soft	Returns an error if the server does not respond.
-o hard	Retries a request until the server responds.
-o intr	Allows keyboard interrupts on hard mounts.
-o timeo=n	Sets the Network File System (NFS) timeout period to n tenths of a second.
-o retrans=n	Sets the number of NFS transmissions to n.

7.7.4 The nfsstat command

The **nfsstat** command displays statistical information about the Network File System (NFS) and Remote Procedure Call (RPC) calls.

The syntax of the **nfsstat** command is:

```
nfsstat [ -c ] [ -s ] [ -n ] [ -r ] [ -z ] [ -m ]
```

Some useful **nfsstat** flags are provided in Table 7-6.

Table 7-6 Commonly used flags of the nfsstat command

Flags	Description
-c	Displays client information.
-m	Displays statistics for each NFS file system mounted along with the server name and address, mount flags, current read and write sizes, retransmission count, and the timers used for dynamic retransmission.
-n	Prints NFS information for both the client and server.

7.7.5 The iptrace command

The **iptrace** command provides interface-level packet tracing for Internet protocols.

The syntax of the **iptrace** command is:

```
iptrace [ -a ] [ -e ] [ -PProtocol ] [ -iInterface ] [ -pPort ]
[ -sHost [ -b ] ] [ -dHost [ -b ] ] LogFile
```

Some useful **iptrace** flags are provided in Table 7-7.

Table 7-7 Commonly used flags of the iptrace command

Flags	Description
-a	Suppresses ARP packets.
-s <host>	Records packets coming from the source host specified by the host variable.
-b	Changes the -d or -s flags to bidirectional mode.

7.7.6 The ipreport command

The **ipreport** command generates a packet trace report from the specified packet trace file.

The syntax of the **ipreport** command is:

```
ipreport [ -e ] [ -r ] [ -n ] [ -s ] LogFile
```

Some useful **ipreport** flags are provided in Table 7-8.

Table 7-8 Commonly used flags of the ipreport command

Flags	Description
-s	Prepends the protocol specification to every line in a packet.
-r	Decodes remote procedure call (RPC) packets.
-n	Includes a packet number to facilitate easy comparison of different output formats.

7.7.7 The netstat command

The **netstat** command shows network status.

The syntax of the **netstat** command is:

- To display active sockets for each protocol or routing table information:

  ```
  /bin/netstat [ -n ] [ {  -A -a } |  { -r  -i  -I  Interface } ] [ -f
  AddressFamily ] [ -p  Protocol ] [ Interval ] [ System ]
  ```

- To display the contents of a network data structure:

  ```
  /bin/netstat [ -m | -s |  -ss |  -u |  -v ] [  -f AddressFamily ] [ -p
  Protocol ] [ Interval ] [  System ]
  ```

- To display the packet counts throughout the communications subsystem:

  ```
  /bin/netstat -D
  ```

- To display the network buffer cache statistics:

  ```
  /bin/netstat -c
  ```

- To display the data link provider interface statistics:

  ```
  /bin/netstat -P
  ```

- To clear the associated statistics:

  ```
  /bin/netstat [ -Zc | -Zi | -Zm | -Zs ]
  ```

Some useful **netstat** flags from an NFS point of view are provided in Table 7-9 on page 185.

Table 7-9 Commonly used flags of the netstat command

Flags	Description
-P <protocol>	Shows statistics about the value specified for the Protocol variable.
-s	Shows statistics for each protocol.
-D	Shows the number of packets received, transmitted, and dropped in the communications subsystem.

7.7.8 The chnfs command

The **chnfs** command changes the configuration of the system to invoke a specified number of biod and nfsd daemons.

The syntax of the **chnfs** command is:

```
chnfs [ -n NumberOfNfsd ] [ -b NumberOfBiod ] [ -I | -B | -N ]
```

Some useful **chnfs** flags are provided in Table 7-10.

Table 7-10 Commonly used flags of the chnfs command

Flags	Description
-n <value>	Specifies the number of nfsd daemons to run on the system.
-b <value>	Specifies the number of biod daemons to run on the system.

7.7.9 The rpcinfo command

The **rpcinfo** command reports the status of Remote Procedure Call (RPC) servers.

The syntax of the **rpcinfo** command is:

- To display a list of statistics:

  ```
  /usr/bin/rpcinfo [ -m | -s ] [ Host ]
  ```

- To display a list of registered RPC programs:

  ```
  /usr/bin/rpcinfo -p [ Host ]
  ```

- To report transport :

  ```
  /usr/bin/rpcinfo -T transport Host Prognum  [ Versnum ]
  ```

- To display a list of entries:;

  ```
  /usr/bin/rpcinfo -l  [ -T transport ] Host Prognum  Versnum
  ```

- To report program status:

  ```
  /usr/bin/rpcinfo [ -n PortNum ]  -u  Host Prognum  [ Versnum ]
  ```

- To report response status:

  ```
  /usr/bin/rpcinfo [ -n PortNum ]  -t  Host Prognum  [ Versnum ]
  ```

- To display all hosts running a specified program version:

  ```
  /usr/bin/rpcinfo [ -b ] [ -T transport ] Prognum  Versnum
  ```

- To delete registration of a service:

  ```
  /usr/bin/rpcinfo [ -d ] [ -T transport ] Prognum  Versnum
  ```

Some useful **rpcinfo** flags are provided in Table 7-11.

Table 7-11 Commonly used flags of the rpcinfo command

Flags	Description
-p <host>	Probes the portmap service on the host and displays a list of all registered RPC programs.
-m <host>	Displays a table of portmap operations statistics on the specified host.
-s <host>	Displays a concise list of all registered RPC programs on the host.

7.8 Quiz

The following assessment questions help verify your understanding of the topics discussed in this chapter.

1. A machine is required to mount remote file systems. Which one of the following services should be used?

   A. NFS

   B. NIS

   C. NTP

   D. DHCP

2. By default, for AIX Version 4.3.3 on which one of the following file systems on the AIX NFS client will the AIX automount daemon mount file systems from the NFS server?

   A. / (root) file system

   B. /tmp file system

   C. /mnt file system

D. /var file system

3. In the case where file integrity is very important, which one of the following types of mount is most appropriate for an NFS-mounted writable file system?

   A. Soft mount

   B. Hard mount

   C. Background mount

   D. Foreground mount

4. Which one of the following types of mount is most appropriate so that an NFS server crash should have the minimum effect on the state of the client machine?

   A. Soft mount

   B. Hard mount

   C. Foreground mount

   D. Background mount

5. Given the following contents of the /etc/exports file of an AIX NFS server, which one of the following conclusions is the most appropriate to draw?

```
/usr/local/bin
/src -access=anyone
/usr/spool/mail -root=rs1,-access=rs1
```

   A. /src can be written to by root on and NFS client machine.

   B. /usr/local/bin can be accessed by any NFS client.

   C. Machine rs1 has read-only access to /usr/spool/mail.

   D. The /src file system can be accessed by any NFS client.

6. Given the following contents of the /etc/exports file of an AIX NFS server, which one of the following conclusions is the most appropriate to draw?

```
/usr/local -rw=dopey:hungry:grumpy
/src -access=anyone,ro
/usr/spool/mail -root=rs1,-access=rs1
```

   A. /src can be written to by root on an NFS client machine.

   B. /src can be written to by a machine named anyone.

   C. Machine rs1 has read-only access to /usr/spool/mail.

   D. The /usr/local directory can be written to by machine grumpy.

7. A /home directory from the NFS server MachineA is trying to be mounted to the mount point /MachineA/home on the NFS client MachineB.

   Which one of the following diagnostic commands should be used to determine why the mount is hanging?

   A. **diag**

   B. **errpt**

   C. **nfsstat**

   D. **iptrace**

8. The /MachineA/home directory has been mounted and has been used for several days. Currently, all commands that try to reference files in /MachineA/home hang. **rpcinfo** shows that all RPC services on MachineA are registered. Which one of the following is the most probable cause?

   A. MachineA is down

   B. nfsd is not running on MachineA

   C. **securetcpip** has been run on MachineA

   D. /home has been unexported on MachineA

9. It becomes necessary to unmount /home from the client. **unmount** gives a message warning that the unmount timed out. Which one of the following is the most probable cause?

   A. nfsd is no longer running on the client

   B. rpc.mountd is not running on the client

   C. rpc.mountd is not running on the server

   D. /home has been unexported on the client

10. Machine A is being used as a large file repository and must be capable of transferring large amounts of data both to and from the network. Performance is the primary concern in this case. Instructions have been sent forth to tune the network for optimal performance.

    NFS performance problems have been reported on a server. In order to check for socket buffer overflows, which one of the following commands should be used?

    A. **nfso**

    B. **nfsstat**

    C. **netstat**

    D. **enstat**

11. If the following error occurred while attempting an NFS mount from a client machine, which one of the following actions should be performed?

```
# mount nfs_server:/usr/local /mnt mount: 1831-011 access denied for
nfs_server:/usr/local mount: 1831-008 giving up on: nfs_server:/usr/local
```

The file access permissions do not allow the specified action.

A. Start biod daemons on the NFS client.

B. Start nfsd daemons on the NFS server.

C. Add execute permission for others to the /usr/local directory.

D. Add an entry for /usr/local in the exports file and execute **exportfs -a** on nfs_server.

12. Which one of the following files is used by the NFS server to specify which file system can be mounted on a client?

A. /etc/filesystems

B. /etc/exports

C. /usr/bin/showmount

D. /usr/sbin/exportfs

13. Given the following auto master file, which one of the following will the AIX automount daemon search to find the mount information for a directory?

```
/home?  /-   /etc/auto.direct -ro,intr,soft,rsize=8192,wsize=8192 /home
auto.home -rw,intr,hard,rsize=8192,wsize=8192
```

A. /etc/auto.direct

B. /etc/auto.home

C. /home/auto.home

D. NIS auto.home file

14. Which one of the following specifications will verify that an AIX 5L Version 5.1 automount mount point stays mounted for at least one hour?

A. Specify "-i 3600" as a parameter to the automount command

B. Specify "-v 3600" as a parameter to the automount command

C. Specify "-t 3600" as a parameter to the automount command

D. Additional specifications are not required in the automount command as 1 hour is the default

15. Below are auto.master and auto.home map files. No home directories are currently mounted on the AIX NFS client. The automounter is using the default temporary mount location. Which one of the following actions will

occur when user3, whose home is in /home/user3, logs into the AIX NFS client?

```
# auto.master  /-   /etc/auto.direct -ro,intr,soft,rsize=8192,wsize=8192
/home  auto.home -rw,intr,hard,rsize=8192,wsize=8192 # auto.home  user1
nfs_server:/home/user1  user2 nfs_server:/home/user2  user3
nfs_server:/home/user3  user4 nfs_server:/home/user4
```

A. A `No such file or directory` error message is displayed.

B. The automount daemon mounts /home/user3 on the nfs_server machine.

C. The automount daemon mounts /home from the nfs_server in /mnt and creates a symbolic link from /home/user3 to /mnt/home/user3.

D. The automount daemon mounts /home/user3 from the nfs_server over /tmp_mnt/home/user3 and creates a symbolic link from /home/user3 to /tmp_mnt/home/user3.

16. Using the **`netstat -s`** command, which section of the output should be checked for socket buffer overflows?

A. TCP

B. UDP

C. IP

D. NFS

17. Which command will change the quantity of biod daemons immediately and for each subsequent reboot?

A. **`chnfs -b newvalue`**

B. **`chnfs -N -b newvalue`**

C. **`chnfs -I -b newvalue`**

D. **`netstat -s newvalue`**

7.8.1 Answers

The following are the preferred answers to the questions provided in this section:

1. A
2. A
3. B
4. A
5. B
6. D
7. D
8. B
9. C
10. C
11. D
12. B
13. D
14. C
15. D
16. B
17. A

7.9 Exercises

The following exercises provide sample topics for self study. They will help ensure comprehension of this chapter.

1. Start iptrace and trace a long listing of mounts and a file creation in the NFS mounted file system. Try to make a drawing of the bidirectional traffic going between the NFS daemons from the output of the iptrace.
2. Use the auto.master file to point out both an indirect and a direct map. What differences are there in the use of mount points between indirect and direct mounts? Run **iptrace** when accessing an indirect mount point. What RPCs are used for the action?

8

Domain Name System

The following topics are discussed in this chapter:

- The Domain Name System (DNS) concept.
- Setting up the DNS server.
- Setting up the DNS client.

8.1 DNS overview

When you want to connect to another system you can use the **telnet server4** command. TCP/IP will examine the /etc/hosts file for a host server4, and then read off the IP address. The host's table-based name resolution is convenient for reasonably small networks with few entries to include in the /etc/hosts file. The practice of maintaining identical /etc/hosts files on all UNIX hosts is a time-demanding method, because it requires that changes made to one must be consistently implemented in all others. This approach can easily become impractical as the size of the network increases.

Due to the growth of the number of hosts, this mechanism became too cumbersome and was replaced by a new concept: the *Domain Name System*. Hosts can continue to use a local flat namespace (/etc/hosts file) instead of, or in addition to, the DNS. The Domain Name System allows a program running on a host to perform the mapping of a high-level symbolic name to an IP address for any other host without the need for every host to have a complete database of host names.

DNS is configured on a client/server basis. The server is the name server that makes its data available to the clients. The clients (resolver) generate the query that goes to the name server requesting name-serving information. DNS is implemented by the named daemon in TCP/IP.

8.1.1 The DNS hierarchy

The hierarchical structure of the DNS enables the distribution and delegation of responsibility for host name-to-IP-address mapping. Whereas the /etc/hosts file requires an entry for every possible system you might wish to connect to, DNS requires only that you maintain the data for your administrative domain. Host lookups for a given domain are then serviced by the domain's name server. A DNS hierarchy is organized into an inverted tree that can be traversed to service requests for hosts from another domain. See Figure 8-1 on page 195 for a graphical representation of the DNS hierarchy.

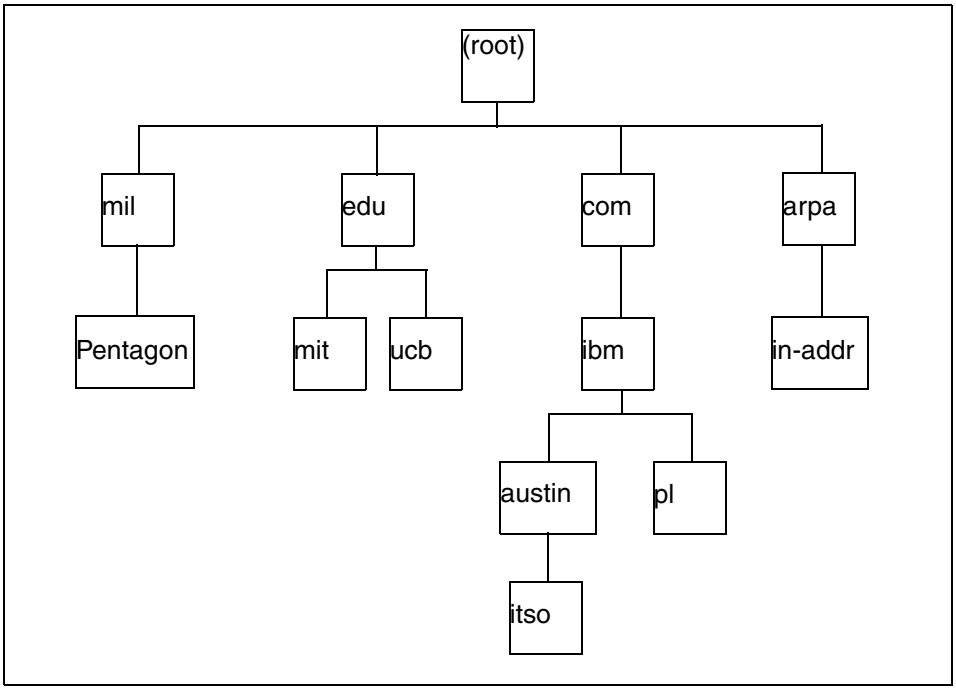


Figure 8-1 DNS structure

This structure has a root domain at the highest level. All domains under the root domain (com, edu, mil, and others) are called top-level domains. A fully qualified domain name is the sequence of names from the local domain up to the root. Each of the top-level domains are subdivided into subdomains. The root name server knows where all the name servers are from top-level domains.

There is one special domain named in-addr.arpa that was created to solve the problem of mapping IP address to host names. IP address are represented in PTR resource records as a domain name, so now it is possible to perform inverse addressing with the same efficiency as regular name service lookup.

8.1.2 Domain name resolution

The domain name resolution process proceeds in the following steps:

1. A user program issues a request for the IP address of a host by passing the host name.
2. The resolver formulates a query to the name server.
3. The name server checks to see if the answer is in its local authoritative database or cache, and if so, returns it to the client. Otherwise, it will query

other available name servers, starting down from the root of the DNS tree or as high up the tree as possible.

4. The user program will finally be given a corresponding IP address.

The query and reply messages are transported by either UDP or TCP.

8.1.3 DNS resource records

Basically, a DNS resource record (RR) is an entry in the DNS database that specifies information for some resources. RRs are stored in the DNS database files, which are read when the DNS server is started. The most common RRs are provided in Table 8-1.

Table 8-1 Common DNS resource record types

Record type	Description
SOA	Start Of Authority: Specifies which host is the definitive authority or primary source of the domain data. An SOA record is required for each defined domain and only one SOA record per database file is permitted.
NS	Name Server: Specifies the name server for the domain. It is possible to have multiple name servers. There should be an entry for each name server in the domain.
A	Address: Each reachable host in the domain will require that an A record be maintained so that tthe name server can perform host name-to-IP-address mapping.
CNAME	Canonical Name: Used in the specification of a host name alias.
PTR	Pointer: The PTR record performs the inverse function of an A record, that is, IP-address-to-host name mapping.
MX	Mail Exchanger: Specifies a host that provides advanced e-mail routing capabilities for the domain.

8.1.4 DNS components

DNS performs host name-to-IP-address mapping using a distributed hierarchical database to maintain mapping. This system consists of a few components: primary server, secondary server, and DNS client.

Primary server

The primary name server provides authoritative name lookup response for the zone it serves. Authoritative response means that the zone data files that are maintained by the network administrator reside on this server.

Secondary server

The secondary server provides the same services as the primary server, but the data for the zone is not kept locally, but is obtained from the primary authoritative server. This data requesting is called zone transfer. Response to queries from a secondary server are known as non-authoritative response.

Caching-only servers

A name server that does not have authority for any zone is called a caching-only name server. A caching-only name server obtains all of its data from primary or secondary name servers as required. Once an answer is received back, the caching-only name server will cache the answer.

Forwarders

This configuration causes the server to forward queries on to another name server for resolution. Name service lookups to this type of server will be forwarded to the specified name server.

8.2 Setting up a primary DNS server

Configuring a DNS server requires several files and databases to be modified or created. The process is time-consuming, but is done only once. Configuration steps are as follows:

1. Create the /etc/named.boot.
2. Create the name zone file.
3. Create the IP zone file.
4. Create the local IP zone file.
5. Create the cache file.
6. Start the named daemon.

8.2.1 The /etc/named.boot file

The /etc/named.boot file is read by the named daemon when it starts. It specifies the location of the database files. The following is a simple /etc/named.boot file for domain test.ibm.com and for network 9.3.240.0:

```
# cat /etc/named.boot
directory       /etc
primary         test.ibm.com            named.test
primary         240.3.9.in-addr.arpa    named.rev.240
primary         0.0.127.in-addr.arpa    named.rev.local
cache           .                       named.cache
```

This file has the following attributes:

- The directory entry tells the named daemon where the configuration files are located. In this example, files are stored in the /etc directory.
- The primary entry indicates the domain for which this named daemon is the primary name server and the file that contains name-to-address resolution mapping information for all machines in the name server's zone of authority. As you can see in the example, this is the primary server for domain test.ibm.com; mappings are stored in /etc/named.test file.
- The third line points to the file /etc/named.rev.240, which maps the IP address for network 9.3.240.0. This is for reverse name resolution purposes. The name server is the primary server for reverse domain 240.3.9.in-addr.arpa. In this file subnetwork, addresses are listed in reverse order because the IP addresses have the most significant octets first.
  - The IN-ADDR.ARPA record

    The structure of names in the domain system is set up in a hierarchical fashion. The address of a name can be found by tracing down the domain structure, contacting a server for each label in the name. Because the structure is based on names, there is no easy way to translate a host address back into its host name.

    In order to allow simple reverse translation, the IN-ADDR.ARPA domain was created. This domain uses host addresses as part of a name that points to the data for that host. The IN-ADDR.ARPA domain provides an index to the resource records of each host based on its address. There are subdomains within the IN-ADDR.ARPA domain for each network, based on network number. Also, to maintain consistency and natural groupings, the 4 octets of a host number are reversed. The IN-ADDR.ARPA domain is defined by the IN-ADDR.ARPA record in the named.boot files and the DOMAIN hosts data file.
- The fourth line is the statement for loopback.
- The last line describes the cache file, which contains addresses for the root domain servers.

Note: You can use any file name you want for data files with the exception of the /etc/named.boot file name.

8.2.2 The name zone file

The host's data file is one of the data files and contains name-to-address resolution mapping information for all machines in the name server's zone of authority. IBM provides two awk scripts that can help you build name zone files.

Be careful when you decide to use these scripts; they do not generate ideal zone files. The /usr/samples/tcpip/hosts.awk builds the name-to-IP-address database and /usr/samples/tcpip/addrs.awk builds the reverse IP file. Here is an example of these scripts:

```
# cd /usr/samples/tcpip/
# ./hosts.awk /etc/hosts > /etc/named.test
# ./addrs.awk /etc/hosts > /etc/named.rev.240
```

The SOA record indicates the start of a zone of authority. There should be only one SOA record per zone. However, the SOA record for the zone should be in each name zone file and IP zone file on each name server in the zone. The name zone file starts with an SOA record.

The primary server name zone file for network itsc.austin.ibm.com, stored on host server4.itsc.austin.ibm.com in the file /etc/named.test, contains the following entries:

```
# cat /etc/named.test
; name server data file
; (also see /etc/named.boot)
;
; NAME          TTL     CLASS   TYPE    RDATA
;
; setting default domain to "itsc.austin.ibm.com"
;
@               9999999 IN      SOA     server4.itsc.austin.ibm.com.
root.server
4.itsc.austin.ibm.com. (
                                        1.1             ; Serial
                                        3600            ; Refresh
                                        300             ; Retry
                                        3600000         ; Expire
                                        86400 )         ; Minimum
                9999999 IN      NS      server4
loopback        9999999 IN      A       127.0.0.1       ; loopback (lo0)name/ad
dress
localhost       9999999 IN      CNAME   loopback
server4         9999999 IN      A       9.3.4.100
server1         9999999 IN      A       9.3.4.97
server5         9999999 IN      A       9.3.4.29
```

Major fields in the SOA record and their meanings:

NAME	Name of the zone. The @ sign indicates the zone is the same as that indicated in /etc/named.boot.
TTL	Time to live. A value 9999999 means no timeout.
CLASS	Internet (IN).

TYPE	Start of authority (SOA).
RDATA	Name of the host on which this data file resides.
Serial	Version number of this data file. This number is incremented each time a change is made to the data. The upper limit for the number to the right of the decimal point is 9999. The secondary name server checks this value to see if it needs to download information again.
Refresh	The number of seconds after which a secondary name server checks with the primary name server to see if an update is needed.
Retry	The number of seconds after which a secondary name server is to retry after a refresh attempt fails.
Expire	The upper limit in seconds that a secondary name server can use the data before it expires because it has not been refreshed.
Minimum	The minimum time, in seconds, to use as time-to-live values in resource records.

Below the SOA record there are entries with name-to-IP-address mapping. The first column indicates host name. The second column defines the length of time, in seconds, that the information from this record should stay in cache. If there is no value, the default becomes the value of the Minimum TTL field in SOA. The third field defines the class of address; IN means Internet address. The next column is the class of record (refer to Table 8-1 on page 196). The last column contains IP addresses except in the case of CNAME records, in which case it contains host names defined elsewhere in the /etc/named.boot file.

8.2.3 The IP zone file

An IP zone file is used for IP-address-to-name mapping. It looks similar to a name zone file with the exception of addresses. What is new in this file is the PTR resource record type in the type field. The PTR records provide address-to-name conversions. The host name in the last column is fully qualified.

The primary server IP zone file for network itsc.austin.ibm.com, stored on host server4.itsc.austin.ibm.com in the file /etc/named.rev.240, contains the following entries:

```
# cat /etc/named.rev.240
; setting default domain to ... itsc.austin.ibm.com
@               9999999 IN      SOA     server4.itsc.austin.ibm.com.
root.server
4.itsc.austin.ibm.com. (
```

```
                                        1.1             ; Serial
                                        3600            ; Refresh
                                        300             ; Retry
                                        3600000         ; Expire
                                        86400 )         ; Minimum
                9999999 IN      NS      server4.itsc.austin.ibm.com.
1.0.0.127       IN PTR loopback.itsc.austin.ibm.com.
100.4.3.9       IN PTR server4.itsc.austin.ibm.com.
97.4.3.9        IN PTR server1.itsc.austin.ibm.com.
29.4.3.9        IN PTR server5.itsc.austin.ibm.com.
```

As previously discussed, use the /usr/samples/tcpip/addrs.awk script to create this file.

8.2.4 The local IP zone file

The local IP zone file contains the PTR record for the loopback address. The SOA record is not required in this file. The presence of the @ sign indicates the current domain. In the example of a primary name server, this file is named named.rev.local and is located in the /etc directory. The following example shows the content of this file:

```
# cat /etc/named.rev.local
@               IN      NS      server4.test.ibm.com.
1.0.0.127       IN      PTR     loopback.
```

8.2.5 The root cache file

Now that all the local information is complete, the name server needs to know about the root name server for the domain. This data is known as the root cache. The root server for the example name server is the machine dhcp240.itsc.austin.ibm.com with IP address 9.3.240.2. The root cache file looks like this:

```
# cat /etc/named.cache
.                               9999999   IN NS dhcp240.itsc.austin.ibm.com.
dhcp240.itsc.austin.ibm.com.    9999999   IN A  9.3.240.2
```

The dot in the first line indicates the default domain.

8.2.6 The /etc/named.hosts file

The named.hosts file on a primary server contains the authoritative information for a zone. Following is an example of a named.hosts file:

```
cat /usr/samples/tcpip/named.hosts
.....
.....
```

```
; OWNER         TTL             CLASS   TYPE    RDATA
;
; define domain nameserver (Note trailing dot)
grandchild.child.parent.top.            99999999        IN      NS      vmail
; address of domain nameserver
vmail                   99999999        IN      A       192.9.200.1
; addresses of other machines in the domain
net2sample              99999999        IN      A       192.9.200.2
net3sample              99999999        IN      A       192.9.200.3
.....
.....
```

8.2.7 Starting named daemon

Create an /etc/resolv.conf file by issuing the following command:

```
# touch /etc/resolv.conf
```

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 either may contain the local host's address and the loopback address or be empty. Alternatively, the /etc/resolv.conf file may contain the following entry:

```
# cat /etc/resolv.conf
nameserver 127.0.0.1
domain test.ibm.com
```

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

Next, change the host name to a fully qualified domain name using the **smitty hostname** or **chdev** command:

```
# chdev -l inet0 -a hostname=server4.test.ibm.com
inet0 changed
```

Now you can start the named daemon with the command **startsrc -s named**. The /etc/rc.tcpip file must be changed so that the named daemon will be started at the system reboot.

8.3 Setting up a secondary DNS server

The difference between the primary and secondary name server is where they get their information. The primary reads its own files, but the secondary downloads information from the primary using a zone transfer. Periodically, the secondary name server checks in with the primary server to see if the database has changed. The advantage of a secondary name server is there is no

maintenance of files. All the file maintenance is done on the primary name server. The /etc/named.boot file, local IP zone file, and cache file must be created on the secondary name server. They are not part of the zone transfer.

8.3.1 The /etc/named.boot file for a secondary name server

The /etc/named.boot file for the secondary name server looks the same as the one used in a primary name server, except that the IP address for the primary server is added. This addition tells the name server that it is the secondary name server for that specified domain. This server is the only primary name server for localhost. The following is an example of a /etc/named.boot file for the secondary name server:

```
# cat /etc/named.boot
directory       /etc
secondary       test.ibm.com            9.3.240.59      named.test.bak
secondary       240.3.9.in-addr.arpa    9.3.240.59      named.rev.240.bak
primary         0.0.127.in-addr.arpa    named.rev.local
cache           .                       named.cache
```

8.3.2 Local IP zone file for secondary name server

The local IP zone file appears the same as what was entered on the primary name server with the exception of indicating itself in the SOA and NS record.

```
# cat /etc/named.rev.local
@               9999999 IN      SOA     server3.test.ibm.com.
root.server3.test.ibm.com. (
                                        1.0             ; Serial
                                        3600            ; Refresh
                                        300             ; Retry
                                        3600000         ; Expire
                                        86400 )         ; Minimum TTL
        IN      NS      server3.test.ibm.com.
1       IN      PTR     loopback.
```

8.3.3 Starting up a secondary name server

Before you start the named daemon, you must copy the root cache file from the primary name server and create empty /etc/resolv.conf file:

```
# touch /etc/resolv.conf
```

Now you are ready to start the daemon. You can use either the **startsrc -s named** or **smitty stnamed** commands. Remember to uncomment the following line in the /etc/rc.tcpip file to make named start automatically after a reboot:

```
start /usr/sbin/named "$src_running"
```

After you start the named daemon files, /etc/named.test.bak and /etc/named.rev.240.bak will be created from the primary name server's database.

8.4 Setting up a cache-only name server

This name server is not authoritative for any domains except localhost. It just responds to clients based on its queries to the other name servers. Every resolved query is cached so it can later respond to clients using its cache. To configure it, you need to set up the /etc/named.boot file, the local IP zone file for localhost, and the cache file.

The /etc/named.boot file appears as follows:

```
# cat /etc/named.boot
directory       /etc
primary         0.0.127.in-addr.arpa    named.rev.local
cache           .                       named.cache
```

Start the named daemon and your cache-only name server is ready to run.

8.5 Setting up the DNS client

When you have the primary and secondary name servers set up, it is time to set up the DNS client. First change the client's host name to a fully qualified domain. You can use **smitty hostname** or **chdev** command to permanently change host name:

```
# chdev -l inet0 -a hostname=client.test.ibm.com
inet0 changed
```

The next step is to create the /etc/resolv.conf file. It should contain the domain name and name servers (primary and secondary) IP addresses:

```
# cat /etc/resolv.conf
domain       test.ibm.com
nameserver   9.3.240.59
nameserver   9.3.240.58
```

To check if the DNS client is set up correctly, use the **nslookup** command and try to resolve a few names of other systems:

```
# nslookup
Default Server:  server4.test.ibm.com
Address:  9.3.240.59

> gateway
```

```
Server:  server4.test.ibm.com
Address:  9.3.240.59

Name:    gateway.test.ibm.com
Address:  9.3.240.1

> 9.3.240.57
Server:  server4.test.ibm.com
Address:  9.3.240.59

Name:    server2.test.ibm.com
Address:  9.3.240.57
```

Resolver routines on hosts running TCP/IP normally attempt to resolve names using the following sources:

- DNS (named)
- Network Information Service (NIS)
- Local /etc/hosts file

By default, resolver routines attempt to resolve names using the above resources. DNS will be tried first. If the /etc/resolv.conf file does not exist or if DNS could not find the name, NIS is queried if it is running. NIS is authoritative over the local /etc/hosts, so the search will end here if it is running. If NIS is not running, then the local /etc/hosts file is searched.

This default order can be overwritten by creating the configuration file, /etc/netsvc.conf, and specifying the desired ordering. The environment variable NSORDER overrides the host settings in the /etc/netsvc.conf file. The example /etc/netsvc.conf file is as follows:

```
# cat /etc/netsvc.conf
hosts = local , nis
```

If both the /etc/netsvc.conf file and the NSORDER are used, NSORDER overrides the /etc/netsvc.conf file.

The values specified and their ordering is dependent on the network configuration. For example, if the local network is organized as a flat network, then only the /etc/hosts file is needed. The /etc/netsvc.conf file would contain the following line:

```
hosts=local
```

The NSORDER environment variable would be set as:

```
NSORDER=local
```

If the local network is a domain network using a name server for name resolution and an /etc/hosts file for backup, specify both services. The /etc/netsvc.conf file would contain the following line:

```
hosts=bind,local
```

The NSORDER environment variable would be set as:

```
NSORDER=bind,local
```

The algorithm will attempt the first source in the list. The algorithm will then determine to try another specified service based on:

- Current service is not running; therefore, it is unavailable.
- Current service could not find the name and is not authoritative.

8.6 Quiz

The following assessment questions help verify your understanding of the topics discussed in this chapter.

1. Given a host with the following /etc/named.boot file:

```
directory /var/named
secondary    nuts.com              128.66.12.1  named.hosts
secondary    132.128.in-addr.arpa  128.66.12.1  named.rev
primary      0.0.127.in-addr.arpa               named.local
cache  .                                        named.ca
```

   Which one of the following statements is valid?

   A. The address 128.66.12.1 is the primary server for the network 132.128.0.0.

   B. The address 128.66.12.1 is the backup secondary server.

   C. The address 128.66.12.1 indicates this is a secondary server for network 128.66.0.0.

   D. The address 128.66.12.1 is the IP address for this host to use to download data for the nuts.com domain.

2. In a DNS environment, the zone file that maps IP addresses to host names (sometimes called the named.rev file) is created on which one of the following servers?

   A. Cache

   B. Primary

   C. Secondary

   D. Primary and secondary

3. A gateway machine has access to the Internet and is trying to reach a machine on the Internet called cactus.org. Although the gateway machine cannot reach cactus.org, another network across town is able to reach cactus.org. Furthermore, the gateway machine can reach the network across town, but cannot ping the cactus.org. Which one of the following tools will best help diagnose the location of the problem?

   A. **iptrace**

   B. **netstat**

   C. **tcpdump**

   D. **traceroute**

4. Assume a host in domain peanut.com. The named.boot is as follows:

   ```
   primary 0.0.127.in-addr.arpa /etc/named.local  cache . /etc/named.ca
   ```

   This host performs which one of the following functions?

   A. A cache only server

   B. A reverse name lookup server for domain peanut.com

   C. A primary name server for domain arpa.com and cache server

   D. A primary name server for domain peanut.com and cache server

5. When a DNS secondary server is set up, which one of the following files is required locally?

   A. /etc/named.ca

   B. /etc/inetd.conf

   C. /etc/named.boot

   D. /etc/named.hosts

6. Which one of the following pieces of information are *not* required in the local IP zone file?

   A. PTR

   B. SOA

   C. NS

   D. Hostname

8.6.1 Answers

The following are the preferred answers to the questions provided in this section:

1. D
2. B
3. D
4. A
5. C
6. B

8.7 Exercises

The following exercises provide sample topics for self study. They will help ensure comprehension of this chapter.

1. On a test system that does not affect any user, set up a primary name server.
2. On the other system, change the reference to the primary name server that you set up previously by editing the /etc/resolv.conf file.
3. Change the name resolution default order by editing the /etc/netsvc.conf file, so that /etc/hosts file will be used before the domain name server. Add an entry to the /etc/hosts file that is not in the name server.
4. Set the value of the NSORDER environment value to override the /etc/netsv.conf file.

9

Mail services

In AIX there are three mail programs available for use, as follows:

- mail
- mh
- bellmail

A user-agent program provides facilities for creating, receiving, sending, and filing mail. In addition, you need a transport-agent program, sendmail, which distributes incoming mail from other systems or packages, and distributes each outgoing mail item and transmits it to a similar program in one or more remote systems.

Note: The mail and mh mail systems are incompatible in the way they store mail; either one mail handler or the other must be used, not both.

In this chapter, the mail user-agent will be used, since this is the most commonly used mail program in AIX.

9.1 Mail system overview

The following sections discuss the basic features of the mail, mh, and bellmail systems.

9.1.1 The mail system

The mail system provides you with a user interface to handle mail to and from both a local network user and a remote system user.

A mail message can be text, entered using an editor, or an ASCII file. In addition to a typed message or a file, you can send:

System messages Informs users the system has been updated. A system message is similar to a broadcast message, but is sent on the local network only.

Secret mail Used to send classified information. A secret mail message is encrypted. The recipient must enter a password to read it.

Vacation message Informs users you are on vacation. When your system receives mail in your absence, it sends a message back to the origin. The message states you are on vacation. Any mail you receive while on vacation can also be forwarded.

Mail relay Relaying of mail messages to another mail server. In order to relay mail, you will need to have sendmail (see 9.3, “The sendmail command” on page 212) running as a background daemon. This, by definition, will allow incoming mail to your mail server. However, you can selectively allow relaying by adding the fully qualified names (or IP addresses) of the allowed hosts to your /etc/mail/relay-domains file. Once you do this, and run the **`refresh -s sendmail`** command, those hosts will be allowed to relay through your machine. The installation of sendmail is automatic.

Mail endpoint Destination mail server. Messages will not be relayed beyond this point.

When you receive mail using the **`mail`** subcommands, you can:

- Leave the mail in the system mailbox.
- Read and delete the mail.
- Forward the mail.
- Add comments to the mail.

- Store the mail in your personal mailbox (mbox).
- Store the mail in a folder you have created.
- Displays a list of aliases and their addresses.

9.1.2 The mh system

The mh mail system is a collection of commands that enables you to perform each mail processing function directly from the command line. These commands provide a broader range of function than the subcommands of mail, and since they can be issued at any time the command prompt is displayed, you gain power and flexibility in creating mail and in processing received mail. For example, you can read a mail message, search a file or run a program to find a particular solution, and answer the message, all within the same shell.

The mh mail system enables you to create, distribute, receive, view, process, and store messages.

9.1.3 The bellmail system

The bellmail mail system is the original AT&T UNIX mail command, which handles mail for users on the same system and also for users on remote systems that can be accessed by means of Basic Network Utilities (BNU), sometimes known as the UNIX-to-UNIX Copy Program (UUCP). These programs support only networks of systems connected by dial-up or leased point-to-point communication lines.

9.2 The mailq command

The **mailq** command prints a list of messages that are in the mail queue. The **mailq** command is the same as the **sendmail -bp** command.

Specify the -v flag to display message priority.

The log file and temporary files associated with the messages in the mail queue are kept in the /var/spool/mqueue directory.

Running the **mailq** command will give the following results:

```
# mailq
                There is 1 request in the mail queue
---QID---- --Size-- -----Q-Time----- ----------Sender/Recipient-----------
0AA19258*      29 Mon Jun 26 14:57 root
                                   root@server2
```

Running the **mailq -v** command will give the following results:

```
# mailq -v
                There is 1 request in the mail queue
--Q-ID-- --Size-- -Priority- ---Q-Time--- ---------Sender/Recipient--------
0AA19258*      29      30047 Jun 26 14:57 root
                                          root@server2
```

9.3 The sendmail command

The **sendmail** command receives formatted text messages and routes the messages to one or more users. Used on a network, the **sendmail** command translates the format of the header information of the message to match the requirements of the destination system. The program determines the network of the destination system by using the syntax and content of the addresses.

The **sendmail** command can deliver messages to:

- Users on the local system.
- Users connected to the local system using the TCP/IP protocol.
- Users connected to the local system using the Basic Networking Utilities (BNU) command protocol.

The **sendmail** command is not intended as a user interface routine; other commands provide user-friendly interfaces. Use the **sendmail** command only to deliver preformatted messages.

The **sendmail** command uses a configuration file (the /etc/sendmail.cf file by default) to set operational parameters and to determine how the command parses addresses. This file is a text file that you can edit with other text editors. After modifying sendmail.cf, refresh the sendmail daemon.

After making any changes to the sendmail.cf file the sendmail daemon must be instructed to re-read the new configuration information in /etc/sendmail.cf.

The /etc/mail/sendmail.cf file:

- Stores information about the type of mailer programs running.
- Defines how the **sendmail** command rewrites addresses in messages.
- Defines how the **sendmail** command operates in the following environments:
  - Local mail delivery.
  - Local area network delivery using TCP/IP.
  - Remote delivery using Basic Utilities Network (BNU).

The /etc/mail/sendmail.cf file consists of a series of control lines, each of which begins with a single character defining how the rest of the line is used. Lines beginning with a space or a tab are continuation lines. Blank lines and lines beginning with a # (pound sign) are comments. Control lines are used for defining:

- Macros and classes for use within the configuration file.
- Message headings.
- Mailers.
- Options for the **sendmail** command.

Macros in the /etc/mail/sendmail.cf file are interpreted by the **sendmail** command. A macro is a symbol that represents a value or string. A macro is defined by a D subcommand in the /etc/mail/sendmail.cf file. The syntax for macro definitions is:

D*xval*

where x is the name of the macro (which may be a single character or a word in braces) and val is the value it should have. There should be no spaces given that do not actually belong in the macro value. Macros are interpolated using the construct $x, where x is the name of the macro to be interpolated.

AIX defines the following macros:

$_	RFC1413-validation & IP source route (V8.1 and above).
$a	The origin date in RFC822 format.
$b	The current date in RFC822 format.
$(bodytype)	The ESMTP BODY parameter.
$B	The BITNET relay.
$c	The hop count.
$(client_addr)	The connecting host's IP address.
$(client_name)	The connecting host's canonical name.
$(client_port)	The connecting host's port name.
$(client_resolve)	Holds the result of the resolve call for $(client_name).
$(currHeader)	Header value as quoted string
$C	The host name of the DECnet relay (m4 technique).
$d	The current date in UNIX (ctime)(3) format.
$(daemon_addr)	The IP address on which the daemon is listening for connections. Unless DaemonPortOptions is set, this will be 0.0.0.0.

$(daemon_family)	**If the daemon is accepting network connections, this is** the network family.
$(daemon_flags)	The flags for the daemon as specified by the Modifiers= part of DaemonPortOptions where the flags are separated from each other by spaces and uppercase flags are doubled.
$(daemon_info)	Information about a daemon as a text string. For example, SMTP+queueing@00.
$(daemon_name)	The name of the daemon from DaemonPortOptions Name= suboption. If this suboption is not used, the default will be set to Daemon#, where # is the daemon number.
$(daemon_port)	The port on which the daemon is accepting connections. Unless DaemonPort Options is set, this will most likely be set to the default of 25.
$(deliveryMode)	The current delivery mode used by **sendmail**.
$e	Obsolete. Used SmtpGreetingMessage option instead.
$(envid)	The original DSN envelope ID.
$E	X400 relay (unused) (m4 technique).
$f	The sender's address.
$F	FAX relay (m4 technique).
$g	The sender's address relative to the recipient.
$h	Host part of the recipient address.
$H	The mail hub (m4 technique).
$(hdrlen)	The length of the header value, which is stored in $(currHeader).
$(hdr_name)	The name of the header field for which the current header check ruleset has been called.
$i	The queue identifier.
$(if_addr)	The IP address of an incoming connection interface unless it is in the loopback net.
$(if_name)	The name of an incoming connection interface.
$j=	The official canonical name.
$k	The UUCP node name (V8.1 and above).
$l	Obsolete. Use UnixFromLine option instead.
$L	Local user relay (m4 technique).

$m	The DNS domain name (V8.1 and above).
$M	Who we are masquerading as (m4 technique).
$(mail_addr)	The address part of the resolved triple of the address given for the SMTP MAIL command.
$(mail_host)	The host from the resolved triple of the address given for the SMTP MAIL command.
$(mail_mailer)	The mailer from the resolved triple of the address given for the SMTP MAIL command.
$n	The error messages sender.
$(ntries)	The number of delivery attempts.
$o	Obsolete. Use OperatorChars option instead.
$opMode	The startup operating mode (V8.7 and above).
$p	The **sendmail** process ID.
$q-	Default form of the sender address.
$(queue_interval)	The queue run interval as defined in the -q flag.
$r	The protocol used.
$R	The relay for unqualified names (m4 technique).
$(rcpt_addr)	The address part of the resolved triple of the address given for the SMTP RCPT command.
$(rcpt_host)	The host from the resolved triple of the address given for the SMTP RCPT command.
$(rcpt_mailer)	The mailer from the resolved triple of the address given for the SMTP RCPT command.
$s	The sender's host name.
$S	The Smart host (m4 technique).
$(server_addr)	The address of the server of the current outgoing SMTP connection.
$(server_name)	The name of the server of the current outgoing SMTP connection.
$t	Current time in seconds.
$u	The recipient's user name.
$U	The UUCP name to override $k.
$v	The sendmail program's version.
$V	The UUCP relay (for class $=V) (m4 technique).

$w	The short name of this host.
$W	The UUCP relay (for class $=W) (m4 technique).
$x	The full name of the sender.
$X	The UUCP relay (for class $=X) (m4 technique).
$y	The home directory of the recipient.
$z	The name of the controlling TTY.
$Y	The UUCP relay for unclassified hosts.
$z	The recipient's home directory.
$Z	The version of this m4 configuration (m4 technique).

Lines in the configuration file that begin with a capital letter H define the format of the headers used in messages. The format of the H control line is:

```
H[?MailerFlags?]FieldName: Content
```

Programs and interfaces to mailers are defined in Mailer (M) line. The format is:

```
Mname, {field=value}*
```

There are several global options (O) that can be set from a configuration file. The syntax of this line is:

```
O option=value
```

The /etc/sendmail.pid file is a sendmail configuration file that allows you to stop and start the sendmail daemon. Root must have access to this file for the sendmail daemon to start and stop successfully.

The /etc/sendmail.pid file, which also has a link to /etc/mail/sendmail.pid in AIX Version 5L and later, contains the process identifier of the current sendmail daemon and the sendmail daemon startup command. The information in this file should be used to correctly stop and restart the sendmail daemon after a change has been made in the /etc/sendmail.cf file. To view the sendmail.pid file, enter the following command:

```
# cat /etc/sendmail.pid
12136
/usr/lib/sendmail -bd -q30
```

If the sendmail daemon was started using the **/usr/sbin/sendmail** command, the following command will stop the current sendmail daemon:

```
# kill ‘head -1 /etc/sendmail.pid‘
```

Once the old sendmail process is gone, a new instance should be started as in the following example:

```
eval ‘tail -1 /etc/sendmail.pid‘
```

Or, if you started the **sendmail** command using the **startsrc** command, enter the following:

```
#refresh -s sendmail
0513-095 The request for subsystem refresh was completed successfully.
```

Both of these commands cause the daemon to reread the /etc/sendmail.cf file, the /etc/aliases file, and the /etc/sendmail.nl file.

The **sendmail** command allows you to define aliases to use when the **sendmail** command handles the local mail. Aliases are alternate names that you can use in place of elaborate network addresses. You can also use aliases to build distribution lists.

Define aliases in the /etc/aliases file. This file is a text file you can edit. The **sendmail** command uses a database version of this file. You must build a new alias database by running the **sendmail -bi** command or the **newaliases** command before any changes made to the /etc/aliases file become effective.

Note: When defining aliases in the /etc/aliases file, use only lowercase characters for nested aliases. Uppercase characters on the right-hand side of an alias are converted to lowercase before being stored in the Database Manager (DBM) database.

Every system must have a user or user alias designated as the postmaster alias. The default postmaster alias is a root file. You can assign this alias to a different user in the /etc/aliases file. The postmaster alias allows other users outside your system to send mail to a known ID and to get information about mailing to users on your system. Also, users on your system can send problem notifications to the postmaster ID.

To add an alias to a system, edit the /etc/aliases file. In the example, the alias that will be added is `certify`, which can reside on the same or different servers. Edit the /etc/aliases file using **vi** or another editor and insert the following line:

```
certify: user02, user5801@server3, root@server4, user5911@server4
```

The new entry in the /etc/aliases is shown as follows:

```
# Alias for mailer daemon
MAILER-DAEMON:root

# Following alias is required by the new mail protocol, RFC 822
```

```
postmaster:root

# Aliases to handle mail to msgs and news
nobody: /dev/null
certify: user02, user5801@server3, root@server4, user5911@server4
```

Rebuild the aliases database file as follows:

```
# sendmail -bi
/etc/aliases: There are 4 aliases. The longest is 56 bytes, with 109 bytes
total.
```

or

```
# newaliases
/etc/aliases: There are 4 aliases. The longest is 56 bytes, with 109 bytes
total.
```

Either the **sendmail -bi** or **newaliases** command can be used, because both commands function the same.

When mail is sent to the user, certify it will now be sent to all the users defined as aliases in the /etc/aliases file.

9.4 Sendmail upgrade enhancements (5.1.0)

AIX 5L Version 5.1 uses Sendmail Version 8.11.0. This version has several enhancements and changes.

- The sendmail files sendmail.cf and aliases have been moved to the /etc/mail directory. Links exist on the POWER platforms that are required for the migration to AIX 5L Version 5.1 from earlier releases of AIX. The sendmail files are in /etc/mail and no links exist between them and the /etc directory.

  ```
  # ls -l /etc/sendmail.cf /etc/aliases
  lrwxrwxrwx 1 root system 21 Mar 07 10:28/etc/sendmail.cf->/etc/mail/sendmail.cf
  lrwxrwxrwx 1 root system 17 Mar 07 10:28 /etc/aliases->/etc/mail/aliases
  ```

- Sendmail supports the Berkeley DB 3.1.14 format to more efficiently store the aliases.db database file. Other databases used can store their data in the Berkeley database formats.
- Support for message submission agents.
- Multiple queues, memory buffered pseudo files, and more control over resolver timeouts improve performance.
- The ability to connect to servers running on named sockets.
- Better LDAP integration and support for LDAP-based routing.

- Improved support for virtual hosting.
- Anti-spam control features.
- Several new map classes, which include arith and macro.

More information on Sendmail Version 8.11.0 is available from the following Web site:

http://www.sendmail.org

9.5 Quiz

The following assessment questions help verify your understanding of the topics discussed in this chapter.

1. Scenario: A network administrator has been asked to integrate a new RS/6000 to be used as a corporate mail server into the network. There are five nodes on the Ethernet II network, with a network address of 193.3.7.0 and a subnet mask of 255.255.255.0. The machine contains ATM, token-ring and integrated Ethernet adapters.

   Once the system has been configured as a mail server. Which one of the following commands should be used to check the status of pending mail?

   A. **`mailx`**

   B. **`mailq`**

   C. **`bellmail`**

   D. **`sendmail`**

2. After editing mail aliases on the mailserver, which one of the following actions should be performed to put the changes into effect?

   A. **`sendmail -bi`**

   B. **`startrc -s sendmail`**

   C. **`refresh -s sendmail`**

   D. Updates are automatic so no action is required

3. A user would like for personal e-mail to be redirected to another system. Which one of the following files may be modified in order to perform this action?

   A. /etc/aliases

   B. /etc/.forward

   C. /etc/sendmail.cf

D. /etc/netsvc.conf

4. Assuming a system administrator has correctly set up a system's fully qualified host name including the correct domain, what is needed to modify to correctly set the identity of the host for sendmail?

   A. Dw, Cw and Dj in /etc/sendmail.cf

   B. $w, $j and $m in /etc/sendmail.cf

   C. Nothing; sendmail will try to find the right values

   D. Dw and $w, Dj and $j in /etc/sendmail.cf

5. Which one of the following in /etc/sendmail.cf can be used to change the host name used for sendmail?

   A. Dw macro

   B. $j macro

   C. Cw macro

   D. $=m value

6. A network administrator has been asked to integrate a new server to be used as a corporate mail server into the network. There are five nodes on the Ethernet network, with a network address of 193.3.7.0 and a subnet mask of 255.255.255.0. The machine contains ATM, token-ring and integrated Ethernet adapters. Which one of the following files contain the name of the process acting as a smux peer?

   A. /etc/inetd.conf

   B. /etc/snmpd.conf

   C. /etc/snmpd.peers

   D. /etc/netsvc.conf

9.5.1 Answers

The following are the preferred answers to the questions provided in this section:

1. B
2. A
3. A
4. C
5. A
6. C

9.6 Exercises

The following exercises provide sample topics for self study. They will help ensure comprehension of this chapter.

1. What does the **`mailq`** command do?
2. How is mail redirected?

10

NIS

In this chapter the following topics are discussed:

- Components of NIS.
- NIS configuration considerations.
- Startup of NIS.
- Managing NIS maps.

Network Information Service (NIS) is a distributed database that allows you to maintain consistent configuration files throughout your network. NIS replaces replicated copies of common configuration files, such as /etc/passwd and /etc/hosts, with data maps for each file located on a central server.

NIS is the current name for the service originally known as Yellow Pages (YP). NIS and YP are functionally identical. When working with NIS, you will recognize that NIS commands usually start with yp (for example: - **ypwhich**, **ypget**, and **ypset**).

NIS is a part of the network file system (NFS) software package that includes commands and daemons for NFS, NIS, and other services. On AIX Version 4.3.3 or later, NFS and NIS are no longer installed together as one package, but require installation of bos.net.nis.server or bos.net.nis.client. Each is independent and each is configured and administered individually.

NIS uses RPC as NFS does. For a brief discussion of RPC, see 7.1.2, “RPC” on page 151.

Support for NIS+ was introduced with AIX 4.3.3. NIS and NIS+ cannot be combined in a single environment.

10.1 Components of NIS

The NIS environment is composed of clients and servers; these are logically grouped together in a *domain*. Each domain has a particular set of characteristics. A domain is not restricted to a physical network layout. Neither should the NIS domain be confused with DNS domains. The NIS domain characteristics are defined in maps, or databases, that specify certain system information such as user names, passwords, and host names.

An NIS domain is a collection of systems that are logically grouped together. A group of hosts that share the same set of NIS maps belong to the same domain. The hosts are usually grouped together in the domain for a common reason, for example when working in the same group at a particular location. Each NIS host is assigned to a domain when the system starts. The domain name must be set on all hosts that intend to use NIS.

There is one master server per NIS domain, and the systems in the domain are typically on the same network. However, access to data served by NIS is independent of the relative locations of the NIS client and server. By design, you cannot add another master server to a domain because there would be two authoritative sources for the maps. To reduce master server load, you can add slave servers to the domain, or define more than one domain. Each new domain, of course, has its own master server.

In the following sections, the master server, slave servers, NIS daemons, and NIS maps are discussed.

10.1.1 NIS servers

An NIS server is a host that provides configuration information to other hosts on the network. Servers retain a set of maps and run the ypserv daemon, which processes requests from clients for information contained in maps. There are two types of servers: a master server and a slave server.

Master servers

A master server is the single host in a particular domain that maintains the authoritative maps. The master server may run the ypupdated daemon, which prompts slave servers to update their copies of the maps (all other hosts in the domain must obtain their map information from the master server, either directly or indirectly through a slave server). If ypupdated is used, secure NFS should be

configured as explained in *AIX 5L Version 5.1 System Management Guide: Communications and Networks,* on the Internet. The master server also runs the yppasswdd daemon, which processes requests to change users' passwords. When choosing a master server, the following criteria should be met:

- Accessible by the system administrator

  If something goes wrong, or if updates need to be made, it is easy to reach the master server.

- Stable

  It needs to be stable so systems that depend on it can rely on uninterrupted service.

- Accessible from the network

  Although networks can be complex with the presence of many gateways or bridges, the master server should be accessible from most systems on the network.

In a small domain, each host can access the master server directly. However, for a larger number of hosts in a domain, the master server can become overloaded. To balance the NIS processing load and provide services when the master server is unavailable, additional hosts can be designated as slave servers.

Slave servers

NIS slave servers act as intermediaries between clients and the master server by keeping exact replicas of the master server's maps. All changes to the maps are made on the master server. Then, the changes are propagated from the master server to the slave servers. Once a slave server is added to the domain, it is able to answer the same queries that the master is able to answer. In this way, slave servers can help the master server without violating the authority of the master server.

Slave servers also act as a backup in case the master server or the network fails. A client requesting information waits until a server responds. Adding slave servers increases the availability of information even if the master server is unavailable.

The number of slave servers in a domain should be balanced to achieve the desired level of availability and response time without adding the expense of copying data to too many systems. There should be at least one slave server for each domain, but normally there is one slave server per subnet, as shown in Figure 10-1 on page 226.

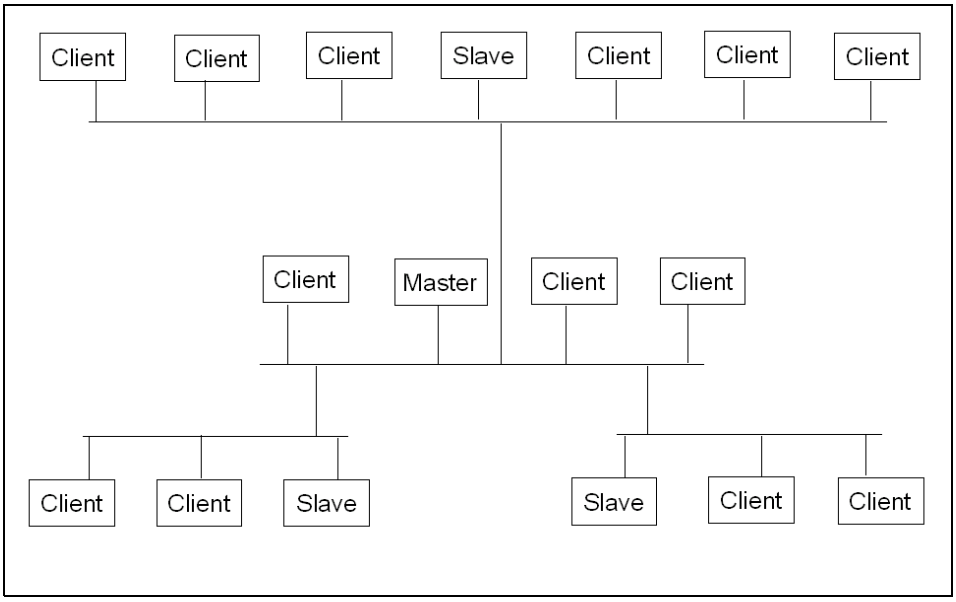


Figure 10-1 NIS domain

Clients

NIS clients make up the majority of hosts in an NIS domain. Clients run the ypbind daemon, which enables client processes to obtain information from a server. Clients do not maintain maps themselves, but rather query servers for system and user account information (clients do not make a distinction between querying the master server or a slave server). To access system information contained in a map, a client makes a Remote Procedure Call (RPC) to a server. The server searches its local database and returns the requested information to the client.

NIS clients locate the server by broadcasting on the networks that are directly connected to the client machine. Since these broadcast messages are not forwarded by network gateways, a slave server per subnet is convenient. If there is no NIS server that can be reached without using a network gateway, the client must specify a server when starting the ypbind daemon.

Note that every request for system information requires a server contact, and the speed of your network can affect the response time.

10.1.2 NIS daemons

There are only four NIS daemons included in the yp group. They are as follows:

```
# lssrc -g yp
Subsystem       Group        PID       Status
ypbind          yp                     inoperative
ypserv          yp                     inoperative
ypupdated       yp                     inoperative
yppasswdd       yp                     inoperative
```

As mentioned in previous sections, the client daemon, ypbind, is the daemon that has to establish connections. On the server side, the ypserv daemon is accepting and serving all yp requests. If NIS is used for centralized password management, then the **yppasswd** command on the client contacts the yppasswdd daemon. Finally, there is the ypupdated daemon that is used with Secure NFS. If Secure NFS is not used, this daemon should not be started (this is why the **startsrc -g yp** is not a good option in some environments). The relationship between NIS daemons is shown in Figure 10-2.

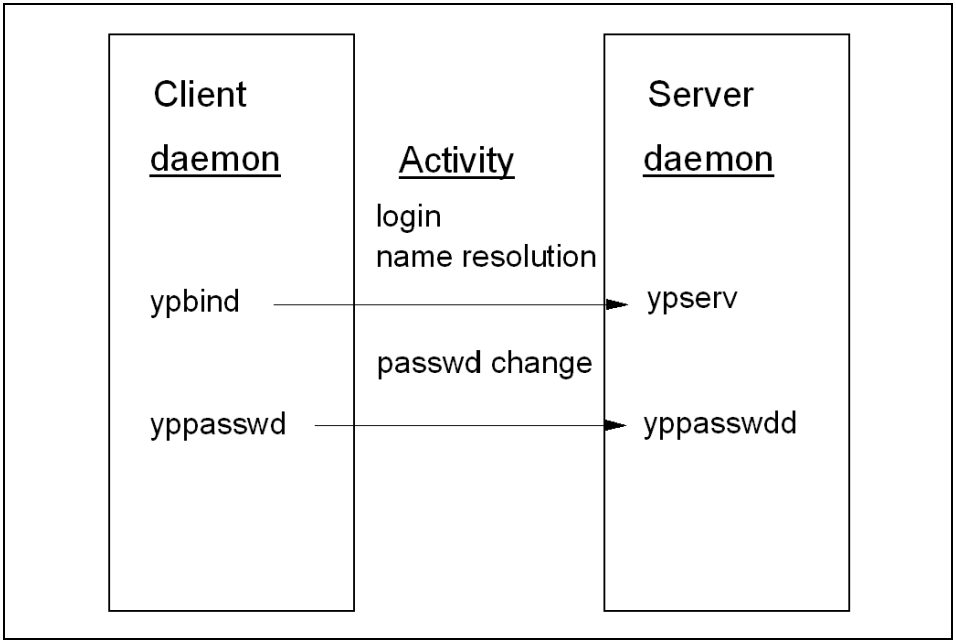


Figure 10-2 NIS daemons

10.1.3 NIS maps

NIS maps are databases that specify certain system information such as user names, passwords, and host names, in a database format called DBM. Most maps are constructed from a standard text files by associating an index key with a value. For example, the information in the master server's /etc/hosts file is used to create a map that uses each host name as a key and the IP address as the

value. The key and value pairs (also known as records) that are created from the entries in the /etc/hosts file comprise the hosts.byname map. In addition to the hosts.byname file, a hosts.byaddr file is also provided for reverse name resolution. For these two functions, name resolution and reverse name resolution, a total of four files are needed:

- hosts.byname.dir
- hosts.byname.pag
- hosts.byaddr.dir
- hosts.byaddr.pag

Files ending in .dir contain an index in the .pag files containing the key/value pair for faster searching.

Note: An NIS record has a maximum size of 1024 bytes. This limitation applies to all NIS map files. For example, a list of users in a group can contain a maximum of 1024 characters in single-byte character set file format. NIS cannot operate correctly with map files that exceed this maximum.

The most commonly used maps have nicknames that some commands can translate into map names. For example:

```
#ypcat hosts
```

The output you receive is actually the contents of the hosts.byname map, because there is no map called hosts in the NIS database. The **ypcat -x** command produces a list of available nicknames.

By default, the maps listed in Table 10-1 are created if their corresponding source files are available on the master server:

Table 10-1 NIS default map files

Map	Nickname	Source file
passwd.byname	passwd	/etc/passwd
passwd.byuid		
group.byname	group	/etc/group
group.bygid		
hosts.byaddr	hosts	/etc/hosts
hosts.byname		

Map	Nickname	Source file
ethers.byaddr	ether	/etc/ethers
ethers.byname		
networks.byaddr	networks	/etc/networks
networks.byname		
rpc.bynumber		/etc/rpc
services.byname	service	/etc/service
protocols.byname	protocols	/etc/protocols
protocols.bynumber		
netgroup		/etc/netgroups
netgroup.byhost		
netgroup.byuser		
bootparams		/etc/bootparams
mail.aliases	aliases	/etc/aliases
mail.byaddr		
publickey.byname		/etc/publickey
netid.byname		/etc/passwd , /etc/groups /etc/hosts /etc/netid
netmasks.byaddr		/etc/netmasks
ypservers		N/A

10.2 NIS configuration considerations

All NIS systems must meet these conditions before you start configuring NIS:

- TCP/IP must be running.
- The portmap daemon must be running.
- NFS must be installed.
- The bos.net.nis.server or bos.net.nis.client fileset must be installed. These filesets are not installed by default on AIX Version 4.3.3 or later.

10.2.1 Master server configuration

There are a few steps to perform on the server before starting to configure the clients.

If you want to increase the security in your NIS environment, you can use the /var/yp/securenets file. The ypserv daemon (used both on the master and the slave to answer ypbind requests) uses the /var/yp/securenets file and, if present, only responds to IP addresses in the range given. This file is read only when the ypserv daemon starts. To cause a change in /var/yp/securenets to take effect, you must kill and restart the daemon. The format of the file is netmask netaddr. For example:

```
255.255.255.0 9.3.240.0
```

Next, define a domain name.

Master server domain name definition

When starting to configure a server, you can start with defining the domain name. This can be done with the **smitty chypdom** command, as shown in Figure 10-3.

```
                    Change NIS Domain Name of this Host

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                      [Entry Fields]
* Domain name of this host                           [tcdomain]
* CHANGE domain name take effect                      both                    +
   now, at system restart or both?




F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure 10-3 Change NIS domain name screen in smitty

When choosing to set both or restart, as values in the CHANGE domain name take affect field, the domain name will be set in /etc/rc.nfs. This can also be done by editing the /etc/rc.nfs file directly, for example:

```
# Uncomment the following lines and change the domain
# name to define your domain (domain must be defined
# before starting NIS).
if [ -x /usr/bin/domainname ]; then
        /usr/bin/domainname tcdomain
```

If you choose to use the **domainname** command, the domain name will be set in the current login session, as shown in the following example:

```
#domainname tcdomain
```

This is an easy way to activate maps that are on other NIS domains. When using the **domainname** command without any arguments, it will return the current NIS domain. For example:

```
# domainname
tcdomain
```

Edit map source files

The next step is to edit the source files needed for map creation. In this example, the /etc/passwd and the /etc/hosts will be used. There is support for a multitude of map files, as shown in Table 10-1 on page 228.

The /etc/passwd file

The /etc/passwd file on the NIS master server needs to include all the user account information for all users on all NIS clients on the network that will belong to the NIS domain that the master is serving. It is also common that the server will be a client as well. This way, the server will have access to all information from the maps. If the NIS master server is to have local users (not to be administered through NIS), then another text input file may be used to build these maps.

If you choose to use a password file other than /etc/passwd to build the password map, you must specify to the yppasswdd daemon the path to that file. By default, the yppasswdd daemon changes passwords for entries in the /etc/passwd file. To change the default password file to another file, perform the following steps:

1. Edit the /etc/rc.nfs file, locate the following stanza and change the DIR statement so that it specifies the path to your alternate passwd file. For example, if you use the /var/yp/passwd file, the DIR statement should look like this:

   ```
   #Uncomment the following lines to start up the NIS
   #yppasswd daemon.
   DIR=/var/yp
   if [ -x /usr/lib/netsvc/yp/rpc.yppasswdd -a -f $DIR/passwd ]; then
           start rpc.yppasswdd /usr/lib/netsvc/yp/rpc.yppasswdd /etc/passwd -m
   fi
   ```

2. Enter the following three commands:

```
# stopsrc -s yppasswdd
0513-004 The Subsystem or Group, yppasswdd, is currently inoperative.
# chssys -s yppasswdd -a '/var/yp/passwd -m passwd'
0513-077 Subsystem has been changed.
# startsrc -s yppasswdd
0513-059 The yppasswdd Subsystem has been started. Subsystem PID is 23334.
```

The yppasswdd daemon will now use your alternate password file.

Figure 10-4 provides the values used for the sample system, with their copies of /etc/passwd and /etc/hosts before configuring NIS.

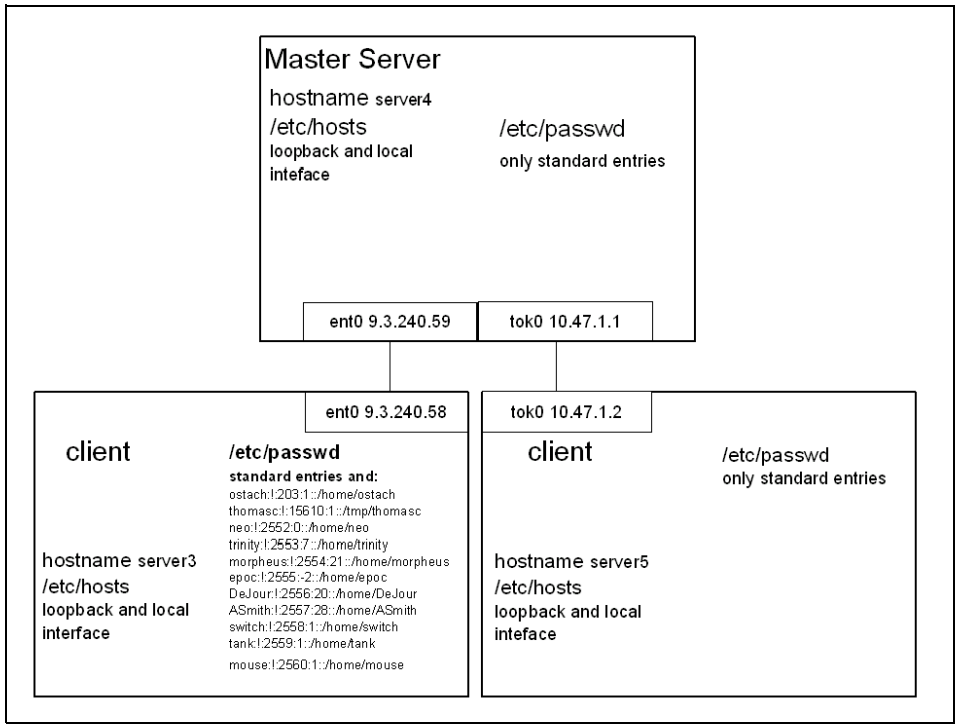


Figure 10-4 Hosts in example before NIS

In the example case, server3 has the most users, server4 will serve as a master, and server5 has no users defined in /etc/passwd. The entries for server3 users will be edited into the server4:/etc/passwd file.

The /etc/hosts file

Next the /etc/hosts file has to include all systems involved in the domain:

```
127.0.0.1               loopback localhost      # loopback (lo0) name/address
9.3.4.100       server4  server4.itsc.austin.ibm.com
9.3.4.29        server5
```

Now the server is prepared for defining the domain and editing the map source files. Before starting up the master server, consider what work must be done on slave servers and clients.

10.2.2 Client configuration considerations

In the example, the client will be fully dependent on the master server for password management and for name resolution. Therefore, all locally defined users may be removed from the /etc/passwd file, since they have been copied into the /etc/passwd file on the master server. After the removal of these, an *escape sequence*, `+::0:0:::`, should be added to the end of /etc/passwd. This escape sequence tells the system to use NIS for password handling. For example:

```
# echo +::0:0::: >> /etc/passwd
```

The /etc/hosts file needs only the loopback interface and the entry for the host. Next, make the system use NIS for name resolution, by editing /etc/netsvc.conf. For example:

```
# more /etc/netsvc.conf
host = nis,bind,local
```

You can override the default order by modifying the /etc/irs.conf configuration file and specifying the desired ordering.

The settings in the /etc/netsvc.conf configuration file override the settings in the /etc/irs.conf file. The NSORDER environment variable overrides the settings in the /etc/irs.conf and the /etc/netsvc.conf files.

Also remember to define your domain name, either by editing the /etc/rc.nfs, by using **`smitty chypdom`** as shown in Figure 10-3 on page 230 for permanent domain name setting, or by using the **`domainname`** command for temporary domain name setting.

10.2.3 Slave server configuration considerations

An important thing to remember is the slave server behaves like a client, in that it is dependent on the master server for updates of its records. In our example, the slave server will copy the /etc/passwd and /etc/hosts from the master, so the

same editing that was done on the client should be done on the slave server (/etc/passwd, /etc/hosts and /etc/netsvc.conf).

At this stage, the hosts in the example would appear as shown in Figure 10-5.

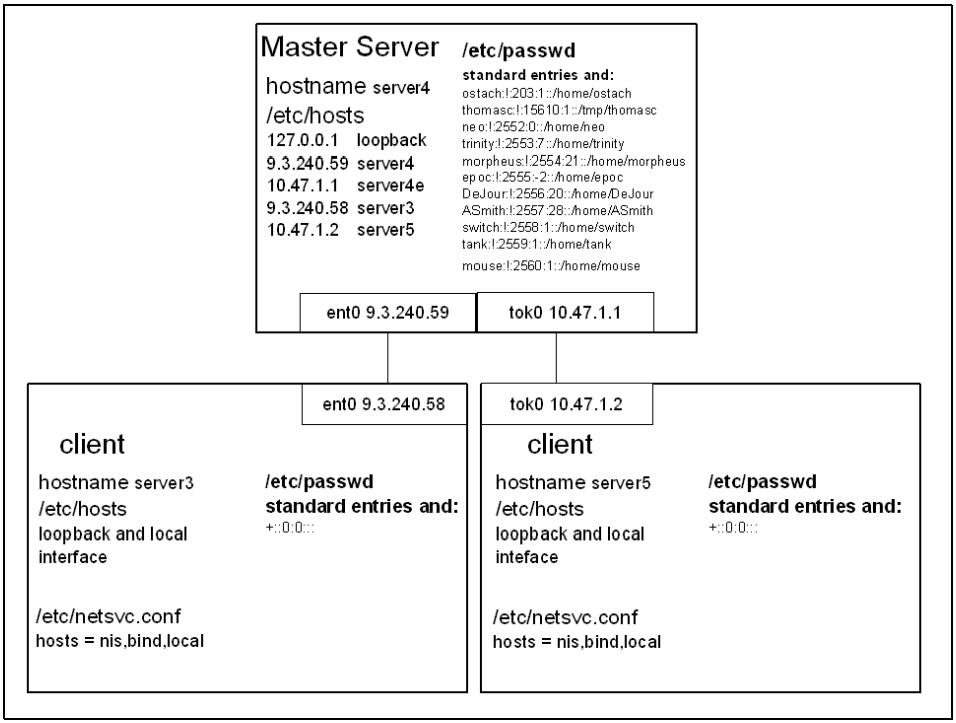


Figure 10-5 Hosts ready for NIS startup

Now the setup is ready. In the following section, NIS startup is discussed.

10.3 Starting NIS

Depending on the role the host has in the NIS domain, there are some differences in how to start NIS. In the following sections, the master, slave, and client startup are discussed.

10.3.1 Master server startup

To start NIS for the master, enter **`smitty mkmaster`**, as shown in Figure 10-6 on page 235.

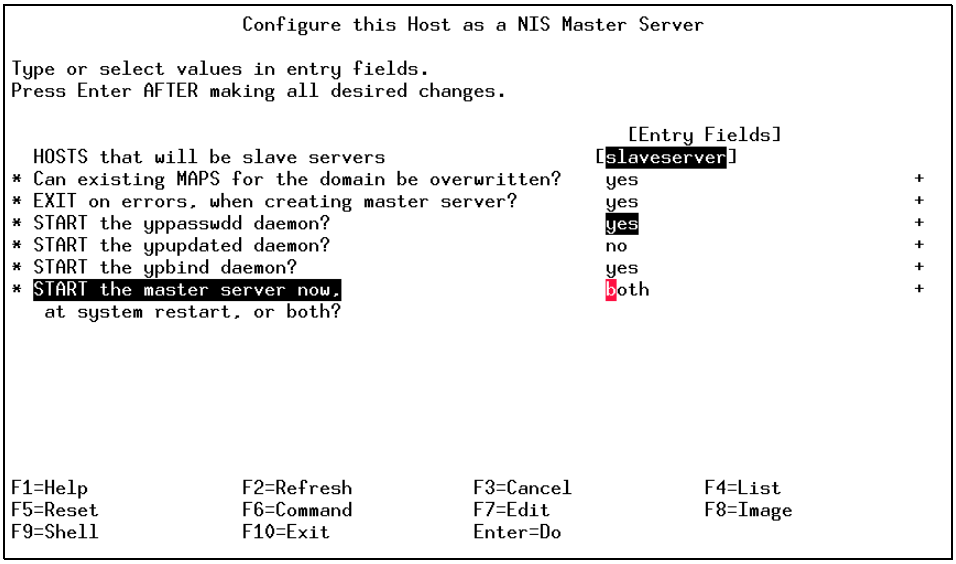


Figure 10-6 smitty mkmaster screen

The shortcut with smitty is that the /etc/rc.nfs will be updated, and the daemons chosen will be started after every reboot.

The startup can also be done through an interactive command called **ypinit,** (actually it is a script). The **ypinit** command does not update the etc/rc.nfs file; neither will it start the daemons (this must be done separately).

On the master server, the **ypinit** command should be started with the -m flag (for master). When the command is executed, you will have to answer a few questions. Among other things, it will prompt you for a list of slave servers:

```
# ypinit -m
Installing the NIS data base will require that you answer
a few questions.
Questions will all be asked at the beginning of the procedure.
Do you want this procedure to quit on non-fatal errors? [y/n: n] n
OK, please remember to go back and redo manually
whatever fails. If you don't, some part of the system
(perhaps the NIS itself) won't work.

At this point, we have to construct a list of the
hosts which will run NIS servers. server4 is in the list of NIS
server hosts. Please continue to add the names for the other
hosts, one per line. When you are done with the list, type a <control D>.
next host to add: server4
next host to add: ^D
```

```
The current list of NIS servers looks like this:

server4

Is this correct? [y/n: y] y
There will be no further questions. The remainder
of the procedure should take 5 to 10 minutes.
Building /var/yp/tcdomain/ypservers...
Running /var/yp/Makefile...
updated passwd
updated group..........
```

The **ypinit -m** command will call the **makedbm** command, which will create the database format file, the actual map file, and place these by default in /var/yp/<domainname>. In this example, the target directory will be /var/yp/tcdomain. The target directory can be changed by editing /var/yp/Makefile.

The **ypinit** command is dependent on the existence of the input files listed in Table 10-1 on page 228, but the database file ypservers does not have a standard input file like the rest of the map files. If you want to update the ypservers map file (for example, after adding another slave server to the domain), you need to directly use the **makedbm** command, as in the following example:

```
# cd /var/yp
# (makedbm -u tcdomain/ypservers ; echo server1) | makedbm - ypservers
```

In the previous command example, the -u flag will undo the DBM file. It prints out a DBM file one entry per line, with a single space separating keys from values. In this instance, the -u output, as well as the line echoed - server1, will be piped into the next **makedbm** command rather than being directed to the display. By doing this, a new ypserver map is created including the new slave server - server1.

After the **ypinit -m** command, the /var/yp/tcdomain includes the following maps:

```
# ls
group.bygid.dir         mail.byaddr.dir         protocols.bynumber.dir
group.bygid.pag         mail.byaddr.pag         protocols.bynumber.pag
group.byname.dir        netid.byname.dir        publickey.byname.dir
group.byname.pag        netid.byname.pag        publickey.byname.pag
hosts.byaddr.dir        passwd.byname.dir       rpc.bynumber.dir
hosts.byaddr.pag        passwd.byname.pag       rpc.bynumber.pag
hosts.byname.dir        passwd.byuid.dir        services.byname.dir
hosts.byname.pag        passwd.byuid.pag        services.byname.pag
mail.aliases.dir        protocols.byname.dir    ypservers.dir
mail.aliases.pag        protocols.byname.pag    ypservers.pag
```

10.3.2 Slave server startup

After configuring the master server, you will configure hosts chosen to act as slave servers. Slave servers keep exact replicas of the master server's maps and share the processing burden by answering queries when the master server is busy or unavailable. Before starting the slave servers, the NIS master server must be configured and started. In the example, no slave server is configured.

When using subnets, a slave server should be configured on each subnet that has NIS clients for the given NIS domain. This allows clients to bind at startup without pointing out the IP address to the ypbind daemon.

Create NIS domain as described in “Master server domain name definition” on page 230. It is the same domain name as the master server.

You can now create the directory for this domain, start the NIS daemons, and obtain copies of the NIS maps from the master server, by using **smitty mkslave**, as shown in Figure 10-7.

```
                    Configure this Host as a NIS Slave Server

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                      [Entry Fields]
* HOSTNAME of the master server                      [server4]
* Can existing MAPS for the domain be overwritten?    yes                      +
* START the slave server now,                         both                     +
   at system restart, or both?
* Quit if errors are encountered?                     yes                      +

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure 10-7 smitty mkslave screen

The system takes a few minutes to perform several tasks. First, it runs the **ypinit -s <master>** command. It creates the directory /var/yp/<domainname>, where domainname is the domain name you defined earlier. Then it runs the **ypxfr** command to obtain the NIS maps from the master server. If the **ypinit** command exits successfully, the system uncomments the entries in the /etc/rc.nfs file for the ypserv and ypbind daemons. Finally, the system starts these daemons.

If this NIS slave server is not on the same IP network as the NIS master server (that is, a gateway router is positioned between the slave server and the master server), you must explicitly identify the NIS master server by using the **ypset** command. For example, enter the command:

```
# startsrc -s ypbind -a "scott_vetter_domain"
0513-059 The ypbind Subsystem has been started. Subsystem PID is 14696.
# ypset 9.3.4.100
```

where 9.3.4.100 is the IP address of the NIS master server.

If you want to use the command-line interface for a startup of the slave server, you first have to start up the ypbind daemon on the slave server to make it able to connect to the master server.

Next, use the **ypinit -s <master>** command. This command prompts you, just as in the case for **ypinit -m**, for various information and takes a few minutes to complete. For example:

```
# ypinit -s server4
```

Edit the /etc/rc.nfs file and uncomment the lines that use the **startsrc** commands to start these daemons. For example:

```
if [ -x /usr/etc/ypserv -a -d /etc/yp/`domainname` ]; then
   startsrc -s ypserv
fi
```

This should also be done for the ypbind daemon. By doing this, the slave server will be available after the next reboot.

Finally, the escape sequence should be added into the /etc/passwd file. If there are users to be locally administered, the escape sequence should be placed after the users that are to be administered locally. The /etc/passwd file will be sequentially scanned at login, and when finding the escape sequence, NIS will be used instead of local password verification.

At this stage, the ypserv daemon has not yet been started, although you prepared the system to start that daemon after restart. Start the daemon with:

```
# startsrc -s ypserv
```

10.3.3 NIS client startup

The client startup is the last configuration task. With the escape sequence in /etc/passwd and domain name set, you only need to start ypbind, which is the client daemon. For example:

```
# startsrc -s ypbind
```

```
0513-059 The ypbind Subsystem has been started. Subsystem PID is 27134.
# ypwhich
ypwhich: 1831-178 Domain tcdomain not bound.
# ypwhich
server4
```

This command sequence shows that the **ypwhich** command has not received an answer when executed directly after the startup of the ypbind daemon. This is because the broadcast on the subnet for an NIS server has not yet received an answer. When executed the next time, the binding is set up.

At this point, it is good to use the **ypcat** command to check the listings available (for example, which hosts are defined by the master server hosts.byname map), as follows:

```
# ypcat hosts
9.3.240.59      server4
9.3.240.58      server3
127.0.0.1               loopback localhost      # loopback (lo0) name/address
10.47.1.2       server5
10.47.1.1       server4e
```

The client setup is done.

If you administer passwords through NIS, you need to start the yppasswd daemon (named yppasswdd) on the master server. When doing this, it is good to remember that all password changes would be handled by the **yppasswd** command, as follows:

```
# yppasswd thomasc
Old NIS password:
thomasc's New password:
Enter the new password again:
```

One downside of using the **yppasswd** command is shown in the following output of the /etc/passwd file on the master server:

```
# more /etc/passwd
morpheus:*:2554:21::/home/morpheus:/usr/bin/ksh
anonymou:*:202:1::/home/ftp:/usr/bin/ksh
trinity:*:2553:7::/home/trinity:/usr/bin/ksh
thomasc:M.BHTz4w35RKQ:15610:1::/tmp/thomasc:/usr/bin/ksh
```

As you can see, the encrypted password is in /etc/passwd, not in /etc/security/passwd, as with local password management.

10.3.4 Managing NIS maps

System information, such as a new user account or a changed password, can require constant updating. Whenever you need to modify an NIS map, you should do so on the master server and then propagate the changes to the slave servers. The only exception to this rule is when users change their password with the **yppasswd** command. When changing a map, you need to start with editing the source file. For example, in editing /etc/hosts, add server1 (9.3.240.56) to the file.

Even though the source file has been edited, the NIS subsystem is not yet aware of the changes:

```
# ypcat hosts
9.3.240.59      server4
9.3.240.58      server3
127.0.0.1               loopback localhost      # loopback (lo0) name/address
10.47.1.2       server5
10.47.1.1       server4e
```

The map files must be rebuilt. This can be done either with **smitty mkmaps** or with the **make** command:

```
# cd /var/yp
# make hosts
0+1 records in.
0+1 records out.
updated hosts
pushed hosts
Target "hosts" is up to date.
```

Afterwards, the information as seen by the client will be up to date:

```
# ypcat hosts
9.3.240.59      server4
9.3.240.58      server3
9.3.240.56      server1
127.0.0.1               loopback localhost      # loopback (lo0) name/address
10.47.1.2       server5
10.47.1.1       server4e
```

The map is now changed, and the master server has requested that all the slave servers update their maps.

To manually propagate NIS maps from the master server to slave servers, you can choose to use the **ypxfr <mapname>** command at the slave server or use the **yppush <mapname>** command at the master server.

10.4 NIS configuration summary

- The master server runs the ypserv and yppasswdd daemons.
- The master server updates the slave servers with yppush.
- The slave servers runs the ypbind and ypserv daemons.
- The slave servers update maps with ypxfr.
- Clients do not have local maps.
- Clients request information from a master or slave server through the ypbind daemon.

10.5 Command summary

The following sections provide a list of the key commands discussed in this chapter. For a complete reference of the following commands, consult the AIX product documentation.

10.5.1 The ypbind command

The **ypbind** command enables client processes to bind, or connect, to an NIS server.

The syntax for **ypbind** is:

```
/usr/lib/netsvc/yp/ypbind [ -s -ypset -ypsetme ]
```

The commonly used flags are provided in Table 10-2.

Table 10-2 Commonly used flags of the ypbind command

Flags	Description
-ypset	Indicates the local host accepts **ypset** commands from local or remote hosts.
-ypsetme	Indicates that the local host accepts **ypset** commands only from the local host.

10.5.2 The ypset command

The **ypset** command directs a client machine to a specific server.

The syntax for **ypset** is:

```
ypset [ -V1 ] [ -d Domain ] [ -h Host ] Server
```

The commonly used flags are provided in Table 10-3 on page 242.

Table 10-3 Commonly used flags of the ypset command

Flags	Description
-d <domain>	Specifies a domain other than the default domain.
-h <host>	Sets the binding for the ypbind daemon on the specified host instead of on the local host. The host can be specified as a name or as an IP address.

10.5.3 The ypinit command

The **ypinit** command sets up NIS maps on a Network Information Services (NIS) server.

The syntax for **ypinit** is:

```
ypinit [ -o ] [ -n ] [ -q ] -m [ SlaveName ... ]
```

The commonly used flags are provided in Table 10-4.

Table 10-4 Commonly used flags of the ypinit command

Flags	Description
-m <slave name(s)>	Indicates that the local host is to be the NIS master. If the -q flag is used, the -m flag can be followed by the names of the machines that will be the NIS slave servers.
-q	Indicates that the **ypinit** command is to get arguments from the command line instead of prompting for input.
-s <MasterName>	Copies NIS maps from the server workstation you specify in the MasterName parameter.

10.5.4 The yppush command

The **yppush** command prompts the Network Information Services (NIS) slave servers to copy updated NIS maps.

The syntax for **yppush** is:

```
yppush [ -v ] [ -d Domain ] MapName
```

The commonly used flags are provided in Table 10-5 on page 243.

Table 10-5 Commonly used flags of the yppush command

Flags	Description
-d <domain>	Specifies a domain other than the default domain. The maps for the specified domain must exist.
-v	Displays messages as each server is called and then displays one message for each server's response (if you are using the Version 2 protocol). If this flag is omitted, the command displays error messages only.

ypxfr

The **ypxfer** command transfers a Network Information Services (NIS) map from an NIS server to a local host.

The syntax for **ypxfr** is:

```
ypxfr [ -f ] [ -c ] [ -d Domain ] [ -h Host ] [ -s Domain ] [ -C TID Program
IPAddress Port ] [ -S ] MapName
```

The commonly used flags are provided in Table 10-6.

Table 10-6 Commonly used flags of the ypxfr command

Flags	Description
-f	Forces the transfer to occur even if the version at the master is not more recent than the local version.
-d < domain>	Specifies a domain other than the default domain. The maps for the specified domain must exist.
-h <host>	Gets the map from the host specified, regardless of what the map says the master is. If a host is not specified, the **ypxfr** command asks the NIS service for the name of the master and tries to get the map from there. The Host variable can contain a name or an Internet address in the form a.b.c.d.

ypcat

The **ypcat** command prints out a Network Information Services (NIS) map.

The syntax for **ypcat** is:

```
ypcat [ -k ] [ -t ] [-d DomainName ] MapName
```

The commonly used flags are provided in Table 10-7 on page 244.

Table 10-7 Commonly used flags of the ypcat command

Flags	Description
-x	Displays the nickname translation table.
-k	Displays the keys for those maps in which the values are null or for which the key is not part of the value.

10.5.5 The yppasswd command

The **yppasswd** command changes your network password in Network Information Services (NIS).

The syntax for **yppasswd** is:

```
yppasswd [ -f [ Name ] | -s [ Name [ ShellProg ]] ]
```

The commonly used flags are provided in Table 10-8.

Table 10-8 Commonly used flags of the yppasswd command

Flags	Description
-f <name>	Changes user Name's gecos information in the NIS maps. Gecos information is general information stored in the /etc/passwd file.

10.6 Quiz

The following assessment questions help verify your understanding of the topics discussed in this chapter.

1. Which one of the following files determines whether host names are looked up at DNS or NIS first?
   A. /etc/irs.conf and /etc/hosts
   B. /etc/resolv.conf and /etc/hosts
   C. /etc netsvc.conf and /etc/hosts
   D. /etc/irs.conf and /etc/netsvc.conf
2. All of the following files affect DNS lookups except:
   A. /etc/hosts
   B. /etc/irs.conf
   C. /etc/resolv.conf
   D. /etc/netsvc.conf

3. All of the following are generally involved with looking up addresses for public Internet hosts except:

   A. DNS

   B. NIS

   C. An /etc/hosts file

   D. /etc/resolv.conf file

4. Which one of the following commands can be executed on an NIS slave server to transfer a NIS map from the NIS master server?

   A. **ypcat**

   B. **ypxfr**

   C. **yppush**

   D. **ypmatch**

5. By default, which one of the following names are most appropriate for the NIS map versions of the /etc/passwd file?

   A. password.dir and password.pag

   B. /etc/passwd.NIS and /etc/security/password.NIS

   C. /etc/passwd.byname and /etc/security/passwd.byuid

   D. password.byname.pag, password.byname.dir, password.byuid.pag, and password.byuid.dir

6. Which one of the following entries should be in the /etc/passwd file so password lookups will search the NIS maps?

   A. +::0:0:::

   B. *::0:0:::

   C. -::!0:0:::

   D. @::0:0:::

10.6.1 Answers

The following are the preferred answers to the questions provided in this section:

1. D
2. A
3. B
4. B
5. D
6. A

10.7 Exercises

The following exercises provide sample topics for self study. They will help ensure comprehension of this chapter.

1. Create a subnet with at least three hosts for this exercise. Set up one as the master server and set up at least two clients. If you have access to a fourth host on the test subnet, then set it up as a slave server.
2. Transfer all user accounts to the master server. Set up the /etc/passwd file on all clients to point out the use of NIS.
3. Update the master with a new user. Recreate the passwd map.
4. Use the **ypxfr** command to get an updated version of /etc/passwd from the master server.

11

Serial Line Internet Protocol

In this chapter the following topics are discussed:

- Setting up the hardware for a connection
- Configuring SLIP
- Activating and Deactivating SLIP

Serial Line Internet Protocol (SLIP) is the protocol designed to handle TCP/IP traffic when operating through a serial connection as shown in Figure 11-1 on page 248. It is commonly used on dedicated serial links and dial-up connections that operate at speeds of 1200 bps or higher.

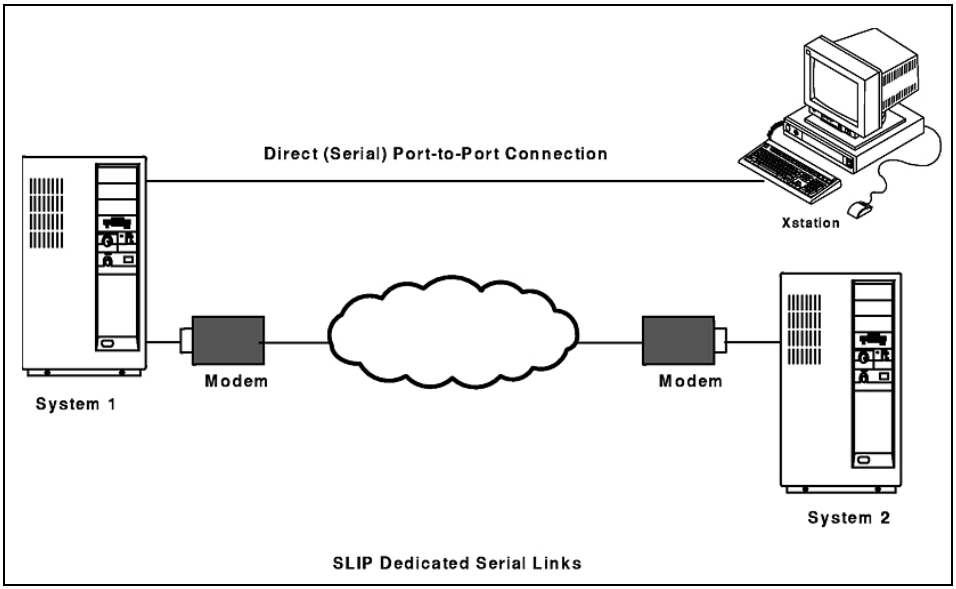


Figure 11-1 SLIP serial links

11.1 Setting up the serial port and modem

When setting up the serial link and modems, ensure it is done on both machines. The procedure is the same for both machines. This example assumes that there is a telephone line at each site and that both systems have all the hardware required to install this protocol. The UNIX-to-UNIX Copy Program (UUCP) must be installed on the system. To validate, enter:

```
# lslpp -f | grep bos.net.uucp
  bos.net.uucp 5.1.0.25
  bos.net.uucp 5.1.0.25
```

To begin setting up the TTY device, enter:

```
# smitty tty
```

Select the ADD a TTY option (Figure 11-2 on page 249), or if the port is already set up, choose the **Change / Show Characteristics of a TTY** option, and press Enter.

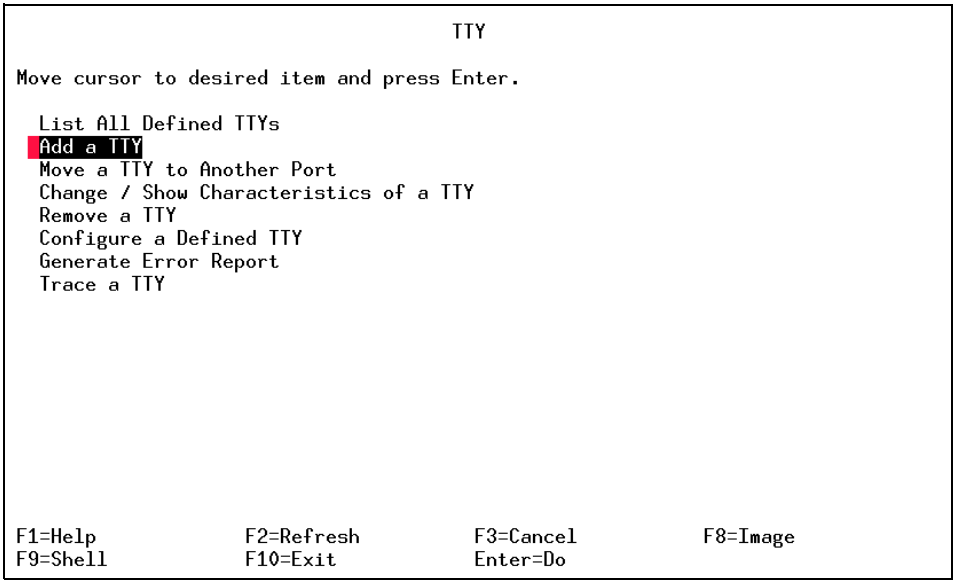


Figure 11-2 Smit TTY screen

Select the **tty rs232 Asynchronous Terminal** TTY type (Figure 11-3) and press Enter.

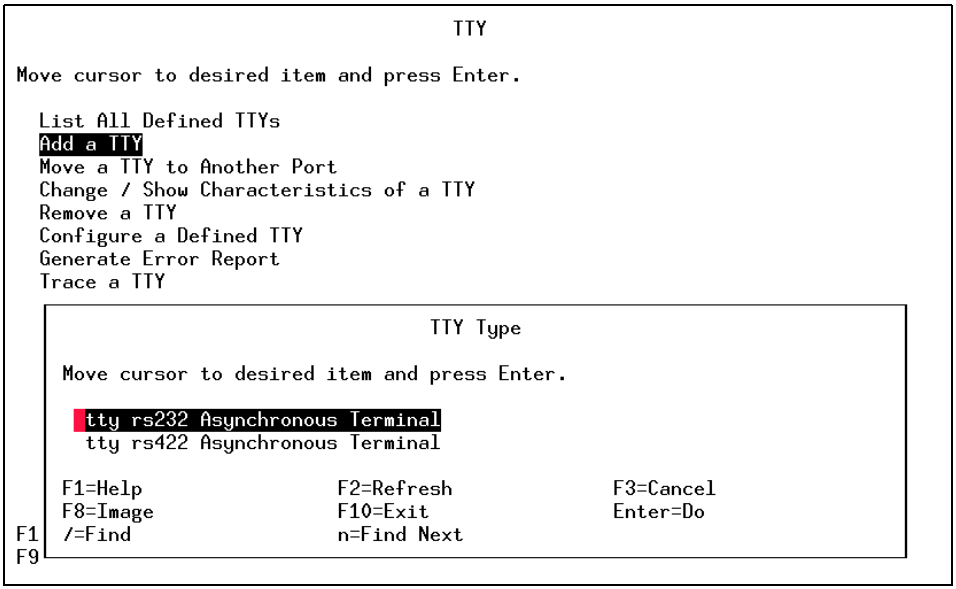


Figure 11-3 SMIT TTY option screen

Select the parent adapter for the TTY port as shown in Figure 11-4.

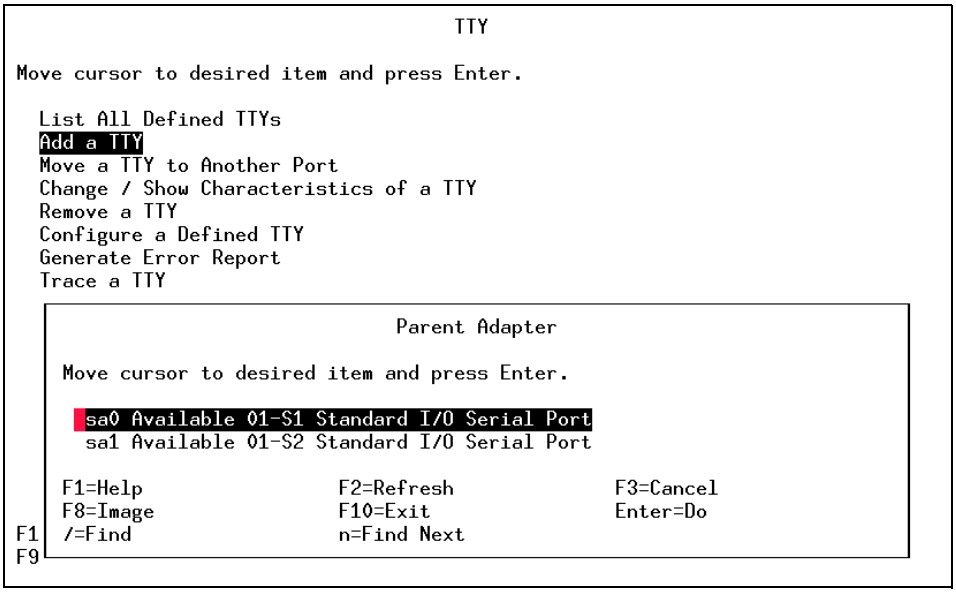


Figure 11-4 SMIT parent adapter option screen

In the Add a TTY option screen, select the port number (a list of available ports can be displayed by pressing the F4 key). There are a few things that need to be considered for the setup:

- Enable Login can be either Disable, Enable, Share, or Delay. For SLIP to work, it should be set to Disable on both systems.

  Disable Indicates no getty process is run on the port.

  Enable Indicates a getty process is run on the port.

  Share Indicates a getty process is run on the port in bi-directional mode. The getty process allows the port to be shared with other programs by waiting for an opening of the port to complete before attempting to get a lock on the TTY device. If an active process already owns the lock, the getty process lets that process own the TTY port until the lock goes away.

  Delay Indicates a getty process is run on the port in bi-directional mode. With the delay setting, no login herald is displayed until the getty process receives a keystroke from the user.

- FLOW CONTROL should be set to either RTS or none. The default option for this is XON. The term “flow control” is used to describe the method by which a

serial device controls the amount of data being transmitted to itself. The selectable types of flow control used with TTYs are:

XON/XOFF (Transmission ON/Transmission OFF) flow control involves the sending of data transmission control characters along the data stream. For this reason, it is often referred to as software flow control.

XON/IXANY XON/IXANY is similar to the XON/XOFF software flow control, except that any character received causes the data transmission to resume.

RTS/CTS Ready To Send/Clear To Send is sometimes called pacing or hardware handshaking. The term hardware handshaking comes from the use of cabling and voltages as a method of data transmission control. Unlike XON/XOFF, which sends control characters in the data stream, RTS/CTS uses positive and negative voltages along dedicated pins or wires in the device cabling.

NONE NONE disables all flow control and overrides any other flow control that was selected.

Press Enter to continue, and a screen like Figure 11-5 is shown completing the steps.

```
                                Add a TTY

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

[TOP]                                                 [Entry Fields]
  TTY type                                           tty
  TTY interface                                      rs232
  Description                                        Asynchronous Terminal
  Parent adapter                                     sa0
* PORT number                                        [0]                  +
  Enable LOGIN                                       disable              +
  BAUD rate                                          [9600]               +
  PARITY                                             [none]               +
  BITS per character                                 [8]                  +
  Number of STOP BITS                                [1]                  +
  TIME before advancing to next port setting         [0]                  +#
  TERMINAL type                                      [dumb]
  FLOW CONTROL to be used                            [rts]                +
[MORE...31]

F1=Help            F2=Refresh         F3=Cancel          F4=List
F5=Reset           F6=Command         F7=Edit            F8=Image
F9=Shell           F10=Exit           Enter=Do
```

Figure 11-5 SMIT Add a TTY option screen

Press F10 to exit back to the command prompt.

The /etc/uucp/Devices file needs to be edited to set up the new modem. Add the following line to the file:

```
Direct tty0 - 9600 direct
```

When inserting this line, it should be the first modem description line in the Devices file. The /etc/uucp/Devices file contains information about the devices on the local system that can establish a connection to a remote computer using the Basic Networking Utilities (BNU) program. This file includes information for hardwired, telephone, and TCP/IP communication links.

Note: To use baud rates higher than 38400, specify a baud rate of 50 in the /etc/uucp/Devices file for the desired TTY, then change the SMIT configuration for that TTY to reflect the actual baud rate desired.

The Devices file must contain a description of each device on the local system that can establish a remote connection using the BNU program. Each line in the Devices file includes the following fields:

Type	Typically specifies the type of hardwired or automatic calling unit (ACU) device.
Line	Specifies the device name for the port.
Line2	Specifies the dialer name if the Line entry specifies an 801 dialer.
Class	Typically specifies the transmission speed.
Dialer-Token Pairs	Specifies a particular type of autodialer (modem) and the token (a defined string of characters) that is passed to the dialer.

Run the **cu** command and set up the modem. After each typed line, the modem should display a status of OK once the Enter key has been pressed, as follows:

```
# cu -ml tty0
Connected
ate1
OK
atq0
OK
at&f
OK
at&d2
OK
ats0=1
OK
ats9=12
OK
```

```
at&c1
OK
at&w
OK
~[server2].

The connection is ended.
```

The **at** command settings that are used do the following:

- E1 turns the echo mode on.
- Q0 enables the displaying of result codes.
- &F is used to reset the modem to factory defaults.
- &D2 sets DTR.
- S0 and S9 set register values.
- &C1 set carrier.
- &W writes the settings to the modem.
- The tilde-period ends the connection.

The modem can be tested in the following manner:

```
# cu -ml tty0
Connected
atdt ### ####
```

This will connect you to the remote system where *### ####* is the remote system's telephone number.

```
AIX Version 4
(C) Copyrights by IBM and by others 1982, 1996.
login: root
root's Password:
*******************************************************************************
*                                                                             *
*                                                                             *
*  Welcome to AIX Version 4.3!                                                *
*                                                                             *
*                                                                             *
*  Please see the README file in /usr/lpp/bos for information pertinent to    *
*  this release of the AIX Operating System.                                  *
*                                                                             *
*                                                                             *
*******************************************************************************
Last login: Tue Jun 27 11:51:59 CDT 2000 on /dev/tty0

#
```

11.2 Configuring the SLIP connection

The modem is working and has been tested; now the system needs to be set up for the SLIP connection. This procedure needs to be done on both systems.

Set up the SLIP attachment as follows:

```
# smit mkinet
```

Select the **Add a Serial Line INTERNET Network Interface** option and press Enter as shown in Figure 11-6.

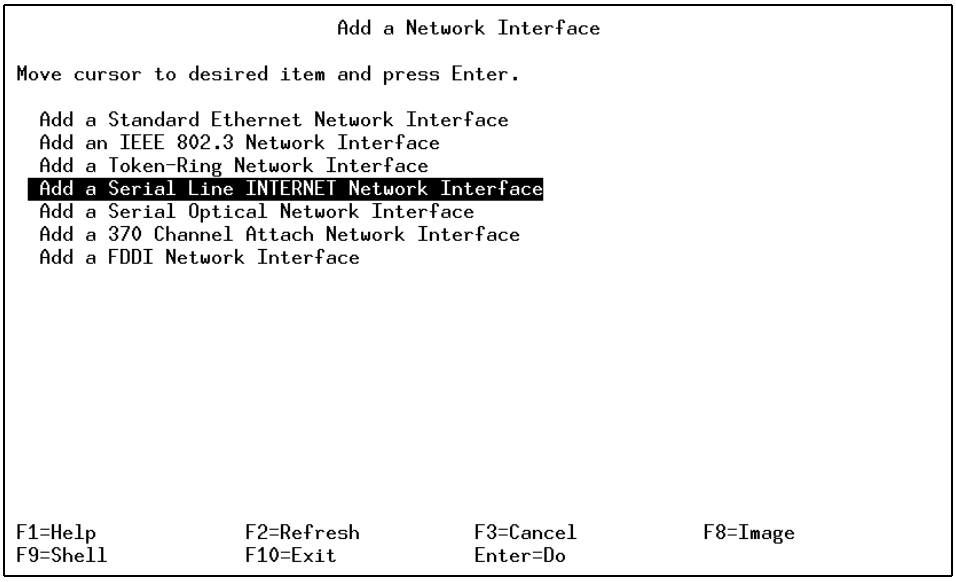


Figure 11-6 SMIT Add a Network Interface screen

Select the TTY device that has been configured for SLIP and press Enter. In this example, it is tty0, as shown in Figure 11-7 on page 255.

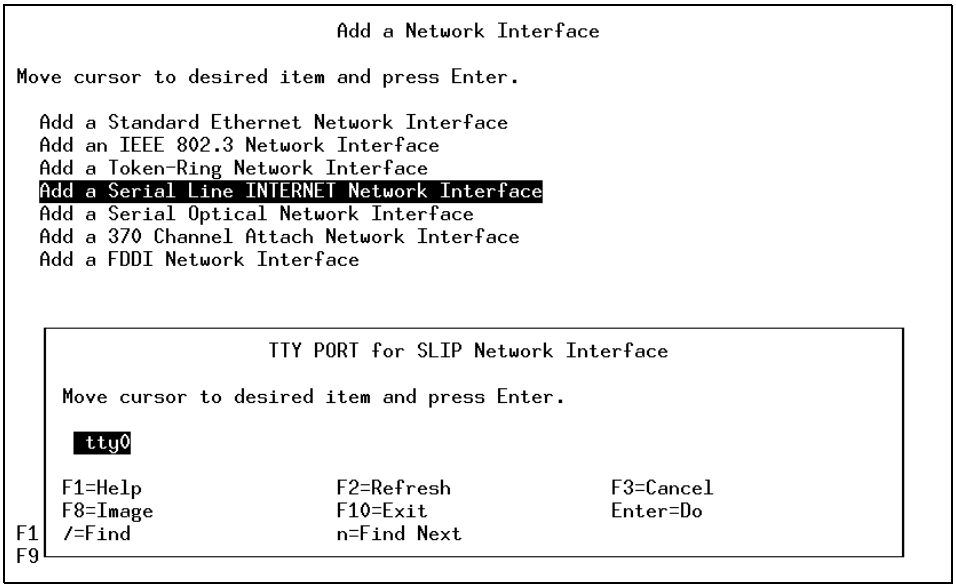


Figure 11-7 SMIT TTY PORT for SLIP Network Interface options screen

Insert the INTERNET ADDRESS and the DESTINATION Address. On the remote server, the addresses will be 10.11.12.2 for the INTERNET ADDRESS and 10.11.12.1 for the DESTINATION Address. The BAUD RATE and DIAL STRING fields are left empty, as the baud rate and number to be dialed are set up using the **slattach** command. These settings are shown in Figure 11-8 on page 256.

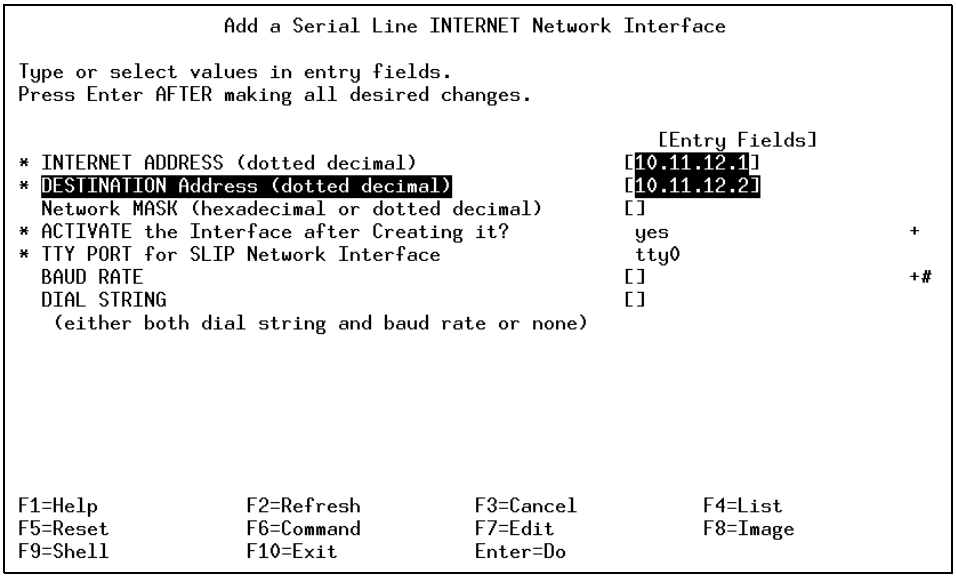


Figure 11-8 SMIT Add a Serial Line INTERNET Network Interface screen

After the command has been successfully completed, press F10 to exit.

Edit the /etc/hosts file and insert the IP addresses and server names for the servers. These names must be unique and cannot be shared with other servers.

```
10.11.12.1              slipserver1
10.11.12.2              slipserver2
```

Run the following command on the remote server to set up the SLIP attachment:

```
# slattach tty0
```

Run the following command on the local server to set up the SLIP attachment:

```
# slattach tty0 9600 '"" AT OK ATDT####### CONNECT ""'
```

The above string is interpreted as "Use `tty0` at `9600` baud, send `AT` and you should get back an `OK`, dial `###-####` and I should get a `CONNECT` back."

```
slattach: Device /dev/tty0 successfully opened.
slattach:using slip interface sl0 for /dev/tty0
slattach: The /dev/tty0 connection is established.
```

The **`netstat`** command can be used to display the link between the two systems, as follows:

```
# netstat -in
Name  Mtu   Network     Address            Ipkts Ierrs    Opkts Oerrs  Coll
```

```
lo0   16896 link#1                              137     0      137     0     0
lo0   16896 127         127.0.0.1               137     0      137     0     0
lo0   16896 ::1                                 137     0      137     0     0
tr0   1492  link#2      0.4.ac.61.9d.c5       15762     0     3726     0     0
tr0   1492  9.3.240     9.3.240.57            15762     0     3726     0     0
sl0   1006  link#3                              483     0      582     0     0
sl0   1006  10          10.11.12.1              483     0      582     0     0
```

On the remote server, the output appears as follows:

```
# netstat -in
Name  Mtu   Network     Address            Ipkts Ierrs    Opkts Oerrs  Coll
lo0   16896 link#1                           131     0      131     0     0
lo0   16896 127         127.0.0.1              131     0      131     0     0
lo0   16896 ::1                              131     0      131     0     0
sl0   1006  link#2                           628     4      520     0     1
sl0   1006  10          10.11.12.2           628     4      520     0     1
```

To test the remote server, the **ping** command can be used:

```
# ping slipserver2
PING slipserver2: (10.11.12.2): 56 data bytes
64 bytes from 10.11.12.2: icmp_seq=0 ttl=255 time=250 ms
64 bytes from 10.11.12.2: icmp_seq=1 ttl=255 time=250 ms
64 bytes from 10.11.12.2: icmp_seq=2 ttl=255 time=250 ms
64 bytes from 10.11.12.2: icmp_seq=3 ttl=255 time=250 ms
64 bytes from 10.11.12.2: icmp_seq=4 ttl=255 time=250 ms
^C
----slipserver2 PING Statistics----
6 packets transmitted, 5 packets received, 16% packet loss
round-trip min/avg/max = 250/250/250 ms
```

The remote SLIP server is now available for use.

In the following example, a FTP transfer between the machines will be done. This will display a SLIP connection that can be used as a normal TCP/IP system would be used.

From the **slipserver1**, type the following and then log in to the remote system:

```
# ftp slipserver2
Connected to slipserver2.
220 localhost FTP server (Version 4.1 Mon Jul 26 19:58:48 CDT 1999) ready.
Name (slipserver2:root):
331 Password required for root.
Password:
230 User root logged in.
```

Change directory to the directory to get files from and to put a file into:

```
ftp> cd /home/user01
```

```
250 CWD command successful.
ftp> pwd
257 "/home/user01" is current directory.
```

List the contents of the directory:

```
ftp> ls
200 PORT command successful.
150 Opening data connection for ..
userprog1
userfile1
226 Transfer complete.
```

Change the file transfer type to binary for the userprog1 file:

```
ftp> binary
200 Type set to I.
```

Get both the files for transfer using the **mget** as opposed to the **get** command:

```
ftp> mget /home/user01/* /home/myuser1
mget userfile1? y
200 PORT command successful.
150 Opening data connection for userfile1 (416 bytes).
226 Transfer complete.
429 bytes received in 0.6014 seconds (0.6967 Kbytes/s)
local: userfile1 remote: userfile1
mget userprog1? y
200 PORT command successful.
150 Opening data connection for userprog1 (1871 bytes).
226 Transfer complete.
1925 bytes received in 2.197 seconds (0.8556 Kbytes/s)
local: userprog1 remote: userprog1
```

Put a single file from the local host onto the remote host:

```
ftp> put /home/myuser1/myfile02 /home/user01/myfile02
200 PORT command successful.
150 Opening data connection for /home/user01/myfile02.
226 Transfer complete.
1310 bytes sent in 0.006891 seconds (185.6 Kbytes/s)
local: /home/myuser1/myfile02 remote: /home/user01/myfile02
```

List the contents of the remote directory:

```
ftp> ls -l
200 PORT command successful.
150 Opening data connection for /bin/ls.
total 8
-rw-r-----   1 root     system      1310 Jun 28 14:49 myfile02
-rw-r--r--   1 root     system       416 Jun 28 14:38 userfile1
-rw-r--r--   1 root     system      1871 Jun 28 14:37 userprog1
```

```
226 Transfer complete.
```

Exit the system:

```
ftp> bye
221 Goodbye.
```

11.2.1 Deactivating the SLIP connection

To temporarily deactivate the slip connection, use the following commands:

```
# ifconfig sl0 down
```

Check for any currently running **slattach** processes:

```
# ps -ef | grep slat
    root 10874     1   0 11:44:11      0  0:00 slattach tty0 9600 ""
    root 11112  2526   0 11:53:57  pts/0  0:00 grep slat
```

Kill the **slattach** process:

```
# kill 10874
```

Note: Do not use **kill -9** to stop the slattach process. This may cause problems and cause a system crash. If the system does crash, the only way to fix this is to remove the SLIP and the TTY using SMIT and then reconfigure the TTY and SLIP using SMIT again.

11.2.2 Activating a SLIP connection

To reactivate the SLIP connection, run the following commands. The **ifconfig** command will have to be run on both machines.

```
# ifconfig sl0 up
```

Run the following command on the remote server to set up the SLIP attachment:

```
# slattach tty0
```

Run the following command on the local server to set up the SLIP attachment:

```
# slattach tty0 9600 '"" AT OK ATDT####### CONNECT ""'
```

11.3 The slattach command

The **slattach** command assigns a TTY line to a network interface.

The **slattach** command is run by the /etc/rc.net file during system startup to automatically configure any Serial Line Internet Protocol (SLIP) network interfaces defined by the System Management Interface Tool (SMIT).

The command syntax for the **slattach** command is as follows:

```
slattach TTYName [ BaudRate  DialString [ DebugLevel ]  ]
```

The commonly used flags are provided in Table 11-1.

Table 11-1 Commonly used flags of the slattach command

Parameter	Description
BaudRate	Sets the speed of the connection. The default speed is 9600.
DebugLevel	Sets the level of debug information desired. A number from 0 through 9 may be specified. A value of 0 specifies no debug information; a value of 9 specifies the most debug information. The default value is 0.
DialString	Specifies a string of expect/respond sequences using the Basic Networking Utility (BNU)/UNIX to UNIX Copy Program (UUCP) chat syntax.
TTYName	Specifies a TTY line. This string is in the form ttyxx or /dev/ttyxx .

11.4 Files

Following are examples of files used for SLIP:

/etc/uucp/Devices This Devices file is used in conjunction with the Dialers file.

/etc/uucp/Systems Lists and describes remote systems accessible to a local system.

/etc/uucp/Dialers Lists modems used for Basic Networking Utilities (BNU) remote communications links.

/etc/uucp/Dialcodes Contains the initial digits of telephone numbers used to establish remote connections over a phone line.

/etc/uucp/Sysfiles Let system administrators specify alternate Systems, Devices and Dialers files to replace the default files in the /etc/uucp directory.

11.5 Quiz

The following assessment questions help verify your understanding of the topics discussed in this chapter. Use the following figure to help answer the first question.

Symptoms Of The Problem

One of the modems on your customer's modem server is having trouble when attempts are made to connect from other systems. The Systems Administrator has made the following observations:

- Whenever someone dials into the system, they get the 'CONNECTED' message from the modem, and are disconnected as soon as a key is depressed.
- If the modem is set to ignore the drop in DTR, the connection is not dropped, but the screen contains garbage characters, and login is not possible.
- If the same modem is used to dial out to any other system which cannot connect, there is no problem getting a good login screen.
- When SLIP is set up between two machines and the remote system is dialing into the problem machine, and connects successfully at 14,400 BPS, the remote system cannot be successfully reached with *ping*. The problem machine (local) starts *slattach* with the following options:

  slattach tty0

netstat

The following output of the *netstat -I sl0* command shows the status of the SLIP interface after 10 *ping* packets were sent, but none received:

```
Name Mtu  Network     Address       Ipkts Ierrs Opkts Oerrs Coll
sl0  1006 <Link>                    0     34    10    0     0
sl0  1006 48.250.106 48.250.106.1 0     34    10    0     0
```

/usr/lib/uucp/Devices

The */usr/lib/uucp/Devices* file has the following entries for 'tty0':

```
ACU tty0 - 38400 hayes \D
ACU tty0 - Any hayes \D
IBMINFONET tty0 - Any hayes
Direct tty0 - Any direct
Direct tty0 - 38400 direct
```

lsattr

The output of *lsattr -l tty0 -F "attribute = value"*

```
modules          = ldterm,tioc
speed            = 19200
kill             = ^u
erase            = ^h
eof              = ^d
dsusp            = ^y
susp             = ^z
lnext            = ^v
start            = ^q
stop             = ^s
werase           = ^w
parity           = none
bpc              = 8
stops            = 1
term             = dumb
login            = share
runmodes         = hupcl,cread,brkint,icrnl,opost,tab3,onlcr,
  isig,icanon,echo,echoe,echok,echoctl,echoke,imaxbel,iexten
quit             = ^\
intr             = ^c
reprint          = ^r
logmodes         = hupcl,cread,echoe,cs8,ixon,ixoff
eol              = ^@
eol2             = ^?
discard          = ^o
autoconfig       = available
imap             = none
omap             = none
csmap            = sbcs
tbc16            = 16
tbc64            = 64
timeout          = 0
logger           =
flow_disp        = xon
open_disp        = dtropen
rtrig            = 3
xprint_priority = 30
xprint_on_str    = \033[5I
xprint off str   = \033[4i
```

Figure 11-9 Customer information

1. A modem and TTY need to be configured for a SLIP link that can be initiated from either direction. Given the information provided in Figure 11-9 on page 261, which one of the following is the best procedure to accomplish the configuration?

   A. Program the modem to lock its DTE speed to 38400, set the speed attribute of the TTY to 38400, and change the first Direct entry for tty0 in /usr/lib/uucp/Devices to 38400 baud.

   B. Set the speed attribute of the TTY to 38400, turn on software flow control in the TTY attributes, and change the first Direct entry for tty0 in /usr/lib/uucp/Devices to 38400 baud.

   C. Change the speed attribute of tty0 to 38400, change the parity to even, bits per character to 7, with 1 stop bit, and use a baud rate of 38400 when starting the SLIP link in both the local and the remote machine.

   D. Make sure the cable is a null modem cable, lock the baud rate of the modem to 38400, change the speed of tty0 to 38400, and start the communications link with a baud rate of 38400.

2. A company decides to add a dial-in modem on a native serial port. Which one of the following filesets should be loaded to enable configuration and use of the modem?

   A. bos.mh

   B. bos.net.uucp

   C. bos.net.tcp.client

   D. bos.net.tcp.server

3. A Hayes compatible modem is being attached to a TTY port and will be shared by all users for both dial-in and dial-out connectivity. What is the best default value for the SMIT TTY Enable LOGIN field?

   A. SHARE

   B. ENABLE

   C. DELAY

   D. HAYES

4. The host's integrated TTY port is DTE. The modem async port is DCE. What is the correct cable?

   A. Straight-thru cable

   B. Roll-over cable

   C. Null-modem cable

   D. RJ-45 cable

11.5.1 Answers

The following are the preferred answers to the questions provided in this section:

1. A
2. B
3. C
4. A

11.6 Exercises

The following exercises provide sample topics for self study. They will help ensure comprehension of this chapter.

1. In the /etc/uucp/Devices file, what line must be inserted for the modem?
2. Make the changes in the /etc/uucp/Devices file and then set up the modem. What command is used to set up the modem?
3. What command can be used to check for problems with the SLIP connection?
4. What fileset needs to be installed for the modem to be connected?
5. Must the modem port be enabled for SLIP to work?
6. Set up the SLIP attachment. What command was used to configure the remote and the local hosts?

Abbreviations and acronyms

ABI	Application Binary Interface
AC	Alternating Current
ACL	Access Control List
ADSM	ADSTAR Distributed Storage Manager
ADSTAR	Advanced Storage and Retrieval
AFPA	Adaptive Fast Path Architecture
AFS	Andrew File System
AH	Authentication Header
AIX	Advanced Interactive Executive
ANSI	American National Standards Institute
APAR	Authorized Program Analysis Report
API	Application Programming Interface
ARP	Address Resolution Protocol
ASCI	Accelerated Strategic Computing Initiative
ASCII	American National Standards Code for Information Interchange
ASR	Address Space Register
ATM	Asynchronous Transfer Mode
AuditRM	Audit Log Resource Manager
AUI	Attached Unit Interface
AWT	Abstract Window Toolkit
BCT	Branch on CounT
BFF	Backup File Format
BI	Business Intelligence
BIND	Berkeley Internet Name Daemon
BIST	Built-In Self-Test
BLAS	Basic Linear Algebra Subprograms
BLOB	Binary Large Object
BLV	Boot Logical Volume
BOOTP	Boot Protocol
BOS	Base Operating System
BSC	Binary Synchronous Communications
CAD	Computer-Aided Design
CAE	Computer-Aided Engineering
CAM	Computer-Aided Manufacturing
CATE	Certified Advanced Technical Expert
CATIA	Computer-Graphics Aided Three-Dimensional Interactive Application
CCM	Common Character Mode
CD	Compact Disk
CDE	Common Desktop Environment
CDLI	Common Data Link Interface
CD-R	CD Recordable
CD-ROM	Compact Disk-Read Only Memory
CE	Customer Engineer
CEC	Central Electronics Complex
CFD	Computational Fluid Dynamics
CGE	Common Graphics Environment
CHRP	Common Hardware Reference Platform

CISPR International Special Committee on Radio Interference

CLIO/S Client Input/Output Sockets

CLVM Concurrent LVM

CMOS Complimentary Metal-Oxide Semiconductor

CMP Certificate Management Protocol

COFF Common Object File Format

COLD Computer Output to Laser Disk

CPU Central Processing Unit

CRC Cyclic Redundancy Check

CSID Character Set ID

CSR Customer Service Representative

CSS Communication Subsystems Support

CSU Customer Set-Up

CWS Control Workstation

DAD Duplicate Address Detection

DAS Dual Attach Station

DASD Direct Access Storage Device

DAT Digital Audio Tape

DBCS Double Byte Character Set

DBE Double Buffer Extension

DC Direct Current

DCE Distributed Computing Environment

DDC Display Data Channel

DDS Digital Data Storage

DE Dual-Ended

DES Data Encryption Standard

DFL Divide Float

DFP Dynamic Feedback Protocol

DFS Distributed File System

DHCP Dynamic Host Configuration Protocol

DIMM Dual In-Line Memory Module

DIP Direct Insertion Probe

DIT Directory Information Tree

DIVA Digital Inquiry Voice Answer

DLT Digital Linear Tape

DMA Direct Memory Access

DMT Directory Management Tool

DN Distinguished Name

DNS Domain Name Service/System

DOE Department of Energy

DOI Domain of Interpretation

DOS Disk Operating System

DPCL Dynamic Probe Class Library

DRAM Dynamic Random Access Memory

DS Differentiated Service

DSA Dynamic Segment Allocation

DSE Diagnostic System Exerciser

DSMIT Distributed SMIT

DSU Data Service Unit

DTE Data Terminating Equipment

DW Data Warehouse

EA Effective Address

EC Engineering Change

ECC Error Checking and Correcting

EEPROM Electrically Erasable Programmable Read Only Memory

EFI Extensible Firmware Interface

EHD Extended Hardware Drivers

EIA Electronic Industries Association

EISA Extended Industry Standard Architecture

ELA	Error Log Analysis
ELF	Executable and Linking Format
EMU	European Monetary Union
EOF	End of File
EPOW	Environmental and Power Warning
ERRM	Event Response resource manager
ESID	Effective Segment ID
ESP	Encapsulating Security Payload
ESSL	Engineering and Scientific Subroutine Library
ETML	Extract, Transformation, Movement, and Loading
F/C	Feature Code
F/W	Fast and Wide
FC	Fibre Channel
FCAL	Fibre Channel Arbitrated Loop
FCC	Federal Communication Commission
FCP	Fibre Channel Protocol
FDDI	Fiber Distributed Data Interface
FDPR	Feedback Directed Program Restructuring
FDX	Full Duplex
FIFO	First In/First Out
FLASH EPROM	Flash Erasable Programmable Read-Only Memory
FLIH	First Level Interrupt Handler
FMA	Floating point Multiply Add operation
FPR	Floating Point Register
FPU	Floating Point Unit
FRCA	Fast Response Cache Architecture
FRU	Field Replaceable Unit
FSRM	File System Resource Manager
FTP	File Transfer Protocol
FTP	File Transfer Protocol
GAI	Graphic Adapter Interface
GAMESS	General Atomic and Molecular Electronic Structure System
GPFS	General Parallel File System
GPR	General-Purpose Register
GUI	Graphical User Interface
GUID	Globally Unique Identifier
HACMP	High Availability Cluster Multi Processing
HACWS	High Availability Control Workstation
HCON	IBM AIX Host Connection Program/6000
HDX	Half Duplex
HFT	High Function Terminal
HIPPI	High Performance Parallel Interface
HiPS	High Performance Switch
HiPS LC-8	Low-Cost Eight-Port High Performance Switch
HMC	Hardware Management Console
HostRM	Host Resource Manager
HP	Hewlett-Packard
HPF	High Performance FORTRAN
HPSSDL	High Performance Supercomputer Systems Development Laboratory
HP-UX	Hewlett-Packard UNIX
HTML	Hyper-text Markup Language
HTTP	Hypertext Transfer Protocol
Hz	Hertz

I/O	Input/Output
I^2C	Inter Integrated-Circuit Communications
IAR	Instruction Address Register
IBM	International Business Machines
ICCCM	Inter-Client Communications Conventions Manual
ICE	Inter-Client Exchange
ICElib	Inter-Client Exchange library
ICMP	Internet Control Message Protocol
ID	Identification
IDE	Integrated Device Electronics
IDS	Intelligent Decision Server
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IHV	Independent Hardware Vendor
IIOP	Internet Inter-ORB Protocol
IJG	Independent JPEG Group
IKE	Internet Key Exchange
ILS	International Language Support
IM	Input Method
INRIA	Institut National de Recherche en Informatique et en Automatique
IP	Internetwork Protocol (OSI)
IPL	Initial Program Load
IPSec	IP Security
IrDA	Infrared Data Association (which sets standards for infrared support including protocols for data interchange)
IRQ	Interrupt Request
IS	Integrated Service
ISA	Industry Standard Architecture, Instruction Set Architecture
ISAKMP	Internet Security Association Management Protocol
ISB	Intermediate Switch Board
ISDN	Integrated-Services Digital Network
ISMP	InstallShield Multi-Platform
ISNO	Interface Specific Network Options
ISO	International Organization for Standardization
ISV	Independent Software Vendor
ITSO	International Technical Support Organization
JBOD	Just a Bunch of Disks
JDBC	Java Database Connectivity
JFC	Java Foundation Classes
JFS	Journaled File System
JTAG	Joint Test Action Group
KDC	Key Distribution Center
L1	Level 1
L2	Level 2
L2	Level 2
LAN	Local Area Network
LANE	Local Area Network Emulation
LAPI	Low-Level Application Programming Interface
LDAP	Lightweight Directory Access Protocol
LDIF	LDAP Directory Interchange Format
LED	Light Emitting Diode
LFD	Load Float Double
LFT	Low Function Terminal

LID	Load ID
LLNL	Lawrence Livermore National Laboratory
LP	Logical Partition
LP64	Long-Pointer 64
LPI	Lines Per Inch
LPP	Licensed Program Product
LPR/LPD	Line Printer/Line Printer Daemon
LRU	Least Recently Used
LTG	Logical Track Group
LV	Logical Volume
LVCB	Logical Volume Control Block
LVD	Low Voltage Differential
LVM	Logical Volume Manager
MAP	Maintenance Analysis Procedure
MASS	Mathematical Acceleration Subsystem
MAU	Multiple Access Unit
MBCS	Multi-Byte Character Support
Mbps	Megabits Per Second
MBps	Megabytes Per Second
MCA	Micro Channel Architecture
MCAD	Mechanical Computer-Aided Design
MDI	Media Dependent Interface
MES	Miscellaneous Equipment Specification
MFLOPS	Million of FLoating point Operations Per Second
MII	Media Independent Interface
MIP	Mixed-Integer Programming
MLR1	Multi-Channel Linear Recording 1
MMF	Multi-Mode Fibre
MODS	Memory Overlay Detection Subsystem
MP	Multiprocessor
MPC-3	Multimedia PC-3
MPI	Message Passing Interface
MPOA	Multiprotocol over ATM
MPP	Massively Parallel Processing
MPS	Mathematical Programming System
MST	Machine State
MTU	Maximum Transmission Unit
MWCC	Mirror Write Consistency Check
MX	Mezzanine Bus
NBC	Network Buffer Cache
NCP	Network Control Point
ND	Neighbor Discovery
NDP	Neighbor Discovery Protocol
NFB	No Frame Buffer
NFS	Network File System
NHRP	Next Hop Resolution Protocol
NIM	Network Installation Management
NIS	Network Information System
NL	National Language
NLS	National Language Support
NT-1	Network Terminator-1
NTF	No Trouble Found
NTP	Network Time Protocol
NUMA	Non-Uniform Memory Access
NUS	Numerical Aerodynamic Simulation
NVRAM	Non-Volatile Random Access Memory
NWP	Numerical Weather Prediction
OACK	Option Acknowledgment
OCS	Online Customer Support
ODBC	Open DataBase Connectivity
ODM	Object Data Manager

OEM	Original Equipment Manufacturer
OLAP	Online Analytical Processing
OLTP	Online Transaction Processing
ONC+	Open Network Computing
OOUI	Object-Oriented User Interface
OSF	Open Software Foundation, Inc.
OSL	Optimization Subroutine Library
OSLp	Parallel Optimization Subroutine Library
P2SC	POWER2 Single/Super Chip
PAM	Pluggable Authentication Mechanism
PAP	Privileged Access Password
PBLAS	Parallel Basic Linear Algebra Subprograms
PCI	Peripheral Component Interconnect
PDT	Paging Device Table
PDU	Power Distribution Unit
PE	Parallel Environment
PEDB	Parallel Environment Debugging
PEX	PHIGS Extension to X
PFS	Perfect Forward Security
PGID	Process Group ID
PHB	Processor Host Bridges
PHY	Physical Layer
PID	Process ID
PID	Process ID
PIOFS	Parallel Input Output File System
PKR	Protection Key Registers
PMTU	Path MTU
POE	Parallel Operating Environment
POP	Power-On Password
POSIX	Portable Operating Interface for Computing Environments
POST	Power-On Self-test
POWER	Performance Optimization with Enhanced Risc (Architecture)
PPC	PowerPC
PPM	Piecewise Parabolic Method
PPP	Point-to-Point Protocol
PREP	PowerPC Reference Platform
PSE	Portable Streams Environment
PSSP	Parallel System Support Program
PTF	Program Temporary Fix
PTPE	Performance Toolbox Parallel Extensions
PTX	Performance Toolbox
PV	Physical Volume
PVC	Permanent Virtual Circuit
PVID	Physical Volume Identifier
QMF	Query Management Facility
QoS	Quality of Service
QP	Quadratic Programming
RAID	Redundant Array of Independent Disks
RAM	Random Access Memory
RAN	Remote Asynchronous Node
RAS	Reliability, Availability, and Serviceability
RDB	Relational DataBase
RDBMS	Relational Database Management System
RDISC	ICMP Router Discovery
RDN	Relative Distinguished Name

RDP	Router Discovery Protocol
RFC	Request for Comments
RIO	Remote I/O
RIP	Routing Information Protocol
RIPL	Remote Initial Program Load
RISC	Reduced Instruction-Set Computer
RMC	Resource Monitoring and Control
ROLTP	Relative Online Transaction Processing
RPA	RS/6000 Platform Architecture
RPC	Remote Procedure Call
RPL	Remote Program Loader
RPM	Red Hat Package Manager
RSC	RISC Single Chip
RSCT	Reliable Scalable Cluster Technology
RSE	Register Stack Engine
RSVP	Resource Reservation Protocol
RTC	Real-Time Clock
RVSD	Recoverable Virtual Shared Disk
SA	Secure Association
SACK	Selective Acknowledgments
SAN	Storage Area Network
SAR	Solutions Assurance Review
SAS	Single Attach Station
SBCS	Single-Byte Character Support
ScaLAPACK	Scalable Linear Algebra Package
SCB	Segment Control Block
SCSI	Small Computer System Interface
SCSI-SE	SCSI-Single Ended
SDLC	Synchronous Data Link Control
SDR	System Data Repository
SDRAM	Synchronous Dynamic Random Access Memory
SE	Single Ended
SEPBU	Scalable Electrical Power Base Unit
SGI	Silicon Graphics Incorporated
SGID	Set Group ID
SHLAP	Shared Library Assistant Process
SID	Segment ID
SIT	Simple Internet Transition
SKIP	Simple Key Management for IP
SLB	Segment Lookaside Buffer
SLIH	Second Level Interrupt Handler
SLIP	Serial Line Internet Protocol
SLR1	Single-Channel Linear Recording 1
SM	Session Management
SMB	Server Message Block
SMIT	System Management Interface Tool
SMP	Symmetric Multiprocessor
SMS	System Management Services
SNG	Secured Network Gateway
SOI	Silicon-on-Insulator
SP	IBM RS/6000 Scalable POWER parallel Systems
SP	Service Processor
SPCN	System Power Control Network
SPEC	System Performance Evaluation Cooperative
SPI	Security Parameter Index

SPM	System Performance Measurement
SPOT	Shared Product Object Tree
SPS	SP Switch
SPS-8	Eight-Port SP Switch
SRC	System Resource Controller
SRN	Service Request Number
SSA	Serial Storage Architecture
SSC	System Support Controller
SSL	Secure Socket Layer
STFDU	Store Float Double with Update
STP	Shielded Twisted Pair
SUID	Set User ID
SUP	Software Update Protocol
SVC	Switch Virtual Circuit
SVC	Supervisor or System Call
SWVPD	Software Vital Product Data
SYNC	Synchronization
TCE	Translate Control Entry
Tcl	Tool Command Language
TCP/IP	Transmission Control Protocol/Internet Protocol
TCQ	Tagged Command Queuing
TGT	Ticket Granting Ticket
TLB	Translation Lookaside Buffer
TOS	Type Of Service
TPC	Transaction Processing Council
TPP	Toward Peak Performance
TSE	Text Search Engine
TSE	Text Search Engine
TTL	Time To Live
UCS	Universal Coded Character Set
UDB EEE	Universal Database and Enterprise Extended Edition
UDI	Uniform Device Interface
UIL	User Interface Language
ULS	Universal Language Support
UP	Uniprocessor
USB	Universal Serial Bus
USLA	User-Space Loader Assistant
UTF	UCS Transformation Format
UTM	Uniform Transfer Model
UTP	Unshielded Twisted Pair
UUCP	UNIX-to-UNIX Communication Protocol
VESA	Video Electronics Standards Association
VFB	Virtual Frame Buffer
VG	Volume Group
VGDA	Volume Group Descriptor Area
VGSA	Volume Group Status Area
VHDCI	Very High Density Cable Interconnect
VLAN	Virtual Local Area Network
VMM	Virtual Memory Manager
VP	Virtual Processor
VPD	Vital Product Data
VPN	Virtual Private Network
VSD	Virtual Shared Disk
VSM	Visual System Manager
VSS	Versatile Storage Server
VT	Visualization Tool
WAN	Wide Area Network
WLM	Workload Manager
WTE	Web Traffic Express
XCOFF	Extended Common Object File Format
XIE	X Image Extension
XIM	X Input Method
XKB	X Keyboard Extension

XLF	XL Fortran
XOM	X Output Method
XPM	X Pixmap
XSSO	Open Single Sign-on Service
XTF	Extended Distance Feature
XVFB	X Virtual Frame Buffer

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 276.

- *IBM @server Certification Study Guide - pSeries AIX System Administration*, SG24-6191
- *IBM @server Certification Study Guide - pSeries AIX System Support*, SG24-6199
- *IBM @server Certification Study Guide AIX 5L Installation and System Recovery*, SG24-6183
- *IBM @server Certification Study Guide AIX 5L Performance and System Tuning*, SG24-6184
- *IBM @server Certification Study Guide AIX 5L Problem Determination Tools and Techniques*, SG24-6185
- *IBM @server Certification Study Guide - pSeries HACMP for AIX*, SG24-6187
- *IBM Certification Study Guide RS/6000 SP*, SG24-5348
- *AIX 5L Performance Tools Handbook*, SG24-6039
- *Problem Solving and Troubleshooting in AIX 5L*, SG24-5496
- *AIX Logical Volume Manager, from A to Z: Introduction and Concepts*, SG24-5432
- *Managing AIX Server Farms*, SG24-6606
- *TCP/IP Tutorial and Technical Overview*, GG24-3376
- *AIX 5L Differences Guide Version 5.1 Edition*, SG24-5765

Other resources

These publications are also relevant as further information sources:

- *PCI Adapter Placement Reference,* SA38-0538
- *SSA Adapters: User's Guide and Maintenance Information*, SA33-3272
- *System Management Concepts: Operating System*, SC23-4311
- The following types of documentation are located at the following URL:

  http://www-1.ibm.com/servers/eserver/pseries/library

  - User guides
  - System management guides
  - Application programmer guides
  - All commands reference volumes
  - Files reference
  - Technical reference volumes used by application programmers

Referenced Web sites

These Web sites are also relevant as further information sources:

- http://www.redbooks.ibm.com
- http://www.opengroup.org/onlinepubs/9629799/toc.htm
- http://www.ibm.com/certify
- http://www-1.ibm.com/servers/eserver/pseries/library/hardware_docs/index.html
- http://www.storage.ibm.com
- http://techsupport.services.ibm.com/server/support?view=pSeries
- http://www-132.ibm.com/content/home/store_IBMPublicUSA/en_US/eServer/pSeries/pSeries.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

B

C

D

E

F

G

H

N

O

P

R

S

T

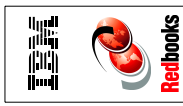
U

V

W

X

Y



IBM @server Certification Study Guide - AIX 5L

(0.5” spine)
0.475”<->0.875”
250 <-> 459 pages

IBM @server Certification Study Guide - AIX 5L Communications

Developed specifically for the purpose of preparing for AIX certification

Makes an excellent companion to classroom education

For experienced AIX professionals

This IBM Redbook is designed as a study guide for professionals wishing to prepare for the AIX 5L Communications certification exam as a selected course of study in order to achieve the IBM @server Certified Advanced Technical Expert - pSeries and AIX 5L certification.

This IBM Redbook is designed to provide a combination of theory and practical experience needed for a general understanding of the subject matter. It also provides sample questions that will help in the evaluation of personal progress and provide familiarity with the types of questions that will be encountered in the exam.

This publication does not replace practical experience, nor is it designed to be a stand alone guide for any subject. Instead, it is an effective tool that, when combined with education activities and experience, can be a very useful preparation guide for the exam. So, whether you are planning to take the AIX 5L Communications certification exam, or if you just want to validate your AIX skills, this redbook is for you.

This publication was updated to include the new content included in Test 236, which was based on AIX 5L Version 5.1.

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

SG24-6186-01 ISBN 0738427039