Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix F, “Special notices” on page 225.

First Edition (December 1999)

This edition applies to the IBM Transmission Control Protocol/Internet Protocol Feature for VM/ESA (TCP/IP Function Level 320), Program Number 5645-030.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HYJ Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

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

Figures ...................................................... ix
Tables ....................................................... xi

Preface ..................................................... xiii
The team that wrote this redbook ...................................... xiii
Comments welcome ........................................... xiv

Chapter 1. What is new in TCP/IP for VM/ESA ........................ 1
1.1 Summary of changes to TCP/IP Function Level 320 ............ 1
1.2 Summary of changes to TCP/IP Function Level 310 ............ 2
1.3 Enhancements to the TCP/IP capabilities of RSCS 3.2.0 .... 4

Chapter 2. TCP/IP overview ...................................... 7
2.1 The growth of TCP/IP . ..................................... 7
2.2 Internet standards and Request For Comments (RFC) ......... 8
2.3 TCP/IP architecture .......................................... 9
2.4 The TCP/IP network interface layer .......................... 10
  2.4.1 Network interface ....................................... 10
  2.4.2 Interfacing with the network layer ....................... 12
2.5 TCP/IP internetwork layer protocols .......................... 14
  2.5.1 Internet Protocol (IP) .................................. 14
  2.5.2 Internet Control Message Protocol (ICMP) ............... 21
2.6 TCP/IP connectivity bridging, switching and routing ........... 21
  2.6.1 Local area network ...................................... 22
  2.6.2 Wide area network ...................................... 24
  2.6.3 Bridging and switching .................................. 25
  2.6.4 Routing ................................................ 29
2.7 TCP/IP transport layer protocols and interfaces ............... 32
  2.7.1 Ports and sockets ....................................... 32
  2.7.2 The sockets application programming interface .......... 33
  2.7.3 User Datagram Protocol (UDP) .......................... 33
  2.7.4 Transmission Control Protocol (TCP) .................... 34
2.8 TCP/IP application protocols ................................ 35
  2.8.1 Remote login and terminal emulation: Telnet ............. 35
  2.8.2 File Transfer Protocols: FTP and TFTP ................. 36
  2.8.3 Remote printing: LPR and LPD .......................... 36
  2.8.4 Remote command execution: RSH and REXEC .......... ... 36
  2.8.5 Domain name system (DNS) ................................ 36
  2.8.6 Simple Mail Transfer Protocol (SMTP) .................... 39
  2.8.7 Multipurpose Internet mail extensions (MIME) .......... 39
  2.8.8 Post office protocol (POP) .............................. 39
  2.8.9 Internet message access protocol (IMAP) ............... 40
  2.8.10 Remote Procedure Call (RPC) ......................... 40
  2.8.11 Portmap ............................................... 40
  2.8.12 Network File System (NFS) ............................ 41
  2.8.13 X Window System ...................................... 41
2.9 TCP/IP configuration and management protocols ............... 41
  2.9.1 Bootstrap protocol (BOOTP) ............................... 41
  2.9.2 Dynamic Host Configuration Protocol (DHCP) ............ 41
  2.9.3 Simple Network Management Protocol (SNMP) .......... 42
Appendix E. Configuration files for server machines ............................. 216
  E.2.1 NSMAIN DATA ...................................................... 216
  E.2.2 VMNFS CONFIG .................................................. 217
  E.2.3 SMTP CONFIG file ................................................ 219
  E.2.4 UFTD CONFIG .................................................... 222
  E.2.5 ROUTED CONFIG .................................................. 223

Appendix F. Special notices .......................................................... 225

Appendix G. Related publications .................................................... 227
  G.1 IBM Redbooks publications ........................................... 227
  G.2 IBM Redbooks collections ............................................. 227
  G.3 Other resources ......................................................... 227
  G.4 Referenced Web sites .................................................... 228

How to get IBM Redbooks ............................................................ 231
IBM Redbook Fax Order Form .................................................... 232

List of abbreviations ........................................................................ 233

Index ......................................................................................... 239

IBM Redbooks evaluation .............................................................. 245
Tables

1. Internet growth .......................................................... 7
2. IP address ........................................................................ 14
3. IP - Class A address concept without subnets ....................... 16
4. IP Class A subnet mask .................................................. 17
5. IP Class A subnet host address ......................................... 17
6. IP Class A subnet number .............................................. 17
7. DNS: Some top-level Internet domains .............................. 38
8. Growth of the World Wide Web ....................................... 47
9. Sample OAT definitions .................................................. 103
10. Publications which may be required for enterprise printing ....... 148
11. LPDXMANY CONFIG tokens .......................................... 158
12. LPRXFORM CONFIG description .................................... 164
13. PCL prefix and suffix strings for HP890C .......................... 164
14. PostScript prefix and suffix strings for IBM Network Printer 24... 166
15. Requests for Comments .................................................. 203
Preface

TCP/IP is the most important protocol in the Internet networking environment. This redbook explains the basics of TCP/IP and also includes an overview of the latest developments in the world of TCP/IP and the Internet.

This redbook provides information that will ease the evaluation, installation and use of the VM/ESA Version 2 Release 4, TCP/IP feature FL 320. It focuses on new functions (NFS, FTP) and in addition covers special areas like routing, enterprise printing, IP mobility, address management, directory protocols and additional TCP/IP solutions like Web server, mail server and clients.

The team that wrote this redbook

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

Erich Amrehn is a certified Senior IT Specialist at the International Technical Support Organization, Poughkeepsie Center. Before joining the ITSO he worked as technical consultant to the IBM System/390 division for e-commerce on S/390 in Europe, the Middle East and Africa. He also has 13 years of VM experience in various technical positions in Germany and other areas in Europe and worldwide.

Jim Elliott is the VM and VSE Product Manager for IBM North America. Prior to this he provided technical marketing support for VM and VSE in Canada for 14 years. He has 26 years of experience in mainframe systems at IBM, most of that in the VM field, where he has worked in computing center operations, systems programming, sales and development. He has coauthored several redbooks on the subjects of VM and the IBM 4381 and IBM 9121 processors.

Kevin Hubbard has over twenty years experience with IBM, developing expertise in the design, marketing, installation, and support of IBM mainframe systems for customers in the manufacturing, distribution, retail, and education sectors.

Josef Minderlein is an Advisory IT Specialist in the S/390 Technical Support Group for VM/VSE in Germany. He has been employed by IBM for 30 years. He worked as a Customer Engineer on mainframe systems from 1969 to 1978, then as System Engineer in the VM field, and later as a Network Specialist on SNA. He has been a Project Leader for client/server projects in PSS since 1996. He has authored a redbook on the subject of IBM 3174 TCP/IP Implementation.

Mary Sue Record is an Advisory Software Engineer working with the VM TCP/IP development group. She has 19 years of experience in software development. She worked in VM development for 9 years and then worked on a variety of other operating systems such as TPF, ALCS and OS/390. She has recently returned to VM.

Heinz Ulbrich is a VM/ESA systems programmer at Suedwestrundfunk Stuttgart Germany. He has extensive experience in mainframe and network.

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

Your comments are important to us!

We want our redbooks to be as helpful as possible. Please send us your comments about this or other redbooks in one of the following ways:

- Fax the evaluation form found in “IBM Redbooks evaluation” on page 245 to the fax number shown on the form.
- Use the online evaluation form found at http://www.redbooks.ibm.com/
- Send your comments in an internet note to redbook@us.ibm.com
Chapter 1. What is new in TCP/IP for VM/ESA

This section highlights the changes in TCP/IP Function Level 310 (delivered as a feature of VM/ESA Version 2 Release 3) and in TCP/IP Function Level 320 (delivered as a feature of VM/ESA Version 2 Release 4). Changes in RSCS Version 3 Release 2 related to these TCP/IP function levels are also listed.

1.1 Summary of changes to TCP/IP Function Level 320

The following configuration enhancements were made:

- Dynamic storage pools will now dynamically grow to satisfy demand.
- New configuration parameters were added to manage the dynamic expansion.
- Monitoring information about new storage pool allocation will be generated.
- OBEYFILE command now allows changes to ASSORTEDPARMS and in INTERALCLIENTPARMS statements.
- New DOMAINLOOKUP and DOMAINSEARCH configuration statements were added to the TCPIP DATA file to support more flexible host name resolution processing.
- ATMARPSERVER, ATMLIS, and ATMPVC statements, and ATM-specific device and link statements are now supported for the TCP/IP virtual machine configuration.

Native ATM support was added, which enables TCP/IP to use an Open Systems Adapter configured to support native asynchronous transfer mode.

RouteD was updated to provide the following enhancements:

- RIP Version 2 Support - This is an extension of the Routing Information Protocol Version 1, and provides the following features:
  - Multicasting for RIP-2 Packets to Reduce load on Hosts
  - Authentication of RIP-2 Packets for Routing Update Security
  - Compatibility Switch
  For detailed information about these features, refer to Chapter 7 of TCP/IP Function Level 320 Planning and Customization, SC24-5847.
- Virtual IP addressing (VIPA), which provides:
  - Reduced traffic load on network interfaces
  - Automatic and transparent recovery from device and adapter failures
  - Recovery from TCP/IP stack failure
  For detailed information about VIPA refer to 2.5.1.4, “Virtual IP addressing (VIPA)” on page 18.

The following enhancements were made to the FTP Server:

- The VM Special Message Facility (SMSG) is available to query and modify FTP server operation without requiring a server restart. This means increased availability.
- Support was added for an FTP welcome banner, which displays a site-specific message when connecting to the FTP server or a user-specific message when a login occurs.
• A new startup parameter will enable FTP reader file support, allowing users to
direct files to the virtual reader of a VM user ID. CD and DELETE
subcommands were enhanced to support the VM virtual reader.

• FTP server exit routines may be provided to monitor and control FTP client
activity.

The following enhancements were made to the SMTP server:
• Support for the 8BITMIME and SIZE service extensions to improve throughput
and reduce complexity.
• User specified mail routing may be disallowed to improve security.

The following enhancements were made to the VMNFS server machine:
• NFS Version 3 (RFC 1813) support for improved performance and
interoperability.
• Improved security features.
• Personal Communication Network File System Daemon (PCNFSD) support for
Windows NFS clients.
• The SMSG command is used to provide an interface to communicate with the
Network File System service machine to enhance query and availability of the
server.

For TRACE and MORETRACE statements:
• A new process, named PACKET, allows tracing without requiring specification
of an additional process name.
• With MORETRACE it is now possible to generate very detailed trace output of
IP layer communication activity.

Performance improvements:
• Reduce time and overhead of processing TCP segment by checking to see if
next segment in sequence.
• Enhanced support for user-specified translation tables was added to the FTP,
LPR, LPRSET, and MOUNT commands. For more information, refer to TCP/IP
Function Level 320 Planning and Customization, SC24-5847

1.2 Summary of changes to TCP/IP Function Level 310

The following statements were added to, or changed in, TCP/IP for VM/ESA
Function Level 310:
• ATSIGN, SMTPSERVERID, and UFTSERVERID for defining TCP/IP system
parameters were added.
• VERIFYCLIENT, VERIFYCLIENTDELAY, and TRACE for the SMTP server
configuration file were added.
• DATABUFFERLIMITS, MONITORRECORDS, TN3270E, and TRACEONLY for
the TCP/IP server configuration file were added.
• FAILEDJOB for the LPD server configuration file was added.
• The PORT statement was extended to permit reservations to be made for
particular home addresses.
• The HOME statement's method of handling multiple IP addresses for the same link has changed.

Several new parameters were added to the INTERNALCLIENTPARMS statement to configure the Telnet server, an internal client of the TCP/IP virtual machine.

The DUMMYprobe and ACTIVEprobe VMCF interrupt header types were added to the VMCF TCP/IP Communication CALLCODE Notifications. These interrupt headers notify you that the TCP/IP virtual machine is monitoring your machine.

The following changes were made to the Network File System (VMNFS) virtual machine:

• The VMNFS CONFIG file allows you to configure your VMNFS server machine. You can define the values used for file extension translation, and whether or not Personal Communication Network File System Daemon is supported.

• Shared File System and Byte File System files and directories can be specified on the NFS MOUNT command.

• ANONYMOUS requests to mount BFS and SFS directories can be allowed or disallowed.

• More detailed tracing is available with the use of the MASK startup or SMSG command. Subnetting of trace information is also provided.

The SMSG ERROR command allows a CMS user to retrieve more detailed error information when using a BFS or SFS file.

The following sample configuration files were updated:

• Profile TCP/IP
• TCP/IP DATA
• SMTP CONFIG

The following virtual machines were added:

• Bootstrap Protocol Daemon (BOOTPD)
• Dynamic Host Configuration Protocol Daemon (DHCPD)
• Trivial File Transfer Protocol Daemon (TFTPD)
• Universal File Transfer Daemon (UFTD)

Sending electronic mail over TCP/IP networks is now supported by the CMS e-mail facilities, NOTE and SENDFILE.

Miscellaneous service APARs were added since the previous release to improve performance.

For Telnet server, the new IGNOREEAUDATA data option was added to INTERNALCLIENTPARMS statement. This causes the Telnet server to ignore any data associated with Erase All Unprotected (EAU) commands in the data stream received from the host.
1.3 Enhancements to the TCP/IP capabilities of RSCS 3.2.0

The following enhancements were made to RSCS Version 3 Release 2 related to TCP/IP:

- Addition of an Initial Time Out (ITO) PARM for LPR-type links.
- Ability to override the default orientation, font name, font size, and additional leading size used by the LPR- and TCPASCII-type sample postscript exits when printing plain text files.
- Ability to override the default filter used in the control file sent to a remote daemon by the LPR-type link sample exits.
- Ability of the LPR- LPD- and TCPASCII-type link sample exits to read an external translation table.
- Ability of the LPR- and TCPASCII-type link sample postscript exits to read an external postscript program used when printing plain text files.
- Ability of the LPR- and TCPASCII-type link sample postscript exits to read additional font names used when printing plain text files.
- Ability of the LPD-type link sample exit to read an external configuration file to provide overrides to defaults used by the exit on a printer queue name basis when receiving a file from a remote host:
  - Logical record length up to 1280 characters
  - Number of lines per page
  - Spool file class
  - Spool file form name
  - Job name
  - PSF destination
  - Whether or not to paginate regardless of control file setting
  - Whether or not to translate the data received
  - User ID file should be spooled to
  - Node ID file should be spooled to
  - TCPXLBIN translate table to use
- Ability to customize the control file created by the LPR-type link sample exits via parameters that can be defined in the sample exits configuration file.
- Domain Name Server support for LPR-, TCPNJE-, TCPASCII-, and UFT-type links.
- A TN3270E-Type link intended for use with TELNET printer sessions.
- Ability to override the host name portion of the control and data file names a LPR-type link constructs when sending the 'Receive control file' and 'Receive data file' subcommands to a remote daemon.
- Ability of the LPR- and TCPASCII-type link sample non-postscript printer exits to handle simple PCL strings imbedded at the start of, and SCS transparency orders contained anywhere within, each record of the file being processed for printing.
- Ability of the LPR- and TCPASCII-type link sample exits to split the prefix string before and after the separator page.
- Ability of the LPR- and TCPASCII-type link sample exits to produce a two-page separator useful when duplexing.
• Ability of users to provide keywords at the beginning of a file to contain overrides for a LPR-type link:
  - The fully-qualified host name
  - The dotted decimal host address
  - The one-character filter
  - The printer queue name
  - A prefix string
  - A suffix string
  - A translate table
  - The separator page setting

The PPS EXEC and PPS XEDIT macros were enhanced to support reading these keywords from an RSCS NAMES file. The command and macro syntax are provided later in this chapter.

• Ability of an RSCS administrator to build form block tables to provide keywords that contain overrides for an LPR-type link. Defaults that can be overridden include:
  - The fully-qualified host name
  - The dotted decimal host address
  - The one-character filter
  - The printer queue name
  - A prefix string
  - A suffix string
  - A translate table
  - The form name
  - The separator page setting

The spool file form name is used to select a form block table to provide the overrides used by the LPR-type link when printing. The form block table is built using a new exit routine, called LPRXFORM, which utilizes exit points 0 and 1.

• Ability to have an LPR-type link place a file on hold when it was unable to connect to a remote daemon. This capability can be used in conjunction with the RSCS events file to periodically take the file off hold and try connecting again.

• Ability to limit the size of files, based on number of records, printed by a LPR-type link.

• Support for the European currency character.

• Ability to define the number of days a LPR-type link will place an undeliverable file on hold before requeueing it to a notify link.

• Ability to define a timeout value for a LPR-type link which will cause the link to abort transmission, and place the file on hold, if a response is not received from the remote host.

• Ability to select the local port range a LPR-type link will use.

• Ability to supply TAG information when using the PPS EXEC and PPS XEDIT macro.

• Unsolicited File Transfer (UFT) Support.

• Support for POSIX translate tables in the LPR-type link sample exits.

• Support for a separate translate table used for translating the control file in the LPR- and LPD-type link sample exits.
Chapter 2. TCP/IP overview

The need to interconnect networks that are based on different protocols and platforms was recognized early in the 1970s, during a period when the use and development of networking technology was increasing. The rapid growth in networking over the past three decades has allowed users much greater access to resources and information, but it has caused significant problems when merging, or interconnecting, different types of networks. Open protocols and common applications were required; this led to the development of a protocol suite known as Transmission Control Protocol/Internet Protocol (TCP/IP) which originated with the U.S. Department of Defense (DoD) in the mid-1960s and took its current form around 1978. An interesting article about the history of the Internet can be found at the following URL:

http://www.isoc.org/internet-history/

2.1 The growth of TCP/IP

In the early 1980s TCP/IP became the backbone protocol in multivendor networks such as ARPANET, NFSNET and regional networks. The protocol suite was integrated into the University of California at Berkeley’s UNIX operating system and became available to the public for a nominal fee. The inexpensive availability of TCP/IP in UNIX, combined with its spread to other operating systems, resulted in its increasing use in both local area network (LAN) and wide area network (WAN) environments. Today, TCP/IP provides corporations with the ability to merge differing physical networks while giving users a common suite of functions. It allows interoperability between equipment supplied by multiple vendors on multiple platforms, and it provides access to the Internet.

In fact, the Internet, which has become the largest computer network in the world, is based on the TCP/IP protocol suite. The Internet consists of large international, national and regional backbone networks that allow local and campus networks and individuals access to global resources. Use of the Internet has grown rapidly over the last few years, as illustrated in Table 1. The most recent estimate suggests in excess of 29 million hosts on the Internet today.

Table 1. Internet growth

<table>
<thead>
<tr>
<th>Date</th>
<th>Hosts</th>
<th>Networks</th>
<th>Domains</th>
</tr>
</thead>
<tbody>
<tr>
<td>July 1989</td>
<td>130,000</td>
<td>650</td>
<td>3,900</td>
</tr>
<tr>
<td>July 1992</td>
<td>992,000</td>
<td>6,569</td>
<td>16,300</td>
</tr>
<tr>
<td>July 1993</td>
<td>1,776,000</td>
<td>13,767</td>
<td>26,000</td>
</tr>
<tr>
<td>July 1995</td>
<td>6,642,000</td>
<td>61,538</td>
<td>120,000</td>
</tr>
<tr>
<td>July 1996</td>
<td>12,881,000</td>
<td>134,365</td>
<td>488,000</td>
</tr>
<tr>
<td>July 1997</td>
<td>19,540,000</td>
<td>--</td>
<td>1,301,000</td>
</tr>
</tbody>
</table>

In contrast to the Internet, the term intranet has evolved recently to describe TCP/IP networks that are entirely under the control of a private authority or company. These intranets may or may not have connections to other independent intranets (which would then be referred to as extranets) or the Internet. They may
or may not be fully or partially visible to the outside, depending on the implementation.

TCP/IP also provides for the routing of multiple protocols to and from diverse networks. For example, a requirement to connect isolated networks using IPX, AppleTalk and TCP/IP protocols using a single physical connection can be accomplished by using routers utilizing TCP/IP protocols.

One further reason for the growth of TCP/IP is the popularity of the socket programming interface, which is the programming interface between the TCP/IP transport protocol layer and TCP/IP applications. A large number of applications today have been written for the TCP/IP socket interface.

### 2.2 Internet standards and Request For Comments (RFC)

We mentioned in the previous section that the Internet is a large multinational, multivendor, multiplatform network. This raises some questions, such as:

- Are there any standards for such a diverse network?
- Who establishes and reviews them?
- Who assigns network addresses?
- Who manages the Internet?

The Internet Society (ISOC), formerly known as Internet Activities Board (IAB), is the non-profit, coordinating committee for Internet design, engineering and management. The ISOC members are committed to making the Internet function effectively and evolve to meet a large-scale, high-speed future. The ISOC has established several bodies for administering, standardizing, and researching for the Internet:

- The Internet Architecture Board (IAB)
- The Internet Engineering Task Force (IETF)
- The Internet Research Task Force (IRTF)
- The Internet Assigned Numbers Authority (IANA)

While the IAB oversees and manages the Request For Comments (RFC) publication process, the IETF actually defines the standards through a number of subcommittees or task forces, and the IRTF engages in Internet-related research projects.

RFC is the mechanism through which the Internet protocol suite has been evolving. For example, an Internet protocol can have one of six states: standard, draft standard, proposed standard, experimental, informational and historic. In addition, an Internet protocol has one of five statuses: required, recommended, elective, limited use and not recommended. By communicating using the RFC, new protocols are being designed and implemented by researchers from both academic institutions and commercial corporations. At the same time, some old protocols are being superseded by new ones.

RFCs can be viewed or obtained online from the IETF Web page using the following URL:

http://www.ietf.org/rfc.html
The RFC standards are described in the “Internet Official Protocol Standards” RFC, currently RFC 2400. See Appendix C, “RFCs related to protocol specifications” on page 203.

The task of coordinating the assignment of values to the parameters of protocols is delegated to the IANA. These protocol parameters include op-codes, type fields, terminal types, system names, object identifiers, and so on. The “Assigned Numbers” RFC, currently RFC 1700, documents these protocol parameters.

To obtain registered IP addresses (see 2.5.1.1, “IP addressing” on page 14) and domain names (see 2.8.5, “Domain name system (DNS)” on page 36), you need to contact the Internet Network Information Center (InterNIC), the administrative body for the Internet.

Registration is available online at the NIC Web site using the following URL:

http://rs.internic.net/

2.3 TCP/IP architecture

TCP/IP, as a set of communications protocols, is based on layers. Unlike SNA or OSI, which distinguish seven layers of communication, there are only four layers in the TCP/IP model. They enable heterogeneous systems to communicate by performing network-related processing such as message routing, network control, error detection and correction.

The layering model of TCP/IP is shown in Figure 1, with an explanation of each layer following thereafter:

![TCP/IP architecture model: Layers and protocols](Figure 1)

**Application layer**
The application layer is provided by the program that uses TCP/IP for communication. Examples of applications are:

- Domain Name Server (DNS)
- File Transfer Product (FTP)
- Teletypewriter Network (Telnet)
- Remote Execution (Rexec)
- Line printer request (LPR).
The interface between the application and transport layers is defined by port numbers and sockets, which are described in more detail in 2.7.1, “Ports and sockets” on page 32.

**Transport layer**
The transport layer provides communication between application programs. The applications may be on the same host or on different hosts. Multiple applications can be supported simultaneously. The transport layer is responsible for providing a reliable exchange of information. The main transport layer protocol is TCP. Another is User Datagram Protocol (UDP), which provides a connectionless service, in contrast to TCP, which provides a connection-oriented service. That means that applications using UDP as the transport protocol have to provide their own end-to-end flow control. Usually, UDP is used by applications that need a fast transport mechanism.

**Internetwork layer**
The internetwork layer provides communication between computers. Part of communicating messages between computers is a routing function that ensures that messages will be correctly delivered to their destination. The Internet Protocol (IP) provides this routing function. Examples of internetwork layer protocols are IP, ICMP, ARP and RARP.

**Network interface layer**
The network interface layer, sometimes also referred to as link layer, data link layer or network layer, is implemented by the physical network that connects the computers. Examples are LAN (Token-Ring = IEEE 802.5 and Ethernet = IEEE 802.3 standards), Ethernet, X.25, ISDN, ATM, SNA, CTC, CLAW or ICA.

### 2.4 The TCP/IP network interface layer

Network protocols define how data is transported over a physical network. After a TCP/IP packet is created, the network protocol adds a transport-dependent network header before the packet is sent out on the network.

#### 2.4.1 Network interface

TCP/IP requires a network processor and associated components for attachment to the teleprocessing network. This section describes possible network attachments.

**IBM 9221 Integrated Communication Processor**
The IBM 9221 Information System can be configured for Token-Ring LANs, Ethernet, and X.25 subsystems, depending on the communication processor installed. The IBM 9221 Communications Processor requires one of the following LAN adapters:

- IBM 9221 802.3 Ethernet LAN
- IBM 9221 Token-Ring LAN (16/4 Mbps or 4 Mbps)
- IBM 9221 X.25 Communications Subsystem Controller

**IBM 2216 Nways FDDI Multiaccess Connector**
The IBM 2216 Nways Multiaccess Connector delivers wide area network access, S/390 host access and remote site concentration. Combined with the Nways Multiprotocol Access Service (MAS), the IBM 2216 helps to increase network performance and operations efficiencies.
The IBM 2216 Multiaccess Connector supports the following types of network interfaces:

- IBM Ethernet (10/100 Mbps)
- IBM Token-Ring (4/16 Mbps)
- IBM FDDI LAN (100 Mbps)
- ATM LAN Emulation Mode
- ATM native
- Multi-Path Channel (MPC+) support
- High Performance Data Transfer for TCP/IP and SNA

**IBM Open Systems Adapter 2 (OSA-2)**

An IBM System/390 (S/390) Open Systems Adapter is an integrated hardware feature that lets its S/390 host platform provide connectivity to clients on directly-attached local area networks or, via attachment to an asynchronous transfer mode (ATM) switch, to clients in an ATM-based network. An OSA-2 therefore positions its host platform to be an open systems platform by bringing S/390 resources directly to networks. Each OSA-2 feature supports:

- ENTR (Ethernet/Token-Ring) feature with two ports. Each port can be configured as Ethernet or Token-Ring. Each port supports attachment to a 10 Mbps Ethernet LAN and/or a 4 or 16 Mbps Token-Ring LAN.
- FDDI feature has one LAN port and supports attachment to a 100 Mbps FDDI LAN. One single ring or dual ring station is supported, as well as attachment to an optical bypass switch.
- 155 ATM multimode feature with one port and attachment to a 155 Mbps ATM network using multimode fiber optic cable.
- 155 ATM single mode feature with one port and attachment to a 155 Mbps ATM network using single mode fiber optic cable.
- Fast Ethernet feature with one port and attachment to either a 10 Mbps or 100 Mbps Ethernet LAN.

The OSA-2 features are supported by the Open Systems Adapter Support Facility (OSA/SF) in the VM/ESA (TCP/IP and SNA/APPN) environment. The ability to configure OSA-2 to allow sharing among logical partitions (LPAR support) is a key advantage offered by OSA/SF. When the S/390 server is running in LPAR mode, TCP/IP and SNA/APPN applications can share access to OSA-2 on the same LAN port.

**IBM 3745/46 Communications Controller Interface**

IBM 3745/3746 Communications Controller family has evolved to meet the demands for enhanced networks connectivity. Examples of these demands include connections to frame-relay networks, LANs, and Wide Area Networks. IBM 3745/3746 supports the following connectivities:

- IBM Token-Ring LANs of 4 Mbps and 16 Mbps
- Ethernet Version 2 LANs up to 10 Mbps
- X.25

**HYPERchannel A220 Processor Adapter**

TCP/IP for VM/ESA supports the HYPERchannel Series A and B devices. In addition, TCP/IP for VM/ESA supports HYPERchannel Series DX devices
provided they function as Series A and B devices. For more information, see the appropriate Networking Systems Corporation documentation.

**Common Link Access to Workstation (CLAW)**
The Block Multiplexer Channel Adapter connectivity allows the system unit (RISC SYSTEM/6000) to communicate with an S/390 host. The support of TCP/IP for VM/ESA is achieved via the high-performance CLAW protocol. This protocol improves the performance on the S/390 processor by reducing the number of I/O interrupts to the CLAW host. This connectivity allows the system unit to be used as a gateway between the VM/ESA host and the downstream networks that consist of LANs or WANs. AIX Version 3.2 or later of the operating system is required for the support.

The system unit supports at most two Multiplexer Channel Adapters, allowing the attachment of two channel interfaces. The two channel interfaces may be attached to the same host or to two different hosts.

The various configurations are:

- Normal mode connectivity between S/390 host and a system unit. The protocol is similar to CTC.
- Dual Attachment configurations: a system unit with two Block Multiplexer Channel Adapters may be connected to a single host or to two different hosts.
- Direct channel attachment between S/390 hosts and a system unit: the system unit connected to the channel is the machine used to communicate with the host. The user can be connected to the system unit by means of an ASCII terminal, an X-station on a local area network, or remotely logged in from a LAN.
- Direct channel attachment between S/390 hosts and a system unit using the 3044 Channel Extender: the only difference with the previously described configuration is that the 3044-Fiber Optic Channel Extender units allow the channel distance to extend to a maximum of 3 km.
- Attachment to ESCON using the 9034 converter: if a 9034 is installed, it requires a dedicated ESCON channel. No other ESCON control unit may be attached; however, other parallel channel control units may be added through the same 9034.

**Channel-to-Channel support (CTC)**
VM/ESA supports the IBM 3088 CTC for S/390 host interconnection.

The virtual CTC support in VM/ESA Control Program (CP) emulates the operation of the IBM 3088 Channel-to-Channel Adapter (CTCA). A Virtual-Channel-to-Channel device (VCTC) is normally used to connect two “virtual machines”.

The TCP/IP device driver CTC is used to transport IP packets.

### 2.4.2 Interfacing with the network layer

Though the network interface layer itself is not covered by the TCP/IP standards, the RFCs do specify certain methods to access that layer from the higher layers. Before we describe some of the protocols that interface with the network layer, we need to distinguish between different types of networks that the Internet layer can be connected with:
**Multiaccess broadcast networks**
In a network of this type, any system (TCP/IP host) can have multiple connections to other hosts simultaneously, and it can also send information to all other hosts on the same network with a single, special kind of message (broadcast). Local area networks typically represent this type of network. Protocols such as ARP, ProxyARP, RARP, BootIP and DHCP are used with this type of network. We will briefly describe some of them in this and following sections.

**Multiaccess non-broadcast networks**
In a network of this type, any host can have multiple connections to other hosts simultaneously, but there are no broadcast mechanisms in place.

Examples of this type of network are X.25, ATM, Frame Relay and AnyNet Sockets over SNA.

**Point-to-point networks**
In a network of this type, a host can only have one connection to one other host at any time, and there are no broadcast mechanisms in place. Examples of this type of network are SNAlink and CTC connections.

Notes:
- The term *connection* in the three paragraphs above applies to any single IP interface of a host in any of the network types mentioned. For instance, a host could have multiple point-to-point interfaces and thus more than one connection at a time, but still only one per interface.
- Some publications only distinguish between broadcast and non-broadcast networks.

### 2.4.2.1 Hardware Address Resolution (ARP and RARP)
The Address Resolution Protocol (ARP) maps Internet addresses to hardware addresses. When an application attempts to send data over a TCP/IP network capable of broadcasting, IP requests the appropriate hardware address mapping using ARP. If the mapping is not in the mapping table (ARP cache), an ARP broadcast packet is sent to all the hosts on the network requesting the physical hardware address for the host. For more information about ARP, see RFC 826.

An exception to the rule constitutes the Asynchronous Transfer Mode (ATM) technology where ARP cannot be implemented in the physical layer as described above. Therefore, an ARP server is used, with which every host has to register upon initialization in order to be able to resolve IP addresses to hardware addresses.

Some network hosts do not know their IP addresses when they are initialized. This especially can be true in the case of a host needing to be booted from diskette. Reverse ARP (RARP) can be used by, for example, a diskless workstation to determine its own IP address. In this case the workstation would already know its hardware address (discovered at initialization) and would broadcast a request to a RARP server to map the addresses. It is necessary to have a RARP server in your network in order to implement RARP.
2.5 TCP/IP internetwork layer protocols

This section provides a short overview of the most important and common protocols of the TCP/IP internetwork layer.

2.5.1 Internet Protocol (IP)

IP is the layer that hides the underlying physical network from the upper-layer protocols. It is an unreliable, best-effort and connectionless packet delivery protocol. Note that best-effort means that the packets sent by IP may be lost, out of order, or even duplicated, but IP will not handle these situations. It is up to the higher-layer protocols to deal with these situations.

One of the reasons for using a connectionless network protocol was to minimize the dependency on specific computing centers that used hierarchical connection-oriented networks. The Department of Defense (DoD) intended to deploy a network that would still be operational if parts of the country were destroyed. During earthquakes, this has been proved to be true for the Internet.

2.5.1.1 IP addressing

IP uses IP addresses to specify source and target hosts on the Internet. (For example, we can contrast an IP address in TCP/IP with a fully qualified NETID.LUNAME in SNA). An IP address consists of 32 bits, which is usually represented in the form of four decimal numbers, one decimal number for each byte (or octet). For example:

<table>
<thead>
<tr>
<th>00001001</th>
<th>01000011</th>
<th>00100110</th>
<th>00000001</th>
<th>A 32-bit IP address</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>67</td>
<td>38</td>
<td>1</td>
<td>Decimal notation (9.67.38.1)</td>
</tr>
</tbody>
</table>

An IP address consists of two logical parts: a network address and a host address. An IP address belongs to one of four classes depending on the value of its first four bits. (A fifth class, class E, is not commonly used.) This is shown in Figure 2 on page 15.
TCP/IP overview

Figure 2. IP - Assigned classes of IP addresses

- Class A addresses use 7 bits for the network and 24 bits for the host portion of the IP address. That allows for 126 \((2^{7}-2)\) networks with 16777214 \((2^{24}-2)\) hosts each; a total of over 2 billion addresses.
- Class B addresses use 14 bits for the network and 16 bits for the host portion of the IP address. That allows for 16382 \((2^{14})\) networks with 65534 \((2^{16}-2)\) hosts each; a total of over 1 billion addresses.
- Class C addresses use 21 bits for the network and 8 bits for the host portion of the IP address. That allows for 2097150 \((2^{21})\) networks with 254 \((2^{8}-2)\) hosts each; a total of over half a billion addresses.
- Class D addresses are reserved for multicasting (a sort of broadcasting, but in a limited area, and only to hosts using the same class D address).
- Class E addresses are reserved for future use.

Some values for these host IDs and network IDs are pre-assigned and cannot be used for actual network or host addressing:

**all bits 0**

Stands for this: this host (IP address with host address=0) or this network (IP address with network address=0). When a host wants to communicate over a network, but does not yet know the network IP address, it may send packets with network address=0. Other hosts on the network will interpret the address as meaning “this network”. Their reply will contain the fully qualified network address, which the sender will record for future use.

**all bits 1**

stands for all: all networks or all hosts. For example:

128.2.255.255

means all hosts on network 128.2 (class B address).
This is called a directed broadcast address because it contains both a valid network address and a broadcast host address.

**Loopback**

The class A network 127.0.0.0 is defined as the loopback network. Addresses from that network are assigned to interfaces which process data inside the local system and never access a physical network (loopback interfaces).

**2.5.1.2 IP subnets**

Due to the explosive growth of the Internet, the principle of assigned IP addresses became too inflexible to allow easy changes to local network configurations. Those changes might occur when:

- A new type of physical network is installed at a location.
- Growth of the number of hosts requires splitting the local network into two or more separate networks.
- Growing distances require splitting a network into smaller networks, with gateways between them.

To avoid having to request additional IP network addresses in these cases, the concept of subnets was introduced. The assignment of subnets can be done locally, as the whole network still appears to be one IP network to the outside world.

Recall that an IP address consists of a network address and a host address. For example, let us take a class A network; the address format is shown in Figure 3.

![Figure 3. IP - Class A address without subnets](image)

<table>
<thead>
<tr>
<th>01</th>
<th>8</th>
<th>16</th>
<th>24</th>
<th>31</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>netID</td>
<td></td>
<td>hostID</td>
<td></td>
</tr>
</tbody>
</table>

*Table 3. IP - Class A address concept without subnets*

Let us use the following IP address:

<table>
<thead>
<tr>
<th>00001001</th>
<th>01000111</th>
<th>00100110</th>
<th>00000001</th>
<th>A 32-bit IP address 9.67.38.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>67</td>
<td>38</td>
<td>1</td>
<td>Decimal notation (9.67.38.1) is an IP address (class A)</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td>The network address</td>
</tr>
<tr>
<td>67</td>
<td>38</td>
<td>1</td>
<td></td>
<td>The host address</td>
</tr>
</tbody>
</table>

Subnets form an extension to this by considering a part of the host address to be a subnetwork address. IP addresses are then interpreted as network address-subnetwork address-host address.

We may, for example, wish to choose the bits from 8 to 25 of a class A IP address to indicate the subnet addresses, and the bits from 26 to 31 to indicate the actual host addresses. Figure 4 on page 17 shows the format of a subnetted address that is thus derived from the original class A address.
We normally use a bit mask, known as the subnet mask, to identify which bits of the original host address field indicate the subnet number. In the above example, the subnet mask is 255.255.255.192 in decimal notation (or 11111111 11111111 11111111 11000000 in bit notation). Note that, by convention, the network address is part of the subnet mask as well.

For each of these subnet values, only \((2^{18})-2\) addresses (from 1 to 262142) are valid because of the all-bits-0 and all-bits-1 number restrictions. This split will therefore give 262142 subnets each with a maximum of \((2^6)-2\) or 62 hosts.

You will notice that the value applied to the subnet number takes the value of the full byte with non-significant bits being set to zero. For example, the hexadecimal value 11 in this subnet mask assumes an 8-bit value 11000000 and gives a subnet value of 192 and not 3 as it might seem.

Applying this mask to our sample class A address 9.67.38.1 would break the address down as follows:

<table>
<thead>
<tr>
<th>Table 4. IP Class A subnet mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>00001001</td>
</tr>
<tr>
<td>11111111</td>
</tr>
<tr>
<td>=======</td>
</tr>
<tr>
<td>00001001</td>
</tr>
</tbody>
</table>

and leaves a host address of:

<table>
<thead>
<tr>
<th>Table 5. IP Class A subnet host address</th>
</tr>
</thead>
<tbody>
<tr>
<td>=======</td>
</tr>
</tbody>
</table>

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 (see 4.1, “Routing terminology” on page 61).

Note that the actual subnet number would be:

<table>
<thead>
<tr>
<th>Table 6. IP Class A subnet number</th>
</tr>
</thead>
<tbody>
<tr>
<td>=======</td>
</tr>
</tbody>
</table>

You will notice that the subnet number shown above is a relative number, that is, it is the 68760th subnet of network 9 with the given subnet mask.

The division of the original host address part into subnet and host parts can be chosen freely by the local administrator; except that the values of all zeroes and all ones in the subnet field are reserved for special addresses.
Note: Because the range of available IP addresses is decreasing rapidly, many routers do support the use of all zeroes and all ones in the subnet field, though this is not consistent with the standards.

2.5.1.3 IP Datagram
The unit of transfer of a data packet in TCP/IP is called an IP datagram. It is made up of a header, containing information for IP and data that is only relevant to the higher level protocols. IP can handle fragmentation and re-assembly of IP datagrams. The maximum length of an IP datagram is 65,535 bytes (or octets). There is also a requirement for all TCP/IP hosts to support IP datagrams up to a size of 576 bytes without fragmentation.

The IP datagram header is a minimum of 20 bytes long, and is formatted as shown in Figure 5.

Figure 5. IP - Format of an IP datagram header

We do not elaborate on the format of the IP datagram header. You can find this information in RFC 791, Internet Protocol. For information on accessing this document, see Appendix C, “RFCs related to protocol specifications” on page 203.

2.5.1.4 Virtual IP addressing (VIPA)
Virtual IP Addressing (VIPA) frees other hosts from depending on a particular physical network interface for communication with a TCP/IP stack. Without VIPA, other hosts are bound to one of the host's home IP addresses and, therefore, to a particular physical network interface (for example, a device or adapter). If that interface fails, the associated connections are terminated. VIPA provides an IP address that is associated with a TCP/IP stack but not with a specific physical network interface. This allows hosts that connect to the TCP/IP stack to send data on whatever paths are selected by the routing protocols; thus, VIPA provides tolerance of physical network interface hardware failures.
To achieve network interface independence, VIPA relies on a virtual device and a virtual IP address. The virtual device is always active and never experiences a failure. A virtual IP address is the home address for a virtual device, which has no associated physical network interface. Inbound packets whose destination is a virtual IP address can be routed through any of the real physical network interfaces used by a TCP/IP stack. Failure of one physical network interface is handled by routing inbound traffic to another active physical network interface. Similarly, outbound packets can be routed around physical network interface outages, assuming an additional physical network interface provides an alternate path to the final destination. Static or dynamic routing protocols may be used to manage alternate paths (see 2.10.3, “Virtual Router Redundancy Protocol (VRRP)” on page 44). In general, VM/ESA provides the following functions:

- **Automatic and transparent recovery from device and adapter failure.**
  
  When a device or a physical adapter (such as a Token Ring or FDDI) fails, and there is another device that provides alternate paths to the destination, and other hosts make connections using the virtual IP addresses, IP:
  
  - detects the failure
  - finds an alternate path for each network
  - routes outbound traffic to hosts and routers on networks via alternate paths
  
  The result is fault tolerance for both inbound and outbound traffic, without the need to reestablish active connections that were using the failed device.

- **Recovery from TCP/IP stack failure (when an alternate TCP/IP stack is configured to provide the necessary redundancy).**

  Assume that an alternate TCP/IP stack is installed to serve as a backup and VIPA is configured on the primary TCP/IP stack. In case of a primary stack failure, the backup can be reconfigured to use the primary's virtual IP addresses. Client/server connections on the failed primary stack are disrupted but can be reestablished on the backup using the primary’s virtual IP addresses as destinations. In addition, the temporarily reassigned virtual IP addresses can be restored to the original primary stack once recovery is complete.

Figure 6 on page 20 shows an example of a VIPA configuration.
This sample configuration shows multiple network interfaces using a single virtual IP address. Depending upon the routing protocols used, the network or host routes in Router1 and Router2 are used to reach the destination virtual IP address (9.1.1.1).

**Configuring VIPA**

Assume that you want to configure TCP/IP to use a virtual IP address. The necessary steps in our example are:

- Add a virtual device and link to the DEVICE and LINK statements:
  ```
  DEVICE VDEF1 VIRTUAL 0
  LINK VIPA VIRTUAL 0 VDEF1
  ```

- Add the virtual link to the HOME statements:
  ```
  HOME
  VIPA 9.1.1.1
  TR1 9.2.1.1
  TR2 9.2.1.2
  ETH1 9.3.1.1
  ETH2 9.3.1.2
  ```

*Figure 6. Single VIPA configuration*
There is no virtual route to define; therefore, a virtual link cannot be defined by a GATEWAY statement.

Specify the SOURCEVIPA option in the ASSORTEDPARMS statement to activate device and adapter failure for requests or connections, originated at the VM/ESA TCP/IP stack.

If the virtual IP address is not the default local host address, use a PRIMARYINTERFACE statement.

Example:

```
PRIMARYINTERFACE TR1
```

For more detailed information about configuring virtual IP addresses, see *TCP/IP Function Level 320 Planning and Customization*, SC24-5847.

### 2.5.2 Internet Control Message Protocol (ICMP)

ICMP is shown in Figure 1 on page 9 as being in the same protocol layer as IP. It is actually an integral part of IP. ICMP is used for reporting errors in datagram delivery, such as “destination unreachable”, and it can assist in discovering routers and maximum transmission units (MTU) along a path that an IP datagram eventually travels.

#### 2.5.2.1 Packet Internet Groper (PING)

Perhaps one of the most useful commands available on all TCP/IP implementations is the Ping application. Ping uses ICMP to send an Echo datagram to a specified IP address and wait for it to return.

#### 2.5.2.2 Traceroute

The Traceroute program can be useful for debugging purposes. Traceroute enables determination of the route that an IP datagram follows from host to host. Traceroute is based upon ICMP and UDP. It sends an IP datagram with a TTL of 1 to the destination host. The first router to see the datagram will decrement the TTL to 0 and return an ICMP Time Exceeded message as well as discarding the datagram. In this way, the first router in the path is identified. This process can be repeated with successively larger TTL values in order to identify the series of routers in the path to the destination host. Traceroute actually sends UDP datagrams to the destination host which reference a port number that is outside the normally used range. This enables Traceroute to determine when the destination host has been reached, that is, when an ICMP Port Unreachable message is received. Traceroute is implemented in all IBM TCP/IP products. This is very useful for debugging purposes and also for learning if a remote host can be reached from the local host.

ICMP is defined in RFC 792.

### 2.6 TCP/IP connectivity bridging, switching and routing

An internetwork is a collection of individual networks, connected by networking devices, that functions as a single large network. Figure 7 on page 22 illustrates some different kinds of network technologies that can be interconnected by routers and other networking devices.
2.6.1 Local area network

A LAN is a high-speed data network. It connects workstations, personal computers, printers, and other devices. LANs offer many advantages, including shared access to devices and applications, file exchange between connected users, and communication via electronic mail and other applications. LAN protocols function at the lowest two layers of the TCP/IP model between the physical layer and the data link layer, as discussed in 2.3, “TCP/IP architecture” on page 9.

2.6.1.1 LAN media-access methods

LAN protocols use a couple of methods to access the physical network medium: Carrier Sense Multiple Access Collision Detection (CSMA/CD) and token passing. An example of the CSMA/CD scheme is ETHERNET/IEEE 802.3; examples of the token passing scheme are Token Ring/IEEE 802.5 and FDDI. See also Figure 8 on page 23 for illustration.
LAN topologies define the manner in which network devices are organized. Four common LAN topologies exist: bus, ring, star, and tree.

A bus topology is a linear LAN architecture in which transmissions from network stations propagate the length of the medium and are received by all other stations. The most common LAN implementations are Ethernet/IEEE 802.3 networks, a bus topology.

A ring topology is a LAN architecture that consists of devices connected by unidirectional transmission links to form a single closed loop. Both Token Ring/IEEE 802.5 and FDDI networks implement a ring topology.

A star topology is a LAN architecture in which the endpoints on a network are connected to a common central hub, or switch, by dedicated links. Logical bus and ring topologies are often implemented physically in a star topology.

A tree topology is a LAN architecture that is identical to the bus topology, except that branches with multiple nodes are possible in this case.

LAN devices

Devices commonly used in LANs are repeaters, hubs, LAN extenders, bridges, LAN switches, and routers.

A repeater is a physical layer device used to interconnect an extended network. Repeaters receive signals from one network and retransmit those signals to another network. These actions prevent signal deterioration caused by long cable lengths and large numbers of connected devices. Repeaters are unable to
perform complex filtering and other processing. In addition, all signals, including errors, are repeated and amplified.

A *hub* is a physical-layer device that connects multiple user stations via a dedicated cable. Hubs are used to create a physical star network while maintaining the logical bus or ring configuration of the LAN. In some respects, a hub functions as a multiport repeater.

A LAN *extender* is a remote-access multilayer switch that connects to a host router. LAN extenders forward traffic from all the standard network-layer protocols (such as IP, IPX, and AppleTalk), and filter traffic based on the MAC address or network-layer protocol type. LAN extenders filters out unwanted broadcasts and multicasts. LAN extenders are unable to segment traffic or create security firewalls. (See 2.6.3, “Bridging and switching” on page 25 and 2.6.4, “Routing” on page 29 for more discussion about bridges, switches and routers.)

### 2.6.2 Wide area network

A wide area network is a data communications network that covers a broad geographic area and often uses transmission facilities provided by common carriers, such as telephone companies. WAN technologies function at the lower three layers of the TCP/IP model: the physical layer, the data link layer, and the network layer.

#### 2.6.2.1 Point-to-point links

A point-to-point link provides a single WAN communications path from a specific customer site, through a carrier network such as a telephone company, to a remote network. A point-to-point link is also known as a leased line because its path is permanent and fixed for each remote network reached through the carrier facilities. The carrier company reserves point-to-point links for the private use of the customer. CTC is an example of a point-to-point link.

Figure 9 shows a typical point-to-point link operating through a WAN to a remote network.

![Figure 9. WAN point-to-point link](image)

#### 2.6.2.2 Circuit switching: ISDN

Circuit switching is a WAN switching method in which a dedicated physical circuit is established, maintained, and terminated through a carrier network for each communication session. Circuit switching operates much like a normal telephone call. Integrated Services Digital Network (ISDN) is an example of a circuit-switched WAN technology.
2.6.2.3 Packet switching
Packet switching is a WAN switching method in which network devices share a single point-to-point link to transport packets from a source to a destination across a carrier network. Asynchronous Transfer Mode (ATM), Frame Relay, and X.25 are examples of packet-switched WAN technologies.

2.6.2.4 WAN devices
Wide-area networks use devices such as switches and access servers.

A WAN switch is a multiport internetworking device used in carrier networks. These devices typically switch such traffic as Frame Relay and X.25, and they operate at the data link layer of the TCP/IP architecture model.

An access server acts as a concentration point for dial-in and dial-out connections.

2.6.3 Bridging and switching
Bridges and switches are data communications devices that operate at Layer 2 of the TCP/IP architecture model. They are referred to as data link layer devices.

Bridges became commercially available in the early 1980s. At the time of their introduction, bridges connected and enabled packet forwarding between networks. More recently, bridging between different networks has been defined and standardized. Several kinds of bridging have proven important as internetworking devices.

Transparent bridging is found primarily in Ethernet environments.

Source-route bridging occurs primarily in Token Ring environments.

Translational bridging provides translation between the formats and transit principles of different media types (usually Ethernet and Token Ring).

Source-route transparent bridging combines the algorithms of transparent bridging and source-route bridging to enable communication in mixed Ethernet/Token Ring environments.

Bridging and switching occur at the link layer, which controls data flow, handles transmission errors, provides physical addressing, and manages access to the physical medium. Bridges provide these functions by using various link-layer protocols with specific flow control, error handling, addressing, and media-access algorithms. Examples of popular link-layer protocols include Ethernet, Token Ring, and FDDI.

Bridges and switches analyze incoming frames, make forwarding decisions based on information contained in the frames, and forward the frames toward the destination. In some cases, such as source-route bridging, the entire path to the destination is contained in each frame. In other cases, such as transparent bridging, frames are forwarded one hop at a time toward the destination.

Upper-layer protocol transparency is a primary advantage of both bridging and switching. Because both device types operate at the link layer, they are not required to examine upper-layer information. This means that they can rapidly forward traffic representing any network-layer protocol. It is common for a bridge
to move AppleTalk, DECnet, IP, XNS, Netbios, and other traffic between two or more networks.

Bridges are capable of filtering frames based on any Layer 2 fields. Because link-layer information often includes a reference to an upper-layer protocol, bridges usually can filter on this parameter. Filters can be helpful in dealing with broadcast and multicast packets.

By dividing large networks into smaller units, called Subnetworks, bridges and switches provide several advantages. Because a certain percentage of traffic is forwarded, a bridge or switch will act as a firewall for some potentially damaging network errors, and can accommodate communication between a larger number of devices than would be supported on any single LAN connected to the bridge. Bridges and switches extend the effective length of a LAN.

One of the differences is that Switches are significantly faster because they switch in hardware, while bridges switch in software. Switches also support higher port densities than bridges. Some switches support cut-through switching, which reduces latency and delays in the network, while bridges support only store-and-forward. Finally, switches reduce collisions on network segments because they provide dedicated bandwidth to each network segment.

2.6.3.1 Types of bridges

Bridges can be grouped into categories based on various characteristics. Using one classification scheme, bridges are either local or remote. Local bridges provide a direct connection between multiple LAN segments in the same area. Remote bridges connect multiple LAN segments in different areas, usually over telecommunications lines. Figure 10 illustrates these two configurations.

Remote bridges cannot improve WAN speeds, but they compensate for speed discrepancies with a buffering capability. If a LAN device capable of a 3-Mbps transmission rate wants to communicate with a device on a remote LAN connected by a 64-Kbps link, the local bridge must regulate the 3-Mbps data stream so that it does not overwhelm the 64-Kbps serial link. This is done by storing the incoming data in on-board buffers and sending it over the serial link at a rate that the serial link can accommodate.
The Institute of Electrical and Electronic Engineers (IEEE) differentiates the OSI link layer into two separate sublayers: the Media Access Control (MAC) sublayer and the Logical Link Control (LLC) sublayer. The MAC sublayer permits media access, such as contention and token passing, while the LLC sublayer deals with framing, flow control, error control, and MAC-sublayer addressing.

Some bridges are MAC-layer bridges, which bridge between homogeneous networks (for example, IEEE 802.3 and IEEE 802.3), while other bridges can translate between different link-layer protocols (for example, IEEE 802.3 and IEEE 802.5).

### 2.6.3.2 Types of switches

Switches are data link layer devices that, like bridges, enable multiple physical LAN segments to be interconnected into a single larger network. Similar to bridges, switches forward traffic based on MAC addresses. Because switching is performed in hardware instead of in software, it is significantly faster. Switches use either store-and-forward switching or cut-through switching when forwarding traffic. Many types of switches exist, including ATM switches, LAN switches, and various types of WAN switches.

#### 2.6.3.3 ATM switch

Asynchronous transfer mode (ATM) switches provide high-speed switching for local and wide area applications. ATM switches support voice, video, and data applications and are designed to switch fixed-size information units called cells, which are used in ATM communications. Figure 11 illustrates an enterprise network comprised of multiple LANs interconnected across an ATM backbone.

![Figure 11. ATM switch with multiple LAN connections](image-url)
2.6.3.4 LAN switch
LAN switches are used to interconnect multiple LAN networks. LAN switching provides dedicated, collision-free communication between network devices, with support for multiple simultaneous conversations. LAN switches are designed to switch data frames at high speeds. Figure 12 illustrates a simple network in which a LAN switch interconnects a 10-Mbps and a 100-Mbps Ethernet LAN.

![LAN switch Ethernet](image)

**Switching**
Switching algorithms are relatively simple and are basically the same for most routing protocols. In most cases, a host determines that it must send a packet to another host. Having acquired a router's address, the source host sends a packet addressed specifically to a router's physical (Media Access Control, or MAC-layer) address, with the protocol (network layer) address of the destination host.

The router determines how to forward the packet to the next hop. If the router does not know how to forward the packet, it typically drops the packet. If the router knows how to forward the packet, it changes the destination physical address to that of the next hop and transmits the packet.

The next hop may be the ultimate destination host. If not, the next hop is usually another router, which executes the same switching decision process. As the packet moves through the internetwork, its physical address changes, but its protocol address remains constant, as illustrated in Figure 13 on page 30.

The preceding discussion describes switching between a source and a destination system. The International Organization for Standardization (ISO) has developed a hierarchical terminology that describes this process.
2.6.4 Routing

Routing is the act of moving information across an internetwork from a source to a destination. Along the way, at least one intermediate node typically is encountered. Routing is often contrasted with bridging, which might seem to be the same thing to the casual observer. The primary difference between the two is that bridging occurs at Layer 2 (the link layer), and routing occurs at Layer 3 (the network layer). Routing and bridging use different protocols in the process of moving data from source to destination.

Routing involves two basic activities: determining optimal routing paths and transporting information groups (typically called packets) through connected subnetworks.

2.6.4.1 Path determination

To aid the process of path determination, routers maintain information about the topology of the network.

They accomplish this by sending each other their routing tables through the transmission of messages. The routing update message is one such message that generally consists of all or a portion of a routing table. By analyzing routing updates from all other routers, a router can build a detailed picture of network topology. A link-state advertisement, another example of a message sent between routers, informs other routers of the state of the sender's links. Link information also can be used to build a complete picture of topology to enable routers to determine optimal routes to network destinations.
Figure 13 illustrates a system in which numerous routers come into play during the switching process.

2.6.4.2 Routing algorithms
The following sections analyze some routing algorithm types with different key characteristics:

Static and dynamic routing
Static routing requires that routes be configured manually by the network administrator. Because static routing systems cannot react to network changes, they are considered unsuitable for today's large, changing networks. Most of the dominant routing algorithms in the 1990s are dynamic routing algorithms, which adjust to changing network circumstances by analyzing incoming routing update messages. If the messages indicate that a network change has occurred, the
routing software recalculates routes and sends out new routing update messages. These messages change the routing tables accordingly.

**Single-path and multipath routing**

Some routing protocols support multiple paths to the same destination. Unlike single-path algorithms, these multipath algorithms permit traffic multiplexing over multiple lines. The advantages of multipath algorithms are obvious: they can provide better throughput and reliability.

### 2.6.4.3 Routing metrics

Routing tables contain information used by switching software to select the best route. But how are routing tables built? What is the information they contain? How do routing algorithms determine that one route is preferable to others?

Routing algorithms use many different metrics to determine the best route. Sophisticated routing algorithms can base route selection on multiple metrics, combining them in a single (hybrid) metric. All the following metrics can be used:

- **Path length**
  
  Path length is the most common routing metric. Some routing protocols allow network administrators to assign arbitrary costs to each network link. In this case, path length is the sum of the costs associated with each link traversed. Other routing protocols define hop count, a metric that specifies the number of passes through internetworking products, such as routers, that a packet must take en route from a source to a destination.

- **Reliability**
  
  Reliability, in the context of routing algorithms, refers to the dependability (usually described in terms of the bit-error rate) of each network link. Some network links might go down more often than others. After a network fails, certain network links might be repaired more easily or more quickly than other links. Any reliability factors can be taken into account in the assignment of the reliability ratings, which are arbitrary numeric values usually assigned to network links by network administrators.

- **Delay**
  
  Routing delay refers to the length of time required to move a packet from source to destination through the internetwork. Delay depends on many factors, including the bandwidth of intermediate network links, the port queues at each router along the way, network congestion on all intermediate network links, and the physical distance to be travelled. Because delay is a conglomeration of several important variables, it is a common and useful metric.

- **Bandwidth**
  
  Bandwidth refers to the available traffic capacity of a link. All other things being equal, a 10-Mbps Ethernet link would be preferable to a 64-kbps leased line. Although bandwidth is a rating of the maximum attainable throughput on a link, routes through links with greater bandwidth do not necessarily provide better routes than routes through slower links. If, for example, a faster link is busier, the actual time required to send a packet to the destination could be greater.
• Load

Load refers to the degree to which a network resource, such as a router, is busy. Load can be calculated in a variety of ways, including CPU utilization and packets processed per second. Monitoring these parameters on a continual basis can itself be a resource-intensive activity.

• Communication cost

Communication cost is another important metric, especially because some companies may not care about performance as much as they care about operating expenditures. Even though line delay may be longer, they will send packets over their own lines rather than through the public lines that cost money for usage time.

2.6.4.4 Network protocols

Routed protocols are transported by routers across an internetwork. In general, routed protocols in this context also are referred to as network protocols. These network protocols perform a variety of functions required for communication between user applications in source and destination devices, and these functions can differ widely among protocol suites. Network protocols occur at the upper four layers of the OSI reference model: the transport layer, the session layer, the presentation layer, and the application layer.

2.7 TCP/IP transport layer protocols and interfaces

This section provides a brief overview of the protocols of the TCP/IP transport layer.

2.7.1 Ports and sockets

Each process that wants to communicate with another process identifies itself to the TCP/IP protocol suite by one or more ports. A port is a 16-bit number used by the host-to-host protocol to identify to which higher-level protocol or application program (process) it must deliver incoming messages.

As some higher-level programs are themselves protocols, standardized in the TCP/IP protocol suite, such as Telnet and FTP, they use the same port number in all TCP/IP implementations. (Port 23 is used by a Telnet server; ports 20 and 21 are used by an FTP server.) Those assigned port numbers are called well-known ports and the standard applications are called well-known services.

The well-known ports are controlled and assigned by the IANA. On most systems these ports can only be used by system processes or by programs executed by privileged users. The assigned well-known ports occupy port numbers in the range 0 to 1023. The ports with numbers in the range 1024-65535 are not controlled by the Internet central authority and on most systems can be used by ordinary user-developed programs.

Confusion caused by two different applications trying to use the same port number on one host is avoided by writing those applications to request an available port from TCP/IP. Because this port number is dynamically assigned, it may differ from one invocation of an application to the next.
UDP, TCP and ISO TP-4 all use the same port principle. To the extent possible, the same port numbers are used for the same services on top of UDP, TCP and ISO TP-4.

A socket is a special type of file handle that is used by a process to request network services from the operating system. A socket address (also known as a transport address) is made up of three parts: {protocol, local address, local port number} or {protocol, foreign address, foreign port number}. An example of a socket in the TCP/IP suite is {TCP, 193.44.234.3, 12345}.

In other words, a socket is an end point for communication that can be named and addressed in a network.

### 2.7.2 The sockets application programming interface

The socket interface is one of several application programming interfaces (APIs) to the communication protocols. Designed to be a generic communication programming interface, it was first introduced by the 4.2BSD UNIX system. Although it has not been standardized, it has become a de facto industry standard.

The socket interface is differentiated by the services that are provided to applications: stream sockets (connection-oriented), datagram sockets (connection-less), and raw sockets (direct access to lower-layer protocols) services.

A variation of the BSD sockets interface is provided by the Winsock interface developed by Microsoft and other vendors to support TCP/IP applications on Windows operating systems. Winsock 2.0, the latest version, provides a more generalized interface allowing applications to communicate with any available transport layer protocol and underlying network services, including, but no longer limited to, TCP/IP.

### 2.7.3 User Datagram Protocol (UDP)

User Datagram Protocol is basically an application interface to IP. It provides no additional reliability, flow-control or error recovery. It simply serves as a multiplexer/demultiplexer for sending/receiving IP datagrams, using ports to direct the datagram. As a result of this, the maximum amount of data that can be transferred by UDP at any time is what fits within a single IP datagram, less the size of the UDP header. This typically makes UDP a good choice for messaging and querying applications that do not rely on immediate acknowledgements, such as SMTP, DNS and SNMP. See Figure 14 on page 34 for a graphic representation or this interface.
2.7.4 Transmission Control Protocol (TCP)

We have previously discussed IP, the unreliable connectionless packet (datagram) delivery mechanism that forms the basis of the TCP/IP protocol suite. Transmission Control Protocol, or TCP, is the higher-level protocol that provides reliability, flow control and some error recovery. Many of the TCP/IP application protocols, such as Telnet and FTP, use TCP as the underlying protocol.

TCP is a connection-oriented, end-to-end reliable protocol providing logical connections between pairs of processes. Within TCP, a connection is uniquely defined by a pair of sockets (that is, by a pair of processes, on the same or different systems, that are exchanging information).

The two processes communicate with each other over a TCP connection (Inter Process Communication - IPC), as shown in Figure 15.

In this example, processes 1 and 2 communicate over a TCP connection carried by IP datagram.
TCP does not recognize any application’s data patterns but treats data as a stream of bytes that the sending application writes to a buffer known as the TCP window. The following steps explain TCP operation in a simplified way:

- During connection establishment, the two hosts agree on an initial window size for that connection. TCP on the sending system then fills up the window with application data, chops its content into packets that conveniently fit into IP datagrams and passes those on to IP to be submitted to the receiver.
- TCP on the receiving system puts the packets back in order, sends an acknowledgment for every two packets received in order and fills up a receive buffer (window) from which the receiving application retrieves the incoming data stream.
- TCP on the sending system advances the window for as many packets as have been acknowledged (in order) and continues to transmit packages until the end of the data stream is reached. If packets are not acknowledged, TCP will wait and retransmit them before notifying the application of a communications problem. While TCP awaits acknowledgements, the window cannot be advanced and packet transfer will slow down.
- TCP provides flow control during an ongoing communication by varying send and receive window sizes at each transfer as required, which is important if a system runs out of memory for buffer space, or when a very fast system is communicating with a very slow system.
- TCP also provides for urgent data (interrupt signalling) to travel ahead of communication traffic on the current connection.
- When the sending application terminates, normally or abnormally, TCP will try to gracefully shut down the connection.

2.8 TCP/IP application protocols

One of the reasons why TCP/IP is so popular is that there are many simple and useful standard applications available. We summarize several common TCP/IP applications in this section.

2.8.1 Remote login and terminal emulation: Telnet

Teletypewriter network, or Telnet, is the virtual terminal protocol in TCP/IP. It allows users of one host to log into a remote host and interact as normal terminal users of that host. Telnet is a line-oriented protocol using the ASCII character set. Though initially designed for terminal emulation, Telnet is also used as the underlying protocol for file transfer (FTP control sessions) and e-mail (SMTP) operations.

For readers who are familiar with SNA, we can relate Telnet in TCP/IP to the terminal emulators (3270 or 5250 types) in SNA. In fact, all of the IBM TCP/IP product implementations provide Telnet support of 3270 terminal emulation in addition to the many other terminal emulation protocols, such as the widely used DEC VT terminal emulation types.

3270 terminal emulation differs from normal Telnet operation in the following ways:
- It uses block-mode rather than line-mode.
It uses EBCDIC character set rather than ASCII character set.
It uses special key functions such as ATTN and SYSREQ.

A 3270 Telnet (TN3270) server must support those characteristics during initial client/server session negotiations. TN3270 sessions can represent either display or printer devices.

Originally TN3270 sessions were identified as non-SNA devices to a mainframe computer. The TN3270E extensions (see RFC 2355) define methods to map TN3270 sessions to specific SNA logical unit (LU) names, thus effectively turning them into SNA devices. This eases the use of certain applications and allows users to be assigned the same LUs whenever they connect to the server.

2.8.2 File Transfer Protocols: FTP and TFTP

File Transfer Protocol (FTP) provides the functions of transferring files between two TCP/IP hosts. Since FTP is built on the services of TCP in the transport layer, it provides a reliable and end-to-end connection during the file transfer operation. Security is provided by the normal user ID and password authentication.

Trivial File Transfer Protocol (TFTP) is a somewhat simplified companion of FTP. It operates on UDP and therefore does not guarantee reliable end-to-end connection or delivery. It also offers only limited security based on client hostname authorization. Nonetheless, TFTP is quite commonly used in conjunction with BOOTP to distribute startup program code to diskless network stations (see 2.9.1, “Bootstrap protocol (BOOTP)” on page 41 for more information on BOOTP).

2.8.3 Remote printing: LPR and LPD

The line printer requester (LPR) allows access to printers on other computers running the line printer daemon (LPD) as though they were on your computer. The clients provided (LPR, LPQ, LPRM or LPRMON or LPRPORTD) allow the user to send files or redirect printer output to a remote host running a remote print server (LPD). Some of these clients can also be used to query the status of a job, as well as to delegate a job. For more information about remote printing, see RFC 1179.

2.8.4 Remote command execution: RSH and REXEC

Remote shell (RSH) and remote execution (RExec) are similar protocols that allow you to run programs and commands on different computers. The results are received and displayed on the local host. This can be useful for small computers to harness the power of large systems.

2.8.5 Domain name system (DNS)

Recall that TCP/IP hosts are addressed by 32-bit IP addresses that are represented in decimal notation. For example, to Telnet to a remote host with an IP address of 9.67.38.1, the user would typically enter Telnet 9.67.38.1. This approach can be cumbersome and error-prone, so a method was developed to use symbolic high-level machine names that are more meaningful to users than IP addresses.
This introduced the problem of maintaining the mappings between IP addresses and high-level machine names in a coordinated and centralized way. Initially, host names to address mappings were maintained by the Internet Network Information Center (NIC) in a single file (HOSTS.TXT) which was fetched by all hosts using FTP. Most hosts would have a copy of that file, which may or may not have been current or correct.

Due to the explosive growth in the number of hosts, this mechanism became too complicated and time-consuming, and was replaced by a new concept: the domain name system (DNS).

The domain concept lies in decentralizing the naming mechanism by distributing responsibility (and authority) for mapping between names and addresses. For example, consider the internal structure of a large organization. As the chief executive cannot do everything, the organization will probably be partitioned into divisions, each of them having autonomy within certain limits. Specifically, the executive in charge of a division has authority to make direct decisions, without permission from his chief executive.

Domain names are formed in a similar way, and will often reflect the hierarchical delegation of authority used to assign them. For example, consider the name:

small.itso.raleigh.ibm.com

Here, itso.raleigh.ibm.com is the lowest-level domain name, a subdomain of raleigh.ibm.com, which again is a subdomain of ibm.com, a subdomain of com. We can also represent this naming concept by a hierarchical tree (see Figure 16).

![Hierarchical namespace](image)
Table 7 shows some of the top-level domains of today's Internet domain namespace.

Table 7. DNS: Some top-level Internet domains

<table>
<thead>
<tr>
<th>Domain Name</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>com</td>
<td>Commercial organizations</td>
</tr>
<tr>
<td>edu</td>
<td>Educational institutions</td>
</tr>
<tr>
<td>gov</td>
<td>Government Institutions</td>
</tr>
<tr>
<td>mil</td>
<td>US Military</td>
</tr>
<tr>
<td>net</td>
<td>Major network support centers</td>
</tr>
<tr>
<td>org</td>
<td>Non-profit organizations</td>
</tr>
<tr>
<td>country code</td>
<td>ISO standard 2-letter identifier for country-specific domains</td>
</tr>
</tbody>
</table>

2.8.5.1 Mapping domain names to IP addresses

The mapping of names to addresses consists of independent, cooperative systems called name servers. A name server is a server program that holds a master or a copy of a name-to-address mapping database, or otherwise points to a server that does, and that answers requests from the client software, called a name resolver.

Conceptually, all Internet domain servers are arranged in a tree structure that corresponds to the naming hierarchy in Figure 16 on page 37. Each leaf represents a name server that handles names for a single subdomain. Links in the conceptual tree do not indicate physical connections. Instead, they show which other name server a given server can contact.

Figure 17 on page 39 graphically depicts the domain name resolution process that is summarized in the following steps:

- A user program issues a request such as the gethostbyname() sockets call. (This particular call is used to ask for the IP address of a host by passing the hostname.)
- The resolver formulates a query to the name server. (Full resolvers have a local name cache to consult first, stub resolvers do not.)
- 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 server(s), starting down from the root of the DNS tree or as high up the tree as possible.
- The user program will finally be given a corresponding IP address (or host name, depending on the query) or an error if the query could not be answered. Normally, the program will not be given a list of all the name servers that have been consulted to process the query.

The query/reply messages are transported by either UDP or TCP. DNS is conceptually defined in RFC 1034 and RFC 1035.
2.8.5.2 Reverse mapping
In some cases, it may be necessary to find a hostname for a given IP address. This is called reverse mapping and is a standard function of most DNS servers available today. A special domain, in-addr.arpa, is being used for inverse name queries.

2.8.6 Simple Mail Transfer Protocol (SMTP)
The Simple Mail Transfer Protocol (SMTP) is an electronic mail protocol with both client (sender) and server (receive) functions. Since SMTP is a rather old protocol, many aspects of modern electronic mail are missing in its definitions. It basically assumes that messages would only consist of plain text in 7-bit US ASCII format with a line length of no more than 1000 characters. For more information about SMTP see RFCs 821, 822 and 974.

2.8.7 Multipurpose Internet mail extensions (MIME)
To overcome the shortcomings of SMTP, a new architecture has been defined that allows for a much greater variety of what can be contained in an electronic message, such as:

- 8-bit text and lines longer than 1000 characters
- International code pages and character sets
- Binary and multimedia objects such as fonts, images, audio and video objects

Multipurpose Internet mail extensions, or MIME, is defined in RFCs 2045 to 2049 and currently has a state of draft standard. MIME does not solely apply to electronic mail; rather, it defines a way to incorporate different objects in any electronic message. For instance, it is used widely throughout the Internet today (see 2.11.3, “Hypertext transfer protocol (HTTP)” on page 46).

2.8.8 Post office protocol (POP)
The post office protocol (POP) is an electronic mail protocol with both client (sender/receiver) and server (storage) functions. POP allows mail for multiple users to be stored in a central location until a request for delivery is made by an
electronic mail program. A POP client typically has only one server mailbox, and all messages it retrieves are stored locally and then deleted at the server, thus using server resources efficiently. For more information about POP, see RFC 1939.

### 2.8.9 Internet message access protocol (IMAP)

Internet message access protocol (IMAP) is an electronic messaging protocol with both client and server functions. Similar to POP, IMAP servers store messages for multiple users to be retrieved upon client request, but IMAP clients have more capabilities in doing so than POP clients. IMAP allows clients to have multiple remote mailboxes to retrieve messages from and to choose any of those any time. IMAP clients can specify criteria for downloading messages, such as not to transfer large messages over slow links. Also, IMAP always keeps messages on the server and replicates copies to the clients. Transactions performed by disconnected clients are effected on server mailboxes by periodic re-synchronization of client and server. For more information on IMAP and its underlying electronic mail models, see RFC 2060 and RFC 1733.

### 2.8.10 Remote Procedure Call (RPC)

Remote Procedure Call is a standard developed by SUN Microsystems and used by vendors of UNIX systems and VM/ESA.

RPC is an API available for developing distributed applications. It allows programs to call subroutines that are executed at a remote system. The caller program (called client) sends a call message to the server process, and waits for a reply message. The call message includes the procedure's parameters and the reply message contains the procedure's results. RPC also provides a standard way of encoding data in a portable fashion between different systems called External Data Representation (XDR).

The concept of RPC is very similar to that of an application program issuing a procedure call:

- The caller process sends a call message and waits for the reply.
- On the server side, a process is dormant awaiting the arrival of call messages. When one arrives, the server process extracts the procedure parameters, computes the results and sends them back in a reply message.

### 2.8.11 Portmap

Portmap or Portmapper is a server application that will map a program number and its version number to the Internet port number used by the program. Portmap is assigned the reserved (well-known service) port number 111.

Portmap only knows about RPC programs on the host it runs on. In order for Portmap to know about the RPC program, every RPC program should register itself with the local Portmapper when it starts up.

The RPC client (caller) has to ask the Portmap service on the remote host about the port used by the desired server program.
2.8.12 Network File System (NFS)

The Network File System (NFS) enables machines to share file systems across a network. It allows authorized users to access files located on remote systems as if they were local. It is designed to be machine-independent, operating system-independent, and transport protocol-independent.

2.8.13 X Window System

The X Window System (hereafter referred to as X) is one of the most widely used graphical user interfaces, or bitmapped-window display systems.

Current X releases contain two numbers: A version number indicating major protocol or standards revisions, and a release number indicating minor changes. The current support is for the latest version X11.

There are two main components in X that communicate with each other:

2.8.13.1 X-server

This is a dedicated program that provides display services on a graphic terminal, on behalf of a user, at the request of the user's X-client program. It controls the screen and handles the keyboard and the mouse (or other input devices) for one or more X-clients. Additionally, it is responsible for output to the display, mapping colors, loading fonts and keyboard mapping. Typically X-server programs run on high-performance graphics PCs and workstations, as well as on X terminals, which are designed to run only the X-server program.

2.8.13.2 X-client

X-client is the actual application, designed to employ a graphical user interface to display its output. Typically, many X-clients compete for the service of one X-server per display per user. X-term and X-clock are two examples of X-clients.

2.8.13.3 Transport

The X Window System uses sockets to communicate over a TCP/IP network.

2.9 TCP/IP configuration and management protocols

This section provides a brief overview of some of the protocols used to configure and manage TCP/IP networks.

2.9.1 Bootstrap protocol (BOOTP)

The bootstrap protocol (BOOTP) enables a client workstation to initialize with a minimal IP stack and request its IP address, a gateway address and the address of a name server from a BOOTP server. Once again, a good example of a client that requires this service is a diskless workstation. If BOOTP is to be used in your network, then you must make certain that both the server and client are on the same physical LAN segment. BOOTP can only be used across bridged segments when source-routing bridges are being used, or across subnets if you have a router capable of BOOTP forwarding, also known as a BOOTP relay agent.

2.9.2 Dynamic Host Configuration Protocol (DHCP)

The Dynamic Host Configuration Protocol (DHCP) is based on BOOTP and extends the concept of a central server supplying configuration parameters to
hosts in the network. DHCP adds the capability to automatically allocate reusable network addresses to workstations or hosts, and it supports the following functions:

**Automatic allocation**
DHCP assigns a permanent address to a host.

**Dynamic allocation**
DHCP assigns a leased IP address for a limited period of time. This is the only mechanism that allows automatic reuse of addresses that were previously assigned but are no longer in use.

**Manual allocation**
The host's address is manually configured by the network administrator. You may have more than one DHCP server in your network, each server containing a pool of addresses and leases in local storage. However, you must not assign overlapping pools of addresses to several servers because, due to the lack of a server-to-server protocol, that would result in duplicate IP addresses being handed out by those servers. A client may be configured to broadcast a request for address assignment and will select the most appropriate response from those servers that answer the request. One big potential advantage with DHCP is a reduction in the workload required to manually configure addresses for all workstations in a segment.

A DHCP server does not need to be in the same subnet or on the same physical segment as the client. This means you then require a BOOTP relay agent. More information about DHCP can be found in RFC 2131 and RFC 2132.

### 2.9.3 Simple Network Management Protocol (SNMP)
With the growth in size and complexity of the TCP/IP-based networks, the need for network management became very important. The current network management framework for TCP/IP consists of the following:

- Structure of Management Information (SMI) describes how managed objects contained in the Management Information Base (MIB) are defined.
- Management Information Base, second version (MIB-II) describes the managed objects.
- Simple Network Management Protocol (SNMP) defines the protocol used to manage these objects.

A network management station executes network management applications that monitor and control network elements such as hosts, gateways and terminal servers. These network elements use a management agent to perform the network management functions requested by the network management stations. SNMP is used to communicate management information between the network management stations and the agents in the network elements.

### 2.9.4 Lightweight Directory Access Protocol (LDAP)
Any information system relies on some sort of repository to retrieve locations of information, such as data, resources, addresses, and so forth. These repositories are referred to as directories, much like a telephone directory. Without directories, information is not easily found and communication in distributed systems is impossible. Directories are typically used to hold user information (for
authentication purposes), resource locations (for information access), and a whole lot more.

Though directories may contain all sorts of things and every vendor may implement its own style (in fact, many have done so), the Internet more or less demands some sort of directory standard to maintain coherent information and communication among diverse systems, applications, and vendors. The directory standard of choice today is X.500, as adopted by the International Standards Organization (ISO).

The more complex a directory becomes, the more overhead and cost may be involved in accessing it. Lightweight Directory Access Protocol, or LDAP, specifies a simplified way to retrieve information from an X.500-compliant directory in an asynchronous, client/server type of protocol. For more information about LDAP and X.500 implementations, see RFC 1777 and RFC 2116.

### 2.10 TCP/IP routing protocols and techniques

This section provides a brief overview of some of the protocols used to update routing tables among routers in TCP/IP networks. This process is called dynamic routing because the routers take care that updates are sent automatically according to protocol specifications.

In contrast to that, static routing would require system administrators to enter all required routing information at every host and gateway in order for a TCP/IP internetwork to function as desired.

#### 2.10.1 Routing Information Protocol (RIP)

The Routing Information Protocol (RIP) creates and dynamically maintains network routing tables. RIP arranges to have gateways and routers periodically broadcast their routing tables to neighbors. Using this information, a RouteD daemon can update a host's routing tables. For example, RouteD determines if a new route has been created, if a route is temporarily unavailable or if a more efficient route exists.

For more information about routing using RIP, see RFC 1058.

The Routing Information Protocol Version 1 is commonly known as RIP. It uses a distance vector algorithm, which means it calculates the best path to a destination based on the number of hops in the path. Each hop represents a router through which a datagram must pass in order to reach the destination.

RIP is widely used and easy to implement, but it is known to have several limitations:

- The maximum number of hops is 15 (16 refers to an unreachable destination), making RIP inadequate for large networks that may have more than 15 routers on any single path.
- RIP is not a secure protocol. It does not authenticate the source of any routing updates it receives.
- RIP can not choose the best path based on delay, cost, reliability or load.
- RIP does not support variable length subnet masks.
• RIP can take a relatively long time to converge, or stabilize its tables after an alteration to the network configuration has occurred. This is because RIP sends updates to routing tables every 30 seconds, but it takes 3 minutes before an entry that has not been updated expires.

The Routing Information Protocol Version 2 (RIP-2) was created in order to fix some of the limitations of RIP. It has the advantages of being easy to implement and having a lower network overhead. This overhead includes network traffic and CPU time. RIP-2 can interoperate with RIP, and it also supports variable length subnetting and IP multicasting.

2.10.2 Open Shortest Path First (OSPF)

Open Shortest Path First (OSPF) is a complex protocol utilizing a link-state, shortest path first algorithm. In a link-state protocol, each router broadcasts link status information to each of its neighboring routers instead of distance vector information. Each neighboring router then propagates the status information to its own neighbors until the information has been sent to every router in the network. Each router then uses the status information to build a complete routing table utilizing a calculated cost for each link based on load, time delays, or reliability.

The biggest advantage of OSPF in comparison to either RIP or RIP-2 is the shorter time taken to converge after a change to the network. The link-state protocols will always stabilize the propagated routing tables much faster than the distance vector protocols.

OSPF supports variable length subnetting and IP multicasting. It also introduces the concept of areas, where the autonomous system is divided into areas, each responsible for its own topology. Area topology is not propagated to other areas; border routers maintain connectivity between the separate areas across an OSPF backbone, reducing the amount of routing information which must be exchanged.

See RFC 1583 for more information about OSPF.

2.10.3 Virtual Router Redundancy Protocol (VRRP)

Virtual Router Redundancy Protocol (VRRP) was issued to the IETF by IBM, Ascend Communications, Microsoft and Digital Equipment Corp. in April 1998 and is documented in RFC number 2338. Its status is proposed standard.

VRRP introduction

VRRP is designed to eliminate the single point of failure inherent in the static default route environment. VRRP specifies an election protocol that dynamically assigns responsibility for a virtual router to one of the VRRP routers on a LAN. The VRRP router controlling the IP address(es) associated with a virtual router is called the master, and forwards packets sent to these IP addresses. The election process provides dynamic fail-over in the forwarding responsibility should the master become unavailable. Any of the virtual router's IP addresses on a LAN can then be used as the default first hop router by end-hosts. The advantage gained from using VRRP is a higher availability default path without requiring configuration of dynamic routing or router discovery protocols on every end-host.

VRRP overview

VRRP specifies an election protocol to provide the virtual router function described earlier. All protocol messaging is performed using IP multicast
datagrams, so the protocol can operate over a variety of multiaccess LAN technologies supporting IP multicast. Each VRRP virtual router has a single well-known MAC address allocated to it. This document currently only details the mapping to networks using the IEEE 802 48-bit MAC address. The virtual router MAC address is used as the source in all periodic VRRP messages sent by the Master router to enable bridge learning in an extended LAN.

A virtual router is defined by its virtual router identifier (VRID) and a set of IP addresses. A VRRP router may associate a virtual router with its real addresses on an interface, and may also be configured with additional virtual router mappings and priority for virtual routers it is willing to back up. The mapping between VRID and addresses must be coordinated among all VRRP routers on a LAN. However, there is no restriction against reusing a VRID with a different address mapping on different LANs. The scope of each virtual router is restricted to a single LAN.

To minimize network traffic, only the master for each virtual router sends periodic VRRP advertisement messages. A backup router will not attempt to preempt the master unless it has higher priority. This eliminates service disruption unless a more preferred path becomes available. It's also possible to administratively prohibit all preemption attempts. The only exception is that a VRRP router will always become master of any virtual router associated with addresses it owns. If the master becomes unavailable then the highest priority backup will transition to master after a short delay, providing a controlled transition of the virtual router responsibility with minimal service interruption.

The VRRP protocol design provides rapid transition from backup to master to minimize service interruption, and incorporates optimizations that reduce protocol complexity while guaranteeing controlled master transition for typical operational scenarios. The optimizations result in an election protocol with minimal runtime state requirements, minimal active protocol states, and a single message type and sender. The typical operational scenarios are defined to be two redundant routers and/or distinct path preferences among each router. A side effect when these assumptions are violated (i.e., more than two redundant paths all with equal preference) is that duplicate packets may be forwarded for a brief period during master election. However, the typical scenario assumptions are likely to cover the vast majority of deployments, loss of the master router is infrequent, and the expected duration in master election convergence is quite small (<< 1 second). Thus the VRRP optimizations represent significant simplifications in the protocol design while incurring an insignificant probability of brief network degradation.

Figure 18 on page 46 shows a configuration with two virtual routers with the hosts splitting their traffic. This example is expected to be very common on actual practice. Half of the hosts install a default route to virtual router 1 IP Address (IP A), and the other half of the hosts install a default router to virtual router 2 IP Address (IP B).

This has the effect of load balancing the outgoing traffic, while also providing full redundancy.
2.11 Internet user applications and protocols

This section provides a brief overview of some of the protocols and applications that have made the task of using the Internet both easier and very popular over the past couple of years.

2.11.1 Network News

One application that is particularly popular on the Internet is Network News, also known as Usenet News. Based on the Network News Transfer Protocol (NNTP), users on the Internet can view and contribute to news groups covering topics such as science, education, computers, business, politics, recreation, sports, and many more. News groups are stored on news servers. NNTP is used for both server-to-server and client-to-server communication.

Clients use a news agent application, such as the IBM NewsReader/2, to retrieve articles from one or more news groups, and to post articles to one or more news groups. For more information about NNTP, see RFC 977.

2.11.2 Gopher

Gopher is a client/server protocol designed for information location and retrieval. The client function provides a menu-driven interface to access the files stored on a Gopher server. The server function allows descriptive names to be assigned to the files, making it easier to identify the content of each file. Gopher was designed at the University of Minnesota. For more information about Gopher, see RFC 1436.

2.11.3 Hypertext transfer protocol (HTTP)

The World Wide Web is a global hypertext system that was initially developed in 1989 by Tim Berners Lee at the European Laboratory for Particle Physics, CERN in Switzerland to provide an easy way of sharing and editing research documents among a geographically dispersed group of scientists.
In 1993 the Web started to grow rapidly, which was mainly due to the NCSA (National Center for Supercomputing Applications) developing a Web browser program called Mosaic, an X Windows-based application. This application provided the first graphical user interface to the Web and made browsing more convenient.

Today there are Web browsers and servers available for nearly all platforms. You can get them either from an FTP site for free or buy a licensed copy. The rapid growth in popularity of the Web is due in part to the flexible way people can navigate through world-wide resources in the Internet and retrieve them. Table 8 presents some statistics about the growth of the Web.

The number of Web servers is also increasing rapidly and the traffic over port 80, which is the well-known port for HTTP Web servers on the NSF backbone, has had a phenomenal rate of growth too. The NSFNET was converted back to a private research network in 1995, therefore comprehensive statistics of backbone traffic are not as easily available anymore, if they are at all.

Table 8. Growth of the World Wide Web

<table>
<thead>
<tr>
<th>Date</th>
<th>Web Sites (%)</th>
<th>Web Traffic (%)</th>
<th>FTP Traffic (%)</th>
<th>E-mail Traffic (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>June 1993</td>
<td>130</td>
<td>0.5</td>
<td>42.9</td>
<td>6.4</td>
</tr>
<tr>
<td>June 1994</td>
<td>2,738</td>
<td>6.1</td>
<td>35.2</td>
<td>6.4</td>
</tr>
<tr>
<td>March 1995</td>
<td>--</td>
<td>23.9</td>
<td>24.2</td>
<td>4.9</td>
</tr>
<tr>
<td>June 1995</td>
<td>23,500</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>June 1996</td>
<td>230,000</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>January 1997</td>
<td>650,000</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

The hypertext transfer protocol (HTTP) is a protocol designed to allow the transfer of hypertext markup language (HTML) documents. HTML is a tag language used to create hypertext documents. Hypertext documents include links to other documents that contain additional information about the highlighted term or subject. Such documents may contain other elements apart from text, such as graphic images, audio and video clips, and even virtual reality worlds (which are described in VRML, scripting language for that kind of elements).

HTTP is based on request-response activity. A client, running an application called a browser, establishes a connection with a server and sends a request to the server in the form of a request method. The server responds with a status line including the message’s protocol version and a success or error code, followed by a message containing server information, entity information and possible body content.

An HTTP transaction is divided into four steps:

- The browser opens a connection.
- The browser sends a request to the server.
- The server sends a response to the browser.
- The connection is closed.
On the Internet, HTTP communication generally takes place over TCP connections. The default port is TCP 80, but other ports can be used. This does not preclude HTTP from being implemented on top of any other protocol on the Internet, or on other networks. HTTP only presumes a reliable transport; any protocol that provides such guarantees can be used, but the mapping of the HTTP request and response structures onto the transport data units of the protocol in question is outside the scope of this document.

Except for experimental applications, current practice requires that the connection be established by the client prior to each request and closed by the server after sending the response. Both clients and servers should be aware that either party may close the connection prematurely, due to user action, automated time-out, or program failure, and should handle such closing in a predictable fashion. In any case, the closing of the connection by either or both parties always terminates the current request, regardless of its status.

What we have just described means that, in simple terms, HTTP is a stateless protocol because it keeps no track of the connections. To load a page including two graphics for example, a graphic-enabled browser will open three TCP connections: One for the page, and two for the graphics. Most browsers, however, are able to handle several of these connections simultaneously.

This behavior can be rather resource-intensive if one page consists of a lot of elements as quite a number of Web pages nowadays do. HTTP 1.1, as defined in RFC 2068, alleviates this problem to the extent that one TCP connection will be established per type of element on a page, and all elements of that kind will be transferred over the same connection.

However, if a request depends on the information exchanged during a previous connection, then this information has to be kept outside the protocol. One way of tracking such persistent information is the use of cookies. A cookie is a file that is sent to a client by a server to be filled in with information about the client and/or the user at the client. It can be retrieved and checked by the server at any subsequent connection. Because cookies are regarded as a potential security hole, a Web browser should allow the user to decide whether to accept cookies or not. A more secure way of client and server authentication is provided by the Secure Sockets Layer (SSL) which is described in 2.12.1, “Secure Sockets Layer (SSL)” on page 50.

2.11.3.1 Applets and servlets
When a Java program is started from inside an HTML (Web) page, it is called a Java applet, to distinguish it from a Java program that is executed from the command line or otherwise on the local system. Applets are downloaded via the Web browser from a server and, by definition, are somewhat limited in the way they can use resources of the local system.

Originally, no Java applet was supposed to touch anything locally outside of its JVM and could only communicate back to the server from which it was downloaded. With Java 1.1, applets can be signed with security keys and certificates and can therefore be authenticated. Thus, an applet can be authorized to access local resources, such as file systems, and it may communicate with other systems.
In order to spare resources on clients and networks, Java applets can be executed on the server rather than downloaded and started at the client. Such programs are then referred to as *servlets*. Though that method requires a significantly more powerful server, it is highly suitable for environments with medialess systems, such as network computers.

### 2.11.3.2 HotJava
HotJava is a Java-enabled Web browser, developed by Sun Microsystems, which lets you view Java applets.

### 2.11.3.3 JavaScript
JavaScript is an HTML extension and programming language, developed by Netscape, which is a simple object-based language compatible with Java. JavaScript programs are embedded as source directly in an HTML document. They can control the behavior of forms, buttons and text elements. JavaScript is used to create dynamic behavior in elements of the Web page. In addition, it can be used to create forms whose fields have built-in error checking routines. For more information about Java, check out the following URLs:

http://www.ibm.com/javainfo/

http://www.java.sun.com/

### 2.12 TCP/IP and Internet security

In this section we briefly introduce some concepts and protocols that allow you to establish various degrees of security in TCP/IP networks.

One may say that the Internet is great because there is so much information out there that can be accessed very easily and quickly. Electronic communication has become a lot easier because of the Internet, no doubt, but it can also be a dangerous thing at times.

Imagine that someone would get into your system and destroy data at random just because you forgot to implement security that could have prevented it. Or, worse, imagine someone would tap into your communication, learn your passwords and then use your account information to do electronic shopping.

One of the major concerns when providing commercial services on the Internet is providing for transaction security and communications security.

Information exchanges are secure if all the following are true:

- Messages are confidential.
- The information exchange has integrity.
- Both sender and receiver are accountable.
- You can authenticate both parties in the exchange.

There are certainly other ways of compromising information on the Internet that one might think of; so how should one go ahead to protect oneself against them?
2.12.1 Secure Sockets Layer (SSL)

Secure Sockets Layer (SSL) is a security protocol that was developed by Netscape Communications Corporation, along with RSA Data Security, Inc. The primary goal of the SSL protocol is to provide a private channel between communicating applications that ensures privacy of data, authentication of the partners and integrity.

SSL provides an alternative to the standard TCP/IP socket API which has security implemented within it. Hence, in theory it is possible to run any TCP/IP application in a secure way without changing it. In practice, SSL is so far only implemented for HTTP connections.

In fact, the protocol is composed of two layers:

- At the lower layer is the SSL Record protocol. It is used for data encapsulation.
- On the upper layer is the SSL Handshake protocol used for initial authentication and transfer of encryption keys.

The SSL protocol addresses the following security issues:

**Privacy**
After the symmetric key is established in the initial handshake, the messages are encrypted using this key.

**Integrity**
Messages contain a message authentication code ensuring the message integrity.

**Authentication**
During the handshake, the client authenticates the server using an asymmetric or public key.

SSL requires each message to be encrypted and decrypted and therefore has a high performance and resource impact. In addition, since only the server is authenticated, SSL is not suitable for applications, such as electronic banking, which require that the server authenticate their clients.

2.12.2 Firewalls

One way to deal with network security is to install a specialized server, a so-called firewall. Firewalls tend to be seen as a protection between the Internet and a private network. But generally speaking a firewall should be considered as a means to divide the world into two or more networks: One or more secure networks and one or more non-secure networks.

Imagine a company where all the departments are connected to the internal network, including sales, accounts, development and human resources departments. The administrator would like to be able to restrict access from the development department machines to the human resources department machines and from the sales department to the development department.

In order to provide maximum security, a good firewall design is paramount. The following factors should be considered in the firewall design process:

- Anything not explicitly permitted should default to denied.
• Increasing complexity leads to bugs, which lead to opportunities.
• The server should be kept in a physically secure environment.
• Provide extensive logging.
• Turn off known problems and non-essential daemons (applications and services).

Most of the firewalls available today offer one or more of the following services, some of which we briefly describe in the remainder of this section:

• Filtering gateways
• Proxy application layer gateways
• Circuit layer gateways (SOCKS servers)
• Domain name server hiding
• Mail handling
• Auditing and logging

Multiple technologies are needed to provide capabilities and protection. The IBM eNetwork Firewall, for instance, is based on IBM's technology and has been used for more than ten years to protect internal IBM IP networks.

2.12.2.1 Screening filters
The screening filter looks at each IP packet flowing through it, controlling access to machines and/or ports in the private network and possibly limiting access from the private network to the Internet. Screening filters operate at the IP layer and cannot control access at the application layer.

2.12.2.2 Proxy servers
Proxy servers are used to control access to or from the private network relaying only acceptable communications from known users.

Users in the private network can access an application, such as FTP, in the proxy server using their usual utilities (clients). Users authenticate themselves to the proxy server and can then access the application on the desired machine in the public network. Proxy servers can also be used from the public network to access applications in the private network, but this exposes login names and passwords to attackers in the public network.

2.12.2.3 SOCKS servers
SOCKS servers are like proxy servers without the requirement for double connections. With SOCKS, users can benefit from secure communication without needing to be aware that it is happening.

Users have to use new versions of applications called SOCKSified clients. The SOCKSified client code directs its requests to the SOCKS port on the firewall. Sessions are broken at the firewall, as they are with proxy servers. With SOCKS, however, the connection to the destination application is created automatically once the user is validated.

Both the client and the SOCKS server need to have SOCKS code. The SOCKS server acts as an application-level router between the client and the real application server. SOCKS V4 is for outbound TCP sessions only. It is simpler for
the private network user, but does not have secure password delivery so it is not intended for sessions between public network users and private network applications. SOCKS V5 provides for several authentication methods and can therefore be used for inbound connections as well, though one should be cautious with that. SOCKS V5 also supports UDP-based applications and protocols.

The majority of Web browsers are SOCKSified and you can get SOCKSified TCP/IP stacks for many platforms. For additional information, refer to RFC 1928, 1929, 1961, and the following URL:

http://www.socks.nec.com

2.12.3 IP Security Architecture (IPSec)

Several security mechanisms are available that address security, authentication and encryption at the IP layer rather than on upper transport or application layers. The IP Security Architecture defines two specific headers to provide security services for IP datagrams. They may be applied either separately or combined:

2.12.3.1 IP Authentication Header (AH)
The IP Authentication Header (AH) can be used to provide connectionless integrity and data origin authentication for IP datagrams, and optionally to provide anti-replay integrity. This header protects an entire IP datagram, including all immutable fields in the IP header. AH does not provide confidentiality (no encryption).

2.12.3.2 IP Encapsulated Security Payload (ESP)
The Encapsulating Security Payload (ESP) can be used to provide confidentiality (encryption), data origin authentication, connectionless integrity, anti-replay integrity, and limited traffic flow confidentiality. Unlike AH, ESP provides security only for the protocols encapsulated by it, not the protocol that carries it.

2.12.3.3 Key management
IPSec uses shared secret keys to protect every single conversation (called security association). This means that a pair of keys must be exchanged between two systems before they can use IPSec for secure one-way communication, and two key pairs must be exchanged for secure two-way communication.

Because keys are different for each security association, this process is repetitive for any number of hosts that a single system wants to securely communicate with. Even in a small network, this is virtually unachievable, so the IETF has endorsed a standard key management framework called Internet Security Association Key Management Protocol (ISAKMP) and a key determination protocol called Oakley.

Currently, IPSec key management is not yet defined as an Internet standard.

2.12.4 Virtual private networks

As we discribed in 2.2, “Internet standards and Request For Comments (RFC)” on page 8, anyone who wants to communicate directly over the Internet must register with the InterNIC and obtain valid IP addresses. However, such addresses are high in demand and short in supply. Moreover, organizations as well as private citizens have an increasing interest in communicating securely across the Internet, which is known as a rather insecure area to work in.
It would therefore be desirable to extend both the confidentiality and address flexibility of intranets across the Internet. That can be achieved by creating secure IP tunnels between the endpoint of a private network and another such endpoint or a remote system.

IP tunneling can be described and used in the following way:

- As traffic enters a tunnel, the original IP datagram is placed inside another IP datagram as a payload and optionally also encrypted.
- Only the addresses at the endpoints of the tunnels are visible to the Internet and therefore have to be registered.
- Addresses on remote networks are hidden from the outside by the tunnel and can be locally administered. This can be an effective way to prevent address spoofing attacks and effectively turns the Internet into a private network, hence the term *virtual private networks*.
- Traffic inside the tunnel can be encrypted and therefore totally hidden from snooping while traversing the Internet.

IPSec ESP provides a way of tunneling IP in a secure manner. Other tunneling protocols in use today are Point-to-Point Tunneling Protocol (PPTP), developed by Microsoft, 3Com and others, and Layer 2 Forwarding (L2F), developed by Cisco Systems. Based on these two protocols, the IETF is currently working on a standardized Layer 2 Tunneling Protocol (L2TP). As opposed to IPSec, PPTP, L2F and L2TP operate on the network interface layer and are therefore suited to encapsulate and tunnel a variety of protocols.
Chapter 3. TCP/IP for VM/ESA structure

This chapter describes how TCP/IP fits into the VM/ESA structure.

3.1 Control Program services

VM/ESA Control Program (CP) communications facilities provides links between programs in virtual machines. The TCP/IP socket interface uses the Inter User Communications Vehicle (IUCV) and the Virtual Machine Communication Facility (VMCF) to communicate with the TCP/IP virtual machines.

3.1.1 Inter User Communications Vehicle

An IUCV communication takes place between a source communicator and a target communicator. Each communication takes place over a predefined linkage called a path. Each communicator can have multiple paths, and can receive or send multiple messages on the same path simultaneously, as shown in Figure 19.

![Figure 19. IUCV two-way data transfer](image)

3.1.2 Virtual Machine Communication Facility

The Virtual Machine Communication Facility is part of the CP component of VM/ESA. VMCF provides virtual machines with the ability to send data to and receive data from any other virtual machine, like IUCV.

VMCF is maintained in VM/ESA only for compatibility: all new programs should use APPC/VM or IUCV for communication.

3.1.3 Console Communications Services

Console Communications Services (*CCS), which is part of CP program services, provides TCP/IP Telnet and a CMS user virtual machine with console communications. This line-mode interface permits requests to be passed between the user and Telnet virtual machines.
3.1.4 Logical Device Service Facility (LDSF)

The Logical Device Service Facility (LDSF) allows an application running in a virtual machine to within it create one or more logical devices. 3270 extended data streams are supported to enable logical devices to utilize full color, programmed symbol sets, and extended highlighting capabilities. TCP/IP Telnet uses LDSF full-screen interface to simulate a 3270 device.

For more detailed information, see VM/ESA CP Programming Services, SC24-5760.

3.2 TCP/IP virtual machines

TCP/IP under VM/ESA uses many disconnected machines for the different TCP/IP components. In general, each TCP/IP application is implemented in at least one disconnected machine. The communication to the outside world is done via the TCP/IP virtual machine, which is the TCP/IP main task. All servers (TCP/IP applications) communicate with the TCP/IP virtual machine within the local VM/ESA system, via Control Program (CP) functions.

Figure 20 shows a part of all available servers. It is not necessary to configure and start them all; it depends what TCP/IP services you need to provide with your environment.

TCP/IP Server for VM/ESA

Figure 20. The TCP/IP layered architecture for VM/ESA

TCP/IP virtual machine

The TCP/IP virtual machine handles all the interfaces to the network device, as well as all TCP/IP requests from servers or clients attached to TCP/IP for VM/ESA. The Telnet server is implemented within the TCP/IP virtual machine itself. It uses *CCS to create line mode VM/ESA connections for remote users and LDSF to deal with CMS users.
**CMS virtual machines**
Regular CMS users execute the client applications in their own virtual machine. These applications use VMCF or IUCV to communicate via the TCPIP virtual machine to their final destination, a server anywhere in the network.

**SNALINK virtual machine**
The SNALINK virtual machine allows TCP/IP to use an SNA backbone network. The TCPIP virtual machine sends IP datagrams to SNALINK so they can be routed through the SNA backbone network. SNALINK is a VTAM application running under the control of GCS.

Another example of a VTAM application running under the control of GCS is X25IPI that allows TCP/IP to use an X.25 network.

**VTAM virtual machine**
VTAM handles the SNA backbone network. It does not communicate with TCP/IP directly, but it is required for the SNALINK and X25IPI virtual machines.

**FTPSERVE virtual machines**
The FTPSERVE virtual machine serves client requests coming in via VMCF from the TCPIP virtual machine. The TCPIP virtual machine itself gets its requests from anywhere in the network and delivers them to the appropriate server. The file server is just an example for many other virtual server machines.

**Additional virtual machines**
Some of the additional virtual machines and the functions they perform are as follows:

- **ADMSERV** runs the Kerberos database remote administration server.
- **BOOTPD** responds to client requests for boot information using data defined in a BOOTP machine file.
- **DHCPD** responds to client requests for boot information using data defined in a DHCPD machine file.
- **LPSERVE** implements the LPD, which handles client requests to print a file.
- **NAMESRV** implements the DNS server.
- **NCGLBD** supports Network Computing System (NCS) administration.
- **NCSGLBD** runs the NCS Global Location Broker daemon (glbd).
- **NCSLLBD** runs the NCS Local Location Broker daemon (llbd).
- **NDBPMGR** provides Network Database (NDB) Port Manager support, and is used in conjunction with the NDBSRV01 server.
- **NDBSRV01** provides NDB system support, in conjunction with the NDB Port Manager (NDBPMGR) server.
- **PORTMAP** runs the Portmapper function for RPC systems that support the Network File System protocol.
- **REXEC** provides remote execution services for TCP/IP hosts that support the REXEC client.
- **ROUTED** implements the RouteD server, which manages IP route table entries.
- RXAGENT1 is the agent virtual machine used by REXEC to process anonymous rexec client requests.
- SMTP implements the Simple Mail Transfer Protocol (SMTP) server, which provides TCP/IP electronic mail support.
- SNALNKA provides SNA LU 0 connections between multiple hosts.
- SNMPD for the SNMP Agent.
- SNMPQE for the SNMP Query Engine.
- TFTPD transfers files between the BFS and TFTP clients.
- UFTD implements the UFT server.
- VMKERB runs the Kerberos authentication server.
- VMNFS implements the NFS server.
- X25IPI provides an interface which allows the TCPIP virtual machine to communicate with hosts that use the x.25 protocol

### 3.3 TCP/IP server initialization

After you complete the planning, installation and configuration of your TCP/IP System, you need to initialize the appropriate Conversational Monitor System (CMS) servers and the Group Control System (GCS) servers.

#### 3.3.1 TCP/IP CMS servers

All TCP/IP application servers that run under CMS share a common profile, TCPRFIL EXEC. It is copied to each server's 191 disk as PROFILE EXEC. You should never modify this file as it may be replaced by TCP/IP service procedures.

The profile accesses the common disks (198, 591, and 592) and then calls TCPRUN EXEC. TCPRUN determines what kind of server is running and invokes the appropriate server function. The kind of server is referred to as the server class. It is obtained from DTCPARMFILE you choose.

The DTCPARMFILE file contains all of the information needed to establish the necessary runtime environment and to start the server. Exits can be defined to override any value set by a DTCPARMFILE file. A description of the DTCPARMFILE file and the server initialization process can be found in 5.1.1, “DTCPARMFILE” on page 105.

Because the various tags in the DTCPARMFILE file are used to determine what special environments should be created, as well as the options or parameters that will be passed to the server, it may become necessary to determine the precise commands that are issued.

#### 3.3.2 TCP/IP GCS servers

Servers that run under the GCS operating system share a common profile, TCPRFIL GCS. It is copied to each server's 191 disk as PROFILE GCS. You should never modify this file as it may be replaced by TCP/IP service.

The profile search is for and runs userid GCS. The DTCPARMFILE file is not used by the GCS servers.
3.3.3 TCP/IP server startup

The TCP/IP server startup steps are shown in Figure 21.

Figure 21. TP/IP server startup workflow

The TCP/IP startup steps are described as follows:

1. PROFILE EXEC on 191 accesses requires CMS base disks and TCPMAINT 591 or 198 disk.
2. PROFILE EXEC calls TCPRUN EXEC.
3. Locate server and server class definitions in DTCPARMS files.
4. Call any server and global exits with SETUP parameters.
5. Prepare the execution environment, issuing any needed CP and CMS commands.
6. Call any server and global exits with BEGIN parameters.
7. Run the server.
8. Call any server and global exits with END parameters.
9. Return to CMS or logoff.

Each TCP/IP server in VM/ESA is installed with equal startup workflow. It is recommended that you configure the autolog facility in your TCPIP server startup. Once you start the TCPIP server in your system all TCP/IP virtual machines are autologged.
Chapter 4. Routing

Routing is the method by which a host or a gateway decides where to send a datagram. It may be able to send the datagram directly to the destination, if that destination is on one of the networks that is directly connected to the host or gateway. In a case where the destination is not reachable directly, the host or the gateway attempts to send the datagram to a gateway that is nearer the destination. The goal of a routing protocol is very simple: to supply the information needed to do routing.

This chapter provides a brief overview of IP routing. For more details on this subject, refer to *TCP/IP Tutorial and Technical Overview*, GG24-3376; *The Basics of IP Network Design*, SG24-2580; and *TCP/IP Function Level 320 Planning and Customization*, SC24-5847.

In this chapter we also include guidelines on how to establish fault-tolerant network connections to your VM TCP/IP system based on the Virtual IP Addressing (VIPA) concept.

This chapter covers the following topics:

- Routing terminology
- Overview of TCP/IP network routing
- Maintaining routing tables
- Dynamic routing using RIP
- Configuring the RouteD server
- Virtual IP addressing

4.1 Routing terminology

This section describes some of the more common IP routing-related terms and concepts.

4.1.1 General terms

**Direct delivery** The method of delivery used when the destination host is attached to the same physical network to which the source host is attached. An IP datagram can be sent directly, simply by encapsulating the IP datagram in the physical network frame.

**Indirect delivery** The method of delivery used when the destination host is not on a network directly attached to the source host. The only way to reach the destination is via an IP router or gateway. The address of the first of these routers (the first hop) is called an *indirect route* in the context of the IP routing algorithm. The address of the first hop is the only information needed by the source host.

**Router** A device or host that interprets protocols at the Internet layer and forwards datagrams on a path towards their correct destinations.

**Gateway** Another term for router. It is generally used to designate a router that is placed between networks or subnetworks.

**Daemon** A UNIX term for a background server process. In VM TCP/IP, there is one routing daemon, RouteD, that runs in the ROUTED virtual machine.
**Network interface** A device which connects a host to the network. There are several types of network interfaces. Open System Adapter (OSA) is an integrated hardware device that is controlled by VM TCP/IP. A channel-attached device, such as the IBM 2216, is a stand-alone, intelligent device.

**Network** A general term used to describe interconnected devices. There are three IP network classes: A, B, and C. They are differentiated by the number of bits in the IP address that represent the network identifier. The remaining bits identify the host. **Note:** in this chapter, we use the term network to refer to both subnetworks and non-subnetted networks.

**Subnetwork** A method to allow a network to grow without requiring new IP addresses. Subnetwork addressing divides the host identifier portion of the IP address into a subnetwork identifier and a host identifier. A **subnet mask** is used to specify how many bits represent the subnetwork. The remaining bits identify the host.

### 4.2 Overview of TCP/IP network routing

The following discussion provides an overview of the basic concepts of network routing.

#### 4.2.1 Routing tables

Every IP host is capable of routing IP datagrams through the network, whether as the originator of the datagram, or as gateway between two networks. To accomplish this task, each IP host maintains a **routing table**. The table contains:

- The IP addresses of remote hosts, networks, or subnetworks,
- The IP address of the host that should receive the datagrams intended for a particular host, network or subnetwork,
- A **metric** that is used to determine which route should be used in case there is more than one possible route to the destination. A route with a lower metric value is preferred over a route with a higher metric value,
- Optionally, the IP address of another host that is to receive packets when no appropriate routing table entry can be found. This called the **default route**.

A workstation or server typically has only two entries in its routing table: a route for the directly-attached network and a default route to a router on the directly-attached network.

A router, on the other hand, has entries that enable it to reach each remote network or subnetwork. The router is able to do this because it is attached to more than one network and knows the location of other routers on those networks. When the router receives an IP datagram it looks through its routing table to find the best route to the destination host.

The phrase “best route” is important because it forces us to define exactly what we mean by "best". In some cases there is only one way for a datagram to travel between two hosts, so that route is, by definition, the best route. When there is more than one possible route, we depend on the **route metric** to determine the best route. We can assign route metrics based on distance, speed, price,
reliability, or any other relevant factor. The route with the lowest metric value is the best, or preferred, route.

Figure 22 illustrates three subnetworks connected by routers A, B, and C, and the associated metric for each connection between routers. Consider an IP datagram to be sent from workstation A1 to workstation C1. If the links between the routers are all high-speed T1 lines, the choice of route is easy: choose the route with the fewest number of intermediate routers, or hops. In this example, that would be the direct link between router A and router C. But what if the direct link between A and C were a slow, dial-up line instead of a T1 line? In that case, we could assign a metric to the A-C link that is higher than the sum of the A-B and B-C metrics to ensure router A chooses the longer, but faster, route.

As you might imagine, whenever the topology of the network changes and routers are added to or deleted from the network, the routing metrics usually change, so each router must have their routing tables updated to reflect the current state of the network. Keeping track of metric changes is not a trivial task. We'll discuss it in more detail in 4.2.3, “Routing table management” on page 65.
4.2.2 Routing algorithm

Here is the algorithm which is used to choose the best route:

1. If the destination IP network is directly connected, find the associated hardware address and deliver the datagram directly over the associated interface.
   - In Figure 23, one IP datagram is destined for host 192.168.1.2. The destination network, the Class C network 192.168.1.0, is directly connected to the sending IP host via the interface with address 192.168.1.1. The IP datagram will be sent across this interface and delivered to the destination IP Host.

2. If the destination IP network is not directly connected, send the IP datagram to a directly connected IP router that has been selected as a first-hop towards the destination network, or send the IP datagram to a default router.
   - In Figure 23, another IP datagram is destined for host 192.168.2.2. The destination network, the Class C network 192.168.2.0, is reachable via the router at address 192.168.1.3, which is on a directly connected network. The IP datagram will be sent to this router.
   - If an IP datagram is to be sent to a destination network, for which there is no explicit entry in the IP layer routing table of the sending IP host, such an IP datagram will be sent to the default router at 192.168.1.3.

3. If the destination network cannot be found, report an error of “Network Unreachable” and discard the IP datagram.

The contents of the routing table can be displayed using the `NETSTAT GATE` command.

After issuing the `NETSTAT GATE` command, the following display will appear:
1. The word “Default” under NetAddress indicates the route used for routes that are not configured. For every destination that is not configured explicitly as a destination network in the GATEWAY statement, the FirstHop IP address is used.

2. Packets destined for subnet 9.12.3 (the value after applying the subnet mask to the destination IP address) are routed over link L2216. Note: VM TCP/IP uses a subnet mask in which the network part of the subnet mask is represented by zeros. In the above example, where 0.255.255.0 is displayed, most implementations would display it as 255.255.255.0.

3. If the first hop is an IP address, it is an indirect route through the indicated host.

4. In the FirstHop column, “<direct>” indicates that it is a direct connection. “HOST” in the subnet mask field indicates that the link is point-to-point.

If you are not using dynamic routing, you must define all routing information in PROFILE TCPIP.

### 4.2.3 Routing table management

The most complex task in configuring the TCP/IP network routing is establishing the routing tables. It is easy if you use a dynamic routing protocol on all your hosts, especially on the routers. But some hosts do not support dynamic routing protocols, so you have to set up the static routing tables on your own.

#### 4.2.3.1 Static routing

Static routing requires you to manually configure the routing tables yourself. This is part of the configuration steps when you customize TCP/IP. You must know the address of the first router on the way. The task of statically defining all the necessary routes may be simple for a small network, and has the advantage of avoiding the network traffic overhead associated with dynamic routing updates. It also allows you to enforce rigid control on the allocation of addresses and resource access. However, it will require manual reconfiguration if you move or add a network. In many configurations, static routing is sufficient to provide efficient routing. If your configuration is such that you do not have any backup routes to remote networks, there is no need for dynamic routing.
Static definitions are recommended if you have a very simple network or only one physical interface to the network. In this case the routing table is uncomplicated and there is no reason to choose dynamic routing. However, as your routing tables become more and more complex due to network growth, and if the VM TCP/IP system must act as a gateway, then it is far easier to let the system do the work for you by using dynamic routing. Another reason for using dynamic routing is to take advantage of redundant network interfaces and to enable fault-tolerant network attachments.

Static routing may be appropriate for IP hosts that primarily run server or client application programs.

Most workstations that run OS/2, DOS/Windows or Windows 95 belong in this category. VM systems most often also belong in this category, since the VM system executes server programs that are used by client workstations in your IP network. Depending on your network topology, you may be able to define a single static default route on such hosts. This is often desirable, especially for the bulk of IP hosts that are represented by workstations where we want to maintain the fewest and most simple configuration parameters possible. If more optimal routes than the default one exist for selected destinations, we can rely on normal ICMP redirects to optimize the IP layer routing tables in these IP hosts.

4.2.3.2 Dynamic routing
Dynamic routing removes much of the need for static definition of the routing table. The router addresses and information detailing available paths to a destination are built dynamically. These dynamically-built routing tables are automatically exchanged among various routers in your network. This sharing of the routing information enables the routers to calculate the best path through the network to any destination. You can prevent the update of specific routes by specifying them in the ETC GATEWAYS file.

Dynamic routing may be appropriate for IP hosts that interconnect different IP subnets in your IP network and route IP datagrams between these different sections of your network.

Such routers must be able to react quickly to changes in the IP network topology in order to compute new secondary routes if sections of the network become unavailable. To do so, they need to participate in dynamic routing protocols. In certain configurations a VM system may belong in this category. This is certainly the case when you want to implement fault-tolerant network attachments to your VM system. In such a configuration, your VM system relies on dynamic route updates from neighboring routers.

IP route changes may be transparent if you use a dynamic routing protocol. This is because changes in network topology are noticed by the involved IP hosts, and information pertaining to them is exchanged with all routing daemons that participate in dynamic route updates. Keep in mind that the network will need some time to converge. In the meantime, parts of the network may be unreachable. This is due to the propagation of route updates, together with the time needed to compute a new routing table in every router along the way.

An interior gateway protocol (IGP) is a dynamic routing protocol that is used as an application protocol between dynamic routing servers that run on TCP/IP hosts within a single autonomous system. The protocol is used by the servers to
exchange information about which IP routes the IP hosts know about. By exchanging IP routing information with each other, the servers can maintain a complete picture of all available routes inside an autonomous system. The dynamic routing servers maintain the IP layer routing tables in the hosts they run on via special I/O-control (ioctl) socket calls, so the IP layers on the individual hosts include these routes in their normal routing decisions.

The following section describes some of the more common interior gateway protocols.

**Routing Information Protocol Version 1 (RIP-1)**
RIP-1 uses a distance vector algorithm to calculate the best path to a destination based on the number of hops in the path.

RIP-1 is widely used and easy to implement, but has several limitations.

RIP-2 was created in order to fix some of the limitations of RIP-1. Specifically, RIP-2 implements variable length subnetting.

**Open Shortest Path First (OSPF)**
OSPF uses a Link State or Shortest Path First algorithm. The biggest advantage is the reduced time taken to converge after a network change. It will stabilize the propagated routing tables much faster than the distance vector protocols. However, in doing so, it will require more CPU time and may generate extra network traffic.

### 4.3 Dynamic routing using RIP

This section provides an overview of how RIP works, along with some of the characteristics and limitations of the protocol. In this section, we describe both the RIP Version 1 and Version 2 routing protocols.

#### 4.3.1 RIP Version 1

RIP-1 is a protocol that manages IP routing table entries dynamically. The gateways using RIP-1 exchange their routing information so that their neighbors learn of topology changes. The RIP-1 server updates the local routing tables dynamically, resulting in current and accurate routing tables. See RFC 1058 for a complete definition of the RIP Version 1 protocol.

RIP-1 is one of a class of algorithms known as **distance vector algorithms**.

Distance vector algorithms are based on the exchange of only a small amount of information. Each entity (gateway or host) that participates in the routing protocol is assumed to keep information about all of the destinations within the system. Each entry in this routing database includes the next gateway to which datagrams destined for the entity should be sent. In addition, it includes a **metric** measuring the number of hops to the entity. Distance vector algorithms get their name from the fact that it is possible to compute optimal routes when the only information exchanged is the list of these distances. Furthermore, information is only exchanged among entities that are adjacent.
4.3.1.1 RIP Version 1 limitations

Since RIP-1 is designed for a specific network environment, it has some limitations. These should be considered before implementing RIP-1 in your network.

- RIP-1 declares a route invalid if it passes through 16 or more gateways. Therefore, RIP-1 places a limitation of 15 hops on the size of a large network.
- RIP-1 uses fixed metrics to compare alternative routes, instead of actual parameters such as measured delay, reliability, and load. This means that number of hops is the only parameter that differentiates a preferred route from non-preferred routes.
- The routing tables can take a relatively long time to converge or stabilize.
- RIP-1 does not support variable subnet masks or variable subnetting because it does not pass the subnet mask in its routing advertisements. Variable subnet masking refers to the ability to assign different subnet masks to interfaces that belong to the same Class A, B, or C network. You can see an example in Figure 24, where we are using two different masks for the same Class A network, 10.0.0.0: a mask of 255.255.255.0 and a mask of 255.255.0.0.

![Figure 24. Variable subnet masks](image)

If we attempt to telnet from the RIP-1 section of the network to a telnet server in the OSPF section of the network, where the telnet server is located on IP address 10.2.0.255, we may see unpredictable results, depending on which router receives the telnet request. This is one of the reasons why all RIP-1 routers attached to the same IP network must use the same subnet mask.

Some routers can manage variable subnet masking. Nevertheless, even in networks with such routers capable of handling exports or exchanges between RIP-1 and RIP-2 or OSPF, some host addresses may not be accepted. If you have a network that uses such routers (IBM 2210, IBM 2216, IBM 3746-9x0 IP Router), always try to make the RIP-1 part of the network use the shortest subnet mask. Then, if the RIP-2 or the OSPF part of the network uses a longer mask, the RIP-1 part of the network should not experience problems.
RIP-1 does not support *discontiguous subnets*. Discontiguous subnets are built when interfaces belonging to the same Class A, B, or C network, but to different subnets, are not adjacent to each other. Rather they are separated from each other by interfaces that belong to a different network. With RIP-1, discontiguous subnets represent unreachable networks. The following figure shows how we might create a discontiguous subnet with network 10.0.0.0.

![Figure 25. Discontiguous subnets](image)

Subnetworks 10.0.1.0 and 10.0.3.0 both belong to the same Class A 10.0.0.0 network, but they are separated by a Class C network, the 192.168.1.0 network. Such a configuration is a discontiguous subnet configuration. If the two routers are based on RIP-1, routing information will not be exchanged correctly. The result will be that hosts on, for example, the 10.0.1.0 subnet are unable to exchange IP datagrams with hosts on, for example, the 10.0.3.0 subnet.

**Note:** In the network shown in Figure 25, if you had used a class A 10.0.0.0 subnet on the point-to-point link between the two routers, you would not have created a discontiguous subnet configuration and routing tables would have been managed correctly by RIP-1.

RIP-1 does not include the subnet mask in updates. When advertising routing information over an interface that belongs to another network, RouteD merges all its known subnet routes into one network route and advertises that over the other network. It does this because it cannot assume that the router reached over this other network knows the subnet mask of the first network. Only routers that have interfaces on the same network are supposed to have knowledge about the subnet mask.

If you find it necessary to build discontiguous subnets, you must use one of the following techniques:

- Static routing
• RouteD with filtering and some static definitions to reach any discontiguous subnets
• A TCP/IP dynamic routing implementation that uses the RIP Version 2 protocol

4.3.2 RIP Version 2

RIP-2 extends RIP-1 functions and is specified in RFC 1721, 1722, 1723 and 1724. RIP-2 protocol extensions provide features that include the following:

Route Tag
The Route Tag field is used to hold an attribute that is assigned to a route and must be preserved and readvertised with the route. The route tag field may be used to propagate information about a route that was acquired from an EGP (Exterior Gateway Protocol). RouteD does not generate route tags, but will preserve them in received routes and readvertise them when necessary.

Variable Subnetting
Variable-length subnet masks are included in routing information so that dynamically-added routes to destinations outside subnetworks or networks are reachable.

Next Hop
The purpose of the Next Hop field is to eliminate routing packets through extra hops in the network. An address in the next hop field indicates an "immediate" next hop, as opposed to the current multi-hop route. RouteD does not generate immediate next hops, but will preserve them if they are included in RIP packets.

Multicast for RIP-2 Packets
An IP multicast address 224.0.0.9 is reserved for RIP-2 packets. RouteD supports receiving multicasted RIP Version 2 packets on this address. RouteD only supports unicast and broadcast of RIP Version 2 packets.

Authentication of RIP-2 Packets for Routing Update Security
Authentication uses passwords that can be configured to be included in outgoing RIP-2 packets for authentication by adjacent routers as routing update security. Likewise, incoming RIP-2 packets are checked against a local authentication password. Any outgoing or incoming RIP-1 packets are not authenticated. For maximum security, configure RouteD so that it will supply and receive RIP-2 packets only and supply an authentication password. The authentication passwords are configurable on a per-interface (router) basis.

Routing Control
Routing controls are provided to selectively control which versions of RIP packets are to be sent or received over network interfaces. The switches should be set based on the routing capabilities of the network and are configurable.

Supernetting support
Supernetting is part of Classless InterDomain Routing (CIDR) function. It provides a way to combine multiple network routes into fewer "supernet" routes. This means that the number of network routes in the routing tables becomes smaller for advertisements. Supernet routes are received and sent in
RIP-2 messages. If local supernet routes are defined for RouteD, they will be advertised to adjacent routers. Local supernet routes are generated by RouteD for interfaces with subnet masks that are less than the value of the network class mask in value.

In a RIP-2 environment, both of the limitations of RIP-1, which are shown in Figure 24 on page 68 and Figure 25 on page 69, are relieved. Of particular value is the ability in a RIP-2 environment to work with discontiguous subnets, which is fully supported.

RIP-2 packets are compatible with existing RIP-1 implementations. A RIP-1 system will process RIP-2 packets, but without the RIP-2 extensions, and will broadcast them as RIP-1 packets to other routers. Note that routing problems may occur when variable subnet masks are used in mixed RIP-1 and RIP-2 systems. Like RIP-1, RIP-2 is based on a distance vector algorithm.

4.3.2.1 RIP Version 2 sample configuration
RIP-2 extends RIP-1 functions and relieves most RIP-1 limitations.

Following are some examples of a RIP-2 network.

Figure 26. Private addresses on intermediate links

With RIP-2, you can make much better use of private addresses on intermediate links. Using private address space, you can use your limited IP address space more efficiently.

Figure 26 is an example of private addresses (those in the 192.168 network) on intermediate links. All intermediate links in an intranet can be considered private, so they are not assumed to have the requirement for internet communication.

The routers must be dedicated ones that never establish IP communication as endpoint IP hosts with the outside world. The point-to-point link address never gets outside the local IP network. Only the officially assigned network address (in this case, the network address 9.0.0.0) is known outside. Hosts with the officially assigned address can communicate with the outside world through the external router.
As you can see in the previous figure, using a private address on intermediate links causes discontiguous subnets. While RIP-1 cannot support this network, as we mentioned before, RIP-2 can be used in such a network.

A client inside this network can access a VM host using a VIPA address configured on VM, but not private addresses assigned to particular network interfaces. VIPA is described in more detail in 4.5, “Virtual IP addressing (VIPA)” on page 78.

![Figure 27. RIP-2 and channel-attached routers](image)

Figure 27 is another example of private addresses being used on links between a VM host and channel-attached routers. In this sample network, only the VIPA addresses and the backbone network use subnets of the public address spaces. Note that the VIPA addresses and the backbone network are examples of discontiguous subnets of the class A 9.0.0.0.

If you want to use the private address space for point-to-point links, such as CTC or IUCV links, this is the only way to do it; that is, use subnets of the public network, in this example the 9.0.0.0 network.
Figure 28. Variable length subnet mask

Figure 28 is an example of a network with variable-length subnet masks. If every router in this figure supports the RIP-2 protocol, R3 will add the two routers to its routing tables when it receives a RIP-2 packet from R1 or R2. When R3 searches for a router in order to forward an IP packet, it will look in its IP routing tables based on the longest network prefix first.

Suppose R3 has an IP packet for 172.16.3.49 and its tables look as follows:
- 172.16.3.32/255.255.255.240 via R1
- 172.16.3.48/255.255.255.240 via R2
- 172.16.2.0/255.255.255.0 directly connected
- 172.16.1.0/255.255.255.0 directly connected

It will select the second routing table entry, and transmit the IP packet to R2 for delivery.

Even though VM IP does not support OSPF, you can still use OSPF in the IP backbone - it just requires intermediate routers that export routing information between RIP and OSPF and/or other routing protocols. On routers that support route exports, you configure which routing protocols are supported per network interface, and the router will then extract information from one protocol and export it into the other. Therefore, even with a RIP-1 based routing domain in an intranet, you can use OSPF as the backbone interior routing protocol.

4.3.3 RIP database

We previously said that each entity keeps a routing database with one entry for every possible destination in the system. An actual implementation is likely to need to keep the following information about each destination:

**Address** The IP address of the host or network.
**Subnet mask value** The subnet mask value for the destination network. The subnet mask value is available only if RIP-2 is used in a network.

**Gateway** The first gateway along the route to the destination.

**Interface** The physical network that must be used to reach the first gateway.

**Metric** A number indicating the distance to the destination.

**Timer** The amount of time since the entry was last updated.

### 4.3.4 Basic distance vector algorithm

The following procedure is carried out by every entity that participates in the routing protocol. This must include all of the gateways in the system. Hosts that are not gateways may participate as well.

- Keep a table with an entry for every possible destination in the system. The entry contains the distance \( D \) to the destination, and the first gateway \( G \) on the route to the network.
- Periodically, send a routing update to every neighbor. The update is a set of messages that contains all the information from the routing table. It contains an entry for each destination, with the distance to that destination.
- When a routing update arrives from neighbor \( G' \), add the metric associated with the network that is shared with \( G' \). Call the resulting distance \( D' \). Compare the resulting distance with the current routing table entries. If the new distance \( D' \) for \( N \) is smaller than the existing value \( D \), adopt the new route. That is, change the table entry for \( N \) to have metric \( D' \) and gateway \( G' \). If \( G' \) is the gateway from which the existing route came, i.e., \( G' = G \), then use the new metric even if it is larger than the old one.

### 4.3.5 RouteD normal operation

If you want to learn more about RIP-1 or RIP-2, start the RouteD server with the tracing flags enabled (see 4.4.3, “Define RouteD server characteristics” on page 77). You can study the trace output to understand RouteD server activities. The activities of the RouteD server are summarized as follows:

- At initialization:
  - Add routes for directly connected networks, subnetworks, and hosts.
  - Issue a request for full tables to all interfaces.
- Receive requests to send full tables to adjacent RIP routers.
- Receive broadcasted requests from HOME interfaces (non point-to-point only).
- Receive responses from adjacent RIP routers.
- Add new routes and replace existing ones with shorter metrics as network converges.
- Send responses to adjacent RIP routers for honoring requests and for RIP broadcasts (every 30 seconds).
- Advertise routes that are network unreachable and eventually delete these routes when no responses have been received before timeouts.

When you use a RouteD server, several routes might be added and maintained by the RouteD server. These kinds of routes are referred to as RIP routes.
RouteD is the only application that can add or delete RIP routes. Therefore, if RouteD is terminated, RIP routes are not deleted, but kept in the IP routing table forever.

To help maintain the integrity of the routing table, at initialization RouteD attempts to delete all RIP routes from the stack routing table, and then adds RIP routes based on BSDROUTINGPARMS and the optional ETC GATEWAYS file.

### 4.4 Configuring RouteD server

The routing daemon, RouteD, is a server that implements the Routing Information Protocol (RIP). When configured properly, the VM host running with RouteD becomes an active RIP router in a TCP/IP network.

The steps to configure RouteD are as follows:
1. Create the RouteD configuration file (ROUTEDCONFIG).
2. Define explicit routes (ETC GATEWAYS).
   - **Note**: If a default route is to be defined to a destination gateway or router, configure a default route in this ETC GATEWAYS file.
3. Define RouteD server characteristics (DTCPARMS).
4. Update TCP/IP stack (PROFILE TCPIP).

For more information on configuring the RouteD server, see *TCP/IP Function Level 320 Planning and Customization*, SC24-5847.

#### 4.4.1 Create the RouteD configuration file

The RouteD server supports sending and receiving both RIP-1 and RIP-2 packets. The RouteD server configuration file is used to determine the mode of operation. A sample RouteD configuration file is shipped as ROUTED SCONFIG on TCPMAINT 591. It should be copied to TCPMAINT 198 as ROUTED CONFIG and customized.

If a ROUTED CONFIG file is not found, the following default settings are used:
- **RIP_SUPPLY_CONTROL**: RIP1
- **RIP_RECEIVE_CONTROL**: ANY
- **RIP2_AUTHENTICATION**: KEY: none

**Note**: You have to be careful with RIP_SUPPLY_CONTROL. Adjacent routers that support only RIP-1 can misinterpret RIP-2 packets or even crash.

#### 4.4.2 Define explicit routes

The ETC GATEWAYS file is used to identify networks or hosts that are not on a directly connected network, but are still accessible through other gateways. The ETC GATEWAYS file is stored on TCPMAINT 198, which the RouteD server reads when it starts. If the file is changed, the RouteD server must be restarted.

In addition to the routing statements, there is also an OPTION statement that allows you to tailor how a particular link is managed. There are a number of
options and filters that can be used to control traffic flow in a RouteD environment. Figure 29 summarizes these filters.

<table>
<thead>
<tr>
<th>Option or Filter</th>
</tr>
</thead>
<tbody>
<tr>
<td>options gateway ip_addr block</td>
</tr>
<tr>
<td>options gateway ip_addr norecieve</td>
</tr>
<tr>
<td>options gateway ip_addr none</td>
</tr>
<tr>
<td>options interface.poll.interval time-value</td>
</tr>
<tr>
<td>options interface.scan.interval time-value</td>
</tr>
<tr>
<td>options interface name ip_addr block dest_ipaddr (1)</td>
</tr>
<tr>
<td>options interface name ip_addr forward dest_ipaddr fmask n.n.n.n(2)</td>
</tr>
<tr>
<td>options interface name ip_addr noforward dest_ipaddr fmask n.n.n.n</td>
</tr>
<tr>
<td>options interface name ip_addr forward.cond dest_ipaddr fmask n.n.n.n</td>
</tr>
<tr>
<td>options interface name ip_addr none (3)</td>
</tr>
<tr>
<td>options interface name ip_addr norecieve dest_ipaddr fmask n.n.n.n</td>
</tr>
<tr>
<td>options interface name ip_addr forward</td>
</tr>
<tr>
<td>options interface name ip_addr passive</td>
</tr>
<tr>
<td>options interface name ip_addr ripon</td>
</tr>
<tr>
<td>options interface name ip_addr ripoff</td>
</tr>
<tr>
<td>options interface name ip_addr receive dest_ipaddr fmask n.n.n.n (4)</td>
</tr>
<tr>
<td>options interface name ip_addr receive.cond dest_ipaddr fmask n.n.n.n</td>
</tr>
<tr>
<td>options interface name ip_addr supply off</td>
</tr>
<tr>
<td>options interface name ip_addr supply on</td>
</tr>
<tr>
<td>options interface name ip_addr key authentication_key</td>
</tr>
<tr>
<td>options interface name ip_addr nokey</td>
</tr>
<tr>
<td>options interface name ip_addr supply.control keyword (5)</td>
</tr>
<tr>
<td>options interface name ip_addr receive.control keyword (6)</td>
</tr>
</tbody>
</table>

Figure 29. RouteD GATEWAYS options summary

Figure 29 Notes:

1. The NORECEIVE filter is equivalent to the BLOCK filter.
2. The FORWARD filters are used to control which routes are advertised over which interfaces.
   
   The FMASK parameter can be used to apply the filter to multiple routes in just a single options statement. The FMASK is not a subnet mask; it is merely used to filter out multiple routes in a single options statement, such as:
   
   ```
   options interface tr1 9.1.1.1 forward 9.2.0.0. fmask 255.255.0.0
   ```
   
   Any route that matches 9.2.x.x will be forwarded over the tr1 interface and filtered out on all other interfaces.
3. NONE is used to clear all filters for a given interface.
4. The RECEIVE filters are used to control which routes are accepted over which interfaces.
5. Choose one of the following keywords:
   - RIP1
   - RIP2B
   - RIP2M
   - RIP2
   - NONE
6. Choose one of the following keywords:

- RIP1
- RIP2
- ANY
- NONE

Refer to *TCP/IP Function Level 320 Planning and Customization, SC24-5847* for detailed instructions on how to specify the various routing filters.

### 4.4.3 Define RouteD server characteristics

The DTCPARMS file searches for the server configuration definitions for each server. You should modify the DTCPARMS file if you need to specify any of the RouteD server input parameters. These parameters can be specified on the :Parms tag for the RouteD entry in the DTCPARMS file.

In addition, there is an SMSG interface that allows you to change these startup options after the RouteD server is started. Authorization to issue SMSG is determined by the OBEY list in the TCPIP profile.

Some of the input parameters control tracing:

- **-t** Activates tracing of actions by the RouteD server.
- **-t -t** Activates tracing of actions and packets sent or received.
- **-t -t -t** Activates tracing of actions, packets sent or received, and packet history.
- **-t -t -t -t** Activates tracing of actions, packets sent or received, packet history, and packet contents. The packet content trace displays the RIP network routing information.
- **-qt** Turns off all tracing.

See *TCP/IP Function Level 320 Planning and Customization, SC24-5847* for a complete list of the input parameters that can be specified for RouteD.

### 4.4.4 Update TCP/IP stack

In order to start the RouteD server, you have to update your PROFILE TCPIP.

#### 4.4.4.1 ASSORTEDPARMS

The following shows the ASSORTEDPARMS section of a PROFILE TCPIP which contains parameters used to support routing:

```
ASSORTEDPARMS
  VARSUBNETTING   (1)
  IGNORERedirect (2)
  NOSOURCEVIPA   (3)
  ; SOURCEVIPA   (3)
ENDASSORTEDPARMS
```

1. The variable subnet mask support of TCP/IP must be enabled if RouteD will be used to send or receive RIP-2 packets.
2. The RouteD server cannot recognize ICMP Redirect messages. You should specify the IGNOREREDIRECT keyword to allow the RouteD server to have full control over the IP routing table contents.
3. The SOURCEVIPA function is discussed in more detail in 4.5, “Virtual IP addressing (VIPA)” on page 78.
4.4.4.2 PORT
In the PORT statement, you must reserve port 520 UDP for ROUTED. Verify that
the following statements have been added to the PROFILE TCPIP:

```
PORT
  520 UDP ROUTED
```

4.4.4.3 OBEY
Use the OBEY statement to include the ROUTED user ID in the OBEY list. This
allows the RouteD server to change routing information. Verify that the following
statements have been added to the PROFILE TCPIP. Your OBEY statement can
list other privileged users, depending on your configuration.

```
OBEY
  P735FALT TCPMAINT ROUTED
ENDOBEY
```

4.4.4.4 AUTOLOG
You should include ROUTED in the AUTOLOG statement of PROFILE TCPIP.
The ROUTED server is then automatically started when you initiate TCPIP. Verify
that the following statement has been added to PROFILE TCPIP:

```
AUTOLOG ROUTED 0
```

4.4.4.5 BSDROUTINGPARMS

```
; Link Maxmtu Metric  Subnet Mask Dest Addr
BSDROUTINGPARMS TRUE
  EN1   1500   0    255.255.255.0  0      (1)
  TR1   1500   0    255.255.255.0  0      (1)
  LTR2  4000   0    255.255.255.0  0      (1)
  TO3CTCP 4096   0    0          192.168.252.2 (2)
  RAS   32768  0    0          192.168.236.2 (2)
ENDBSDROUTINGPARMS
```

1. Configure the BSDROUTINGPARMS for each network interface with the MTU
   value and subnet mask for use by RouteD.

2. These links are point-to-point links. If you want to transmit RIP-1 packets over
   a point-to-point link, you have to specify the -h option on the RouteD server
   startup option (see 4.4.3, “Define RouteD server characteristics” on page 77).
   The -h option will force RouteD to broadcast host routes and subnetwork
   routes for point-to-point links.

3. Note that the high-order bits of the subnet mask that comprise the network
   class mask must not be zero.

Note The specification of the VARSUBNETTING option requires that the
ASSORTEDPARMS statement be placed before the GATEWAY and
BSDROUTINGPARMS statements.

4.5 Virtual IP addressing (VIPA)

VIPA, in conjunction with the dynamic routing programs of VM TCP/IP, provides
the following functions:

- Fault-tolerant TCP connections providing automatic and transparent recovery
  from controller and interface failure.
• Support for defining primary and secondary interfaces to the same subnet and support to switch dynamically between these interfaces in case one of them malfunctions.

• Recovery from VM host failure where an alternate host has the necessary redundancy to back up a VM host.

• Routing improvements using the customized interface metrics, route forwarding options and VIPA.

For additional information about VIPA, refer to *TCP/IP Function Level 320 Planning and Customization, SC24-5847*

### 4.5.1 Benefits of VIPA

The purpose of VIPA is to free other TCP/IP hosts from dependence on particular network attachments to VM. Prior to VIPA, other hosts were bound to one of VM’s HOME IP addresses and, therefore, to a particular network attachment to VM. If the physical network attachment failed, the IP address became unreachable. Recovery required either network reconfiguration or clients with the ability to retry alternate destination addresses returned by a name server.

Figure 30 shows an example of an IP network based on this concept of end-user-initiated connection recovery.

![Figure 30. TCP connection recovery without VIPA](image)

Some commercial applications attempt recovery by trying every address returned by a name server; many just try the first address. If it becomes unreachable, the application gives up after having learned from TCP that the connection to the address has broken.

VIPA provides a virtual network attachment with a virtual IP address that other TCP/IP hosts can use to select a VM TCP/IP stack without also selecting a specific network attachment. If a specific physical network interface fails, the VIPA interface should still be reachable by another physical interface. Hosts that connect to VM TCP/IP applications can send data to a VM VIPA address via whatever paths are selected by the routing protocols. VIPA provides tolerance of failures of VM network attachment hardware.
As previously noted, VIP A uses a virtual network attachment and a virtual IP address. The virtual network attachment will always be active and will never fail. A virtual IP address will be the HOME address for the virtual network attachment, but there will be no physical interface associated with it. Inbound packets that have the virtual IP address as the destination can be routed through any one of the real physical interfaces to VM.

Figure 31 shows recovery of an inbound connection when VIPA has been implemented at a VM host.

Dynamic routing is required for VIPA to be useful.

The application must have re-transmission and timeout facilities. All TCP applications have that implicitly, because that is part of the TCP layer. UDP applications only have that if such functions are included in the UDP application. The UDP transport layer does not include any recovery functions.

In the example shown, myhost must be able to react to dynamic route updates being sent from VM. If myhost were behind routers, it would not need to use dynamic routing, but could rely instead on the intermediate routers to change their routing tables or on Internet Control Message Protocol (ICMP) redirects from intermediate routers. ICMP redirects are sent by a router to the sender of an IP datagram when the datagram should have been sent to a different router. It provides a means for a host to develop a better routing table.

From the point of view of downstream routers and hosts, it seems as though a virtual subnetwork has been created in a VM host. Network attachments, for example OSA-2 devices, act not only as network interfaces attached to an end node, but also as intermediate nodes. As you might easily imagine, creating a virtual network inside a VM system is the reason we have to use different

---

**Note**

Name servers should be configured to return the VIPA address(es) as the first or only IP address(es) of a VM host.
subnetwork addresses for VIPA and those IP addresses assigned to the physical interfaces attached to the VM system.

Figure 32 shows what the virtual subnetwork inside a VM system looks like from downstream IP nodes.

Figure 32. Virtual network in a VM/ESA system

If a TCP connection is established outbound from VM (a client program on VM connects to a server program in the IP network), the failure of a physical connection can still be surmounted, as you see in Figure 33.

Figure 33. Outbound connection recovery with VIPA

The way to make this happen is to ensure that the VM client program works with a socket IP address of the VIPA interface on VM, and not the IP address of a physical interface.

If you specify the SOURCEVIPA option in the ASSORTEDPARMS section of your PROFILE TCPIP, client programs on VM will use a VIPA address as its local
socket endpoint address, and recovery of the outbound connection can succeed. In case INTFA fails in the scenario shown in Figure 33, any connections between the VIPA address and servers in the IP network will be recovered over the INTFB interface.

With the coding of SOURCEVIP A, the VIPA address that most closely precedes the real interface's IP address in the HOME list will be used as the source IP address for outbound datagrams. In the example in the preceding figure, this means that an outbound datagram flowing over either INTFA or INTFB will carry the VIPALINK address of 10.0.3.1.

If both hosts are VM hosts or hosts that support VIPA, in general, all connections between the two hosts, no matter which direction, may be recoverable if the connections are established between two VIPA addresses.

If you have not implemented the SOURCEVIP A option, outbound connection recovery may fail. That is, the selection of a local IP address for outbound connections will work as always:

> Pick the IP address of the interface over which the connection request (the SYN segment) was sent. This is true for both directly and indirectly connected destinations.

If you have links for which you do not want to use the VIPA address as a source, specify these links before any VIPA links in the HOME list.

### 4.5.2 Host addressing strategies

A VIPA network must be a subnetwork that is separate from the networks and subnetworks used on real interfaces of the VM TCP/IP stack. If you use multiple VIPA addresses in one stack or if you use multiple stacks with VIPA addresses, the same network or subnetwork can be used for all the VIPA addresses as long as the host route broadcast option (-h) or the VIPA host route broadcast option (-hv) has been enabled for RouteD. If you use host routes, be sure to check that other routers in your network accept host routes; it is an optional feature according to RFC 1058 (RIP Version 1).

There is another alternative to support using the same subnetwork address for all VIPA addresses: use RIP-2. Even if you choose RIP-2 as a routing protocol in your network, you have to specify the -hv options to advertise a particular virtual IP address. In this case, the host routes will be transmitted with the subnet value 255.255.255.255. Since all RIP-2 routers are assumed to support this subnet mask value, each VIPA address will be reachable from downstream routers and IP hosts. You can define a VIPA interface even if you use static routes, but the desired fault tolerance in downstream routers will require manual updates to routing tables in the downstream routers. Obviously, dynamic routing is recommended when VIPA has been implemented.

Figure 34 on page 83 shows you one way to assign VIPA addresses to multiple TCP/IP stacks.
Figure 34. Separate VIPA networks or subnets

If you are in doubt whether your downstream routers support host routes, use a separate network or subnet for each VIPA network. In the example shown, three different VIPA networks have been implemented on three different Class C networks. The downstream routers will add network routes to their routing tables based on the network route advertisements from RouteD.

The problem with this approach is the potential lack of available networks or subnetworks in your IP address space, since the VIPA address must be a separate subnet from those that are used on any other interfaces of the VM host.

If your downstream routers support host routes as specified in RFC 1058, then you may share the same subnet across multiple VIPA network addresses, but you must use the -h or -hv option when you start the RouteD server; these options allow RouteD to advertise not only the network routes, but also the host or VIPA host routes. In Figure 35 on page 84 VIPA addresses for three separate hosts are selected from the same Class C network, 192.168.253.0.
Figure 35. One VIPA network or subnet

The RouteD server on VM advertises both a host route and a subnet route for the VIPA networks. The downstream routers will add the subnet route pointing to the first of the three VM systems in the example shown, which advertises it. The downstream routers will not replace the subnet routers when the other VM systems advertise their routes, but simply will add the host routes from the other two systems, ending up with one subnet route and three host routes. The subnet route is not used for anything, because a host route has precedence over a subnet route. Therefore, the routers will find one of the three host routes and will never use the subnet route.

When all VM systems share the same VIPA network address, you have to start their RouteD servers with the -h or -hv option. The -h and -hv options also will force RouteD servers sending RIP-1 packets to broadcast host routes for point-to-point links that interface with the VM stack: IUCV, CLAW, CTC, or SNA. Without these options, RouteD will broadcast RIP-1 packets containing only subnet routes for all interfaces. When your RouteD server is running RIP-2, routing information can be advertised via point-to-point links without the -h or -hv options. However, if you want to use VIPA addresses that belong to the same network, the -h or -hv option is still needed.

4.5.3 Using a secondary network interface for failure toleration

Fault tolerance is provided by VIPA, but only if you have redundant hardware. In Figure 36 on page 85, we have primary and secondary network interfaces (TR1 and TR2).
The first interface to a network is designated as the primary. The primary interface is used for outbound traffic except in case of interface failure, when outbound traffic will be switched to the secondary interface. When a primary interface becomes available again after an outage, RouteD will discover it and add it to the routing table.

There are, in general, two actions required in the event of an interface outage, since communication requires a route in both directions:

1. Switch outbound traffic to another adapter.
2. Notify adjacent routers to switch inbound traffic for VM to VM’s secondary network interface. Normally, downstream routers need some time to accept another route to a given destination. RIP-1 works on the concept of timers, as we discussed earlier: the stale route timer (90 seconds) and the route timeout timer (180 seconds).

How fast convergence can be completed depends on the number of routers in the network topology and the routing protocol.

### 4.5.3.1 Route killing

If we have directly connected LCS-attached networks connected via an OSA, RouteD on VM can use the route killing function to accomplish the switch of inbound traffic. An example of this is shown in Figure 37 on page 86.
A router will change the metric of a route to a new value if it is informed to do so by the router from which it originally learned about the route. By having RouteD send an IP datagram over TR2, but with the IP address of TR1 as the source IP address, it makes the other routers believe that this update comes from the same router from which they originally learned about the route; the routers will then raise the metric to 16 for these routes.

When RouteD on VM follows this message with new routes to the destination networks via the TR2 interface, these routes have a better metric than the current value of 16, and the other routers accept them right away without the 90- or 180-second delay they would otherwise have imposed.

The route killing function is also used to restore the original routing through the primary interface when it recovers from the failure.

Note that the route killing function is supported only over LCS links.

**4.5.3.2 Channel-attached routers**

If we have channel-attached routers, the recovery is not so simple, since route killing cannot be employed. The next diagram, Figure 38 on page 87, contrasts two configurations: one in which route killing works (A), and one in which it does not (B).
In configuration A, there are two interfaces to the same subnet from VM: primary and secondary. When LCS1 fails, interface switching to LCS2 occurs for outbound traffic at the next *interface poll interval*. Route kill will make RouterX update its routing table immediately after the interface switch. Therefore, inbound traffic from RouterX will flow to LCS2 just a few seconds after the interface switch and both inbound and outbound traffic will have moved to the LCS2 interface.

In configuration B, there are two separate point-to-point links from VM to two channel-attached routers. There are no primary and secondary adapters.

- If Router1 or the 10.0.4.0 link fails, VM will depend on normal RIP timeouts in Router1 and Router2 to switch inbound traffic to flow via Router2 instead of Router1.
- If RouterX supports stale routes, it will switch inbound traffic to flow via Router2 instead of Router1 after 90 seconds.
- If it was the 10.0.4.0 subnet that failed, Router1 will continue to advertise a route to the VIPA network for up to 180 seconds before it finally changes the metric to 16, allowing RouterX to accept a better route from Router2.
- If the channel-attached routers support triggered updates, they may switch inbound traffic within seconds.
- If the routers run OSPF, they may switch inbound traffic more quickly, based on link state changes generated by the router that detects a link failure.

For outbound traffic from VM, if a point-to-point link fails, VM will switch outbound traffic to the other channel-attached router at the next *interface poll interval* timer pop.

It is almost impossible to predict exactly how much time will elapse before both inbound and outbound traffic has been re-routed via the other channel-attached router. It depends on what the channel-attached routers support in terms of dynamic routing protocols and features. Examples have been seen where the...
network converged completely within 15 seconds based on the interface.poll.interval timer.

4.5.3.3 Example of route recovery using the secondary interface

In Figure 39, notice the coding in PROFILE TCPIP that causes the first interface on a subnet to become the primary interface (1). The succeeding interface becomes the secondary interface (2).

Figure 39. PROFILE TCPIP with VIPA

The VIPA interface is included in the BSDROUTINGPARMS (3), but there is no need to include a START command for the VIPA device (4).

An examination of the RouteD trace output in Figure 40 on page 89 reveals that no route is added for EN2, the secondary interface (5). This is not a problem,
since EN2 is still a fully normal and operational interface over which we can receive IP packets.

```
* Processing interface LNKVIPA1
  *************************************************
  Virtual interface
  This interface is not point-to-point
  Adding network route for interface
  Tue Mar 10 20:34:21 1998:
  ADD destination 192.168.251.0, router 192.168.251.2, metric 1
  flags UP state INTERFACE|CHANGED|INTERNAL|VIRTUAL timer 0
  Adding host route for interface
  ADD destination 192.168.251.2, router 192.168.251.2, metric 1
  flags UP|HOST state INTERFACE|CHANGED|INTERNAL|VIRTUAL timer 0
  *************************************************
* Processing interface EN1
  *************************************************
  This interface is not point-to-point
  Adding network route for interface
  ADD destination 9.0.0.0, router 9.24.105.76, metric 1
  flags UP state INTERFACE|CHANGED|INTERNAL timer 0
  Adding subnetwork route for interface
  ADD destination 9.24.105.0, router 9.24.105.76, metric 1
  flags UP state INTERFACE|CHANGED|SUBNET timer 0
  *************************************************
* Processing interface EN2
  *************************************************
  This interface is not point-to-point
  Add subnetwork route ignored, route exists on interface EN1 (5)
```

Figure 40. Startup of RouteD with VIPA

When the primary interface fails (in this case interface EN1), we see the messages shown in Figure 41 on page 90 as RouteD switches from the primary EN1 interface to the secondary EN2 interface.
4.5.4 Traffic splitting across multiple interfaces

If you have multiple interfaces, you may wish to split the traffic across them instead of having all the traffic using a single interface.

4.5.4.1 Splitting traffic using network interface metrics

One way to split traffic over multiple interfaces is to customize the interface metrics. Look at the example in Figure 42 on page 91.
Figure 42. Interface metric customization

All routes that are advertised over the EN1 interface will have 2 added to their metric, while those over TR1 will have a metric of 1. This is specified in the BSDROUTINGPARMS definition. As a result, routers in the network will prefer the TR1 interface over the EN1 interface for all traffic destined for VIPA address 192.168.250.1.

Figure 43 shows two LCS connections where outbound traffic from VIPA address 192.168.250.1 will be sent out over the interface to the 10.0.1.0 subnet that is listed first in the HOME list.

Figure 43. Inbound/Outbound traffic splitting

TR1 is the primary adapter and will be used for outbound traffic. TR2 has the lower metric in the routes broadcast to the router at 10.0.1.3; therefore TR2 will be preferred for inbound traffic. Because of this, we can split (but not balance) the
traffic in the network by managing both the sequence of interfaces in the HOME list and metric customization. If one of the interfaces becomes unavailable, both inbound and outbound traffic will use the same interface as long as the remaining route has not become unreachable due to being assigned a metric of 16.

When the primary interface becomes unavailable, RouteD will activate the route killing procedure. Then the outbound interface of VM IP will be switched to the secondary interface within the time specified for the interface poll interval. On the other hand, when the secondary interface fails, the route killing interface will not be started. So the communications between the VM host and downstream routers/hosts will not be available until the downstream routers/hosts change the metric of the route for the VM host to 16 and receive a RIP packet containing route information to the VM host. This would take at least 180 seconds based on the RIP protocol specification. Note that, in this case, VM host broadcasts route information via two network interfaces with different metrics. Therefore, the downstream routers/hosts cannot use the stale route algorithm.

4.5.4.2 Traffic splitting using channel-attached router metrics

You can implement inbound and outbound traffic splitting with channel-attached routers, too. Use the VM RouteD metrics to force downstream routers and hosts to prefer one channel-attached router over another. Use the channel-attached routers’ metrics to force VM to prefer a different channel-attached router. Backup capability is still present. In the example below, the failure of one of the channel-attached routers will cause all traffic to flow over the remaining router.

![Figure 44. Splitting via channel-attached routers](image)

The example in Figure 44 was implemented using two IBM 2216 Nways Multiaccess Connectors.

RouteD on VM and RouteD on OS/2 perform differently from each other in the following ways:

- The IBM 2216 adds the customized RIP-In-Metric of the interface over which the RIP route metric is received before it is installed in the routing table. When a routing table entry is advertised, the IBM 2216 adds to it the customized RIP-Out-Metric of the interface over which the route
advertisement is sent. The default RIP-In-Metric is 1 and the default RIP-Out-Metric is 0.

- VM RouteD adds 1 to a RIP route metric it receives before it is installed in the routing table. When a routing table entry is advertised, RouteD adds to it the customized metric of the interface over which the advertisement is sent.

You can think of VM RouteD as an IBM 2216 with a RIP-In-Metric of 1 and a RIP-Out-Metric equal to the value specified in the BSDRoutingParms profile configuration statement or an entry in the ETC GATEWAYS file.

Based on these rules, the following explanations apply to the metrics used in Figure 44 on page 92.

- Inbound routes to the VIPA network:
  - The VIPA network is directly connected to VM, and RouteD on VM assigns a metric of 1 to reach the VIPA network.
  - When VM RouteD advertises a route to the VIPA network, it will do so over the L1 interface with a metric of 3 (1+2), and over the L2 interface with a metric of 1 (1+0).
  - Both Router1 and Router2 will add 1 to the metrics they receive and Router1 will maintain a route to the VIPA network with a metric of 4 via the L1 interface. Router2 will maintain a route to the VIPA network with a metric of 2 via the L2 interface.
  - Both Router1 and Router2 will advertise their routes over the LAN interfaces, and our host at 10.0.3.3 will add 1 to the metrics of the route advertisements it receives from the two routers, ending up with two possible routes to the VIPA network:
    - One via Router1 with a metric of 5
    - One via Router2 with a metric of 3
  
The host at 10.0.3.3 will select Router2 as the preferred router for IP datagrams that are destined for the outbound routes from the VIPA network.

- Outbound routes from the VIPA network:
  - Both Router1 and Router2 are directly connected to the 10.0.3.0 subnet. Taking the customized metric of the LAN interface into consideration, Router1 will maintain a route in its tables to that subnet with a metric of 1 (1+0), while Router2 will maintain a route with a metric of 3 (1+2).
  - Both routers will advertise their routes to VM, and VM RouteD will add 1 to both of the routes that are advertised by the two routers, ending up with a route through Router1 with a metric of 2 and another route via Router2 with a metric of 4.
  - VM RouteD will select Router1 for outbound IP datagrams to the 10.0.3.0 subnet because its metric is less than that of Router2.

Obviously, you must do thorough testing with the channel-attached routers you decide to use if you are going to work with customized route metrics, since RouteD implementations do differ. You also have to be very careful not to create routing loops, which always is a potential risk when you begin to use customized metrics.
4.5.4.3 Splitting traffic using filters

You may also choose to do route filtering to influence the routes available to inbound and outbound traffic.

**Forward filter**

For example, you may tie a certain VIPA address to a specific interface by using the *forward* filtering option in the GATEWAYS file. This is shown in Figure 45.

![Figure 45. Route filter: Forward](image)

A route to the VIPA address 192.168.250.1 is broadcast only over the EN1 interface, while a route to the VIPA address 192.168.251.1 is broadcast only over the EN2 interface. This forces the downstream hosts to use two different interfaces depending on which VIPA address they send IP datagrams to.

The forward filter has two varieties:

- **Conditional**
  
  If the specified interface becomes unavailable, RouteD will begin to broadcast the route over any other available interface. This can be used to ensure that the route will be available even in backup situations.

  ```plaintext
  options interface TR1 9.24.104.74 forward.cond 192.168.250.1
  ```

- **Unconditional**
  
  If the specified interface becomes unavailable, RouteD will *not* begin to broadcast the route over other interfaces. The route will be broadcast only if the specified interface is up. This can be used to limit the use of available resources in a backup situation.

  ```plaintext
  options interface TR1 9.24.104.74 forward 192.168.250.1
  ```

**Noforwarding filter**

The *noforwarding* filter prevents a route from being broadcast over a specific interface while allowing it to be broadcast over others. It performs a function opposite to that of the forward filter. This is shown in Figure 46 on page 95.
We have used the noforwarding option here so that only the EN2 interface will be used for inbound traffic.

**Receive filter**

You can use the *receive* filter to select the outbound interface. In normal RouteD operation, if routes are configured with the same metric, the first received route is stored in the IP routing table and used to determine the next hop. In some network configurations, this logic would cause unpredictable results.

For example, assume a VM system is connected to two channel-attached routers, such as IBM 2216s. To reduce the number of route entries maintained in VM’s IP routing table, these two channel-attached routers can be configured to advertise only the default route (the RIP packet with IP address 0.0.0.0 in the IP destination address field) via channel links. In such a configuration, RouteD will use the first received RIP packet only and in most cases you cannot choose which router is used as the default.

In this situation, using the *receive* filter, you can choose a particular router as the default (see Figure 47 on page 96).
Since the receive filter supports conditional mode, if the specified interface becomes unavailable, the RouteD server can switch the default router to an alternative one as long as that server has received RIP default route packets from an alternative device.

**Block and noreceive filters**

Another technique available to you with route filtering is blocking. In contrast to metrics, which globally cause one interface to be preferred over another, blocking allows more discretion in limiting the use of a specific interface to a specific VIPA address. This is shown in Figure 48, where we combine the blocking filter with the use of route forwarding filters.

---

**Figure 47. Sample configuration in multi-stack environment**

---

**Figure 48. Route filter: Block**
Based upon a pure metric calculation, traffic between the VIPA address and H1 would go via the EN1 interface. However, by adding the definitions shown to the GATEWAYS file, we force traffic between the VIPA network and H1 to flow via the EN2 interface.

In contrast, hosts on the networks may access the 10.0.1.x host address as they like. We just force traffic to and from the VIPA network over the EN2 interface. The external route entry in the GATEWAYS file blocks the route received over EN1 completely.

In VM, the noreceive filter is also supported. It performs the same function as the block filter and also supports a subnet mask.

**Passive and ripoff filter**

If we want to isolate our host from RIP broadcasts over a selected interface, we can employ the passive filter, which is shown in Figure 49.

There are two CTCs between VM1 and VM2. Only one of them (CTC1) should be used to communicate between the two VIPA addresses. The other interface (CTC2) will be used only to communicate with the 192.168.1.0 network. The passive filter disables RouteD processing for CTC2. If the link extended into a completely different routing domain and we wanted to isolate the routing information exchange completely, we could use this filter on the link into the other domain. If the other end sends any RIP PDUs, our RouteD will ignore them.

In VM, the ripoff filter is also supported. It is functionally the same as the passive filter.

**Supply off filter**

An additional filter, supply off, can be used to eliminate RIP updates in one direction. (The passive filter causes RIP to be turned off in both directions.) The supply off filter could be used on links to other routing domains. With this option we will learn about routes in the other domain and be able to route IP packets into the other domain, but the other domain will not hear anything from us. For the
other domain to be able to route back to us, it would have to rely on passive route definitions into our domain. Figure 50 shows this scenario.

![Route filter: Supply off](image)

**4.5.4.4 Splitting traffic based on applications**

You may choose to split traffic based upon the applications being accessed. For example, you may want all traffic that will use the WEBSRV-A application to flow over EN1 and the traffic for WEBSRV-B to flow over EN2 (see Figure 51). In order to implement session traffic splitting, the server application on VM must be able to bind its listener port to a specific IP address instead of INADDR_ANY. One way to accomplish this specific sort of bind is to specify the interface IP address in the associated PORT statement in PROFILE TCPIP.

![Session traffic splitting](image)
Notice that this is one TCP/IP stack on VM, not two stacks. Our name server has been set up to recognize two different host names and to return two different VIPA addresses for these two host names. You could even customize a third host name with both VIPA addresses in the IP address list for all the other applications running on the stack. The GATEWAYS configuration file has been defined with the conditional forwarding filter so that in the event of an interface failure the routes can still be switched to other interfaces.

Notes:
- To use this function, applications have to be able to bind to a particular IP address. Use interface IP addresses in PORT statements to accomplish this.
- Even though applications can be associated with the specified IP address, you cannot select the outbound interface based on the applications. Since a TCP/IP stack maintains only a single IP routing table, if you want to split interfaces based on applications, you would have to configure multiple stacks, one stack for each application.
- Of course, an application can bind to an IP address associated with a real interface and use only this interface for its traffic, but then it won’t get the benefits of VIPA.

4.5.5 Tolerance of stack failures

If we have two TCP/IP stacks with VIPA implemented, we may want to back up one of the stacks should it fail. To do that, we use a function that is known as host takeover. What we take over is the VIPA address of the failing host. Clients in our network still establish connections to the failed host’s VIPA address, but now the backup stack responds to the connection requests. This is shown in Figure 52.

![Figure 52. Host takeover](image)

When VM1 fails, all connections with VM1 will fail too. In order to reestablish connectivity to the VIPA address of VM1, you can issue an OBEYFILE command (see Figure 52 for the commands in the configuration file read by OBEYFILE) to
assume the VIPA identity of VM1. When clients in the network try to reconnect to the VIPA address of VM1, VM2 responds and establishes the connection with the clients. When VM1 comes back up again, a new OBEYFILE command is run at VM2 to delete the VM1 VIPA entry from the HOME list and the BSDROUTINGPARMS (by removing the HOME entry and BSDROUTINGPARMS entry, the link and the VIPA address of VM1 become unavailable to VM2).

### 4.5.6 Example of fault-tolerant network using VIPA

In this section are two examples of using the various techniques we have presented to implement fault-tolerant VIPA networks.

**Example 1:**

This setup allows you to use inbound/outbound traffic splitting and session traffic splitting when all interfaces are available. By using an interface metric of 3 on A1 and A2, a metric of 2 on B1 and a metric of 1 on B2, VM will use A1 for outbound traffic. However, the downstream routers will use B2 for inbound traffic, which will effectively split inbound traffic not just over the OSA adapters, but also over the two token-rings, and the two backbone routers. If either A1 or B2 becomes unavailable, routing will just switch to use whichever interfaces are still up.

The host at address 10.0.4.3 does not need to run RouteD. It may just act on ICMP redirects from the two backbone routers. The two routers will know when routes become available or unavailable and they will send ICMP redirects to 10.0.4.3 if that host tries to use a router that currently is not the optimal one.
Example 2:

![Routing Example Diagram]

If the channel-attached routers allow you to customize interface metrics, you may be able to implement inbound/outbound traffic splitting in this scenario also. Route recovery time in this scenario depends on which dynamic routing protocol the two channel-attached routers are using and on the type of error situation.

If the link between VM and Router1 fails, both VM and the router are expected to detect this. VM will switch outbound to the other interface at the next RIP broadcast interval from Router2 within less than 30 seconds. When Router1 detects the link problem to VM, it will either send a link-state change to downstream routers (if it is using OSPF) and they will update their tables right away, or Router1 will broadcast a metric of 16 to the VIPA network, and downstream routers will change their inbound routes through Router2 at the next update from Router2. Based on triggered updates, this should last only a few seconds.

If Router1 dies without any last gasps of information to VM or Router2, the recovery may have to go through normal RIP timeouts. VM will switch outbound traffic after 90 seconds (stale route timer, as Router2 offers the same metric as Router1 to 10.0.4.0). Downstream routers will do the same based on updates from Router2 that offer a destination to the VIPA network with the same metric as Router1 did before. If we used customized metrics in VM to enable traffic splitting, the recovery time may be up to 180 seconds due to different metrics through Router1 and Router2. If the backbone network is using OSPF, this recovery will probably take less than 180 seconds.

4.5.7 Using VIPA with shared OSA-2 ports

You can use VIPA addresses over OSA-2 ports shared by one or more LPARs or TCP/IP stacks on a single VM image.

The architecture of the OSA allows port sharing. In this case the OSA must be able to determine what host path to use to send a packet that was received on a
shared port. In order to do that, the OSA must look into the packet and find the destination IP address. It then has to look up that IP address in the OAT table. If it finds a match, it will send the packet to the LPAR and device address associated with that IP address. If there is no match, the OSA will use the LPAR and device address that was configured as the default LPAR (if one exists).

With the changes introduced in OSA/SF APAR OW33394, you have the capability to define multiple IP addresses per unit address pair. A maximum of eight HOME IP addresses can be defined per entry and 16 addresses per port. This gives you the flexibility to have multiple VIPA addresses in your TCP/IP stack. Aside from APAR OW33394, you also need to ensure that the OSA-2 has the firmware level 6.37 or higher. Since we now have the capability to define more than one IP address per OAT entry, we can include the VIPA address together with the interface’s HOME address. This will allow the OSA to properly process any IP packet which has the VIPA as its destination address.

Aside from being able to define multiple addresses per OAT entry, you can now designate a secondary default LPAR which will serve as a backup for the primary LPAR.

Prior to OW33394, there are two ways to configure your TCP/IP and the OAT for VIPA to properly work using a shared OSA-2 port:

- Define the default LPAR and configure VIPAs on it.
  
  In this case, you can use VIPA addresses only on the default LPAR.

- Define dummy devices for VIPA addresses.
  
  In this case, you have to define additional ports and device addresses associated with the VIPA address on the OSA-2 adapter and dummy DEVICE/LINK/HOME statements in your PROFILE TCPIP. If you choose this technique, you can use VIPA addresses on every LPAR that is using the OSA-2 adapter.

**Sample OSA-2 configuration**

With OW33394 applied to your VM system, you can define the VIPA address in the OAT.

The following example has one port shared among the six stacks. This is not a recommended configuration. It is possible because there are other network attachments aside from the OSA-2. The network diagram in Figure 55 on page 103 was used in configuring a VIPA network.
Figure 55. VIPA with sharing OSA-2 port configuration

The following table shows our OAT definitions.

Table 9. Sample OAT definitions

<table>
<thead>
<tr>
<th>LPAR number</th>
<th>Unit address</th>
<th>Assigned IP address</th>
<th>Default LPAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>00,01</td>
<td>9.24.104.113, 192.168.250.3</td>
<td>Primary</td>
</tr>
<tr>
<td>01</td>
<td>02,03</td>
<td>9.24.104.33, 192.168.251.4</td>
<td>no</td>
</tr>
<tr>
<td>02</td>
<td>00,01</td>
<td>9.24.104.42, 192.168.252.26</td>
<td>Secondary</td>
</tr>
<tr>
<td>02</td>
<td>02,03</td>
<td>9.24.104.43, 192.168.253.29</td>
<td>no</td>
</tr>
<tr>
<td>03</td>
<td>00,01</td>
<td>9.24.104.149, 192.168.232.39</td>
<td>no</td>
</tr>
<tr>
<td>03</td>
<td>02,03</td>
<td>9.24.104.38, 192.168.234.38</td>
<td>no</td>
</tr>
</tbody>
</table>
The VIPA address is then defined in the A-stack in system RA03 as follows:

```plaintext
ASSORTEDPARMS
SOURCEVIPA
VARSUBNETTING ; For RIPV2
ENDASSORTEDPARMS
;***********************************************************************
; Device Definitions
;***********************************************************************
DEVICE VIPA3A VIRTUAL 0
LINK VIPA3A VIRTUAL 0 VIPA3A
DEVICE TR1 LCS 2060 autorestart
LINK TR1 IBMTR 0 TR1
...
;***********************************************************************
; Device Home Address
;***********************************************************************
HOME
  192.168.250.3  VIPA3A
  9.24.104.113  TR1
...
START TR1
```

*Figure 56. TCP/IP configuration for TR1 and VIPA*

The definitions for the other five stacks are almost the same as what we have shown.

The OAT entries for a VIPA address and IP addresses configured in the same TCP/IP instance have to be defined on the same OSA-2 port. The reason is that in order to establish communications between a VM host with a VIPA address and a router/host outside, a VIPA address and an IP address assigned to a physical network interface have to be associated with the same MAC address, that is, the same OSA-2 port.

If you configure the primary and secondary interface with shared OSA-2 ports, you have to configure an OAT entry for the VIPA address on not only the primary port but also the secondary port. When the primary and secondary interfaces are configured with virtual IP addresses, you can use the route killing function.
Chapter 5. Configuring TCP/IP

The virtual machines that are necessary to provide basic TCP/IP support, and the minidisk resources they require, are defined to your system as part of the installation of VM/ESA. The directory definitions for these virtual machines include links to the following minidisks used by TCP/IP:

- **TCPMAINT 591**  Server code including default configuration files
- **TCPMAINT 592**  Client code
- **TCPMAINT 198**  Customized configuration files

After installation, you can configure them to meet your installation's needs. Configuration files you change should be stored on TCPMAINT 198.

5.1 Configuration files

Following is an overview of the TCP/IP configuration files. See *TCP/IP Function Level 320 Planning and Customization*, SC24-5847, for additional information about these files.

5.1.1 DTCPARMS

The configuration of all servers is controlled by information in the DTCPARMS file. The DTCPARMS file contains 2 types of definitions: class and server. The class type defines the application protocols available; for example, FTP, RIP, and NFS. The server type defines the VM user ID for a particular TCP/IP server and indicates which class it serves. Parameters define the operational characteristics, such as the command to start the server, security, and parameters to be passed to the server.

An IBM DTCPARMS file is shipped with TCP/IP for VM/ESA Function Level 320. Each server definition includes the VM user ID for the server and associates it with a class. All other parameters are specified on the class definition.

Here is an excerpt from IBM DTCPARMS showing the server definition for the FTP server:

```
server:nick.FTPSERVE :type.server :class.ftp
```

It defines FTPSERVE as the VM user ID for this server and specifies that it handles class ftp.

Here is the definition from IBM DTCPARMS for the ftp class:

```
).* File Transfer Protocol (FTP) daemon
:nick.ftp :type.class
:name.FTP daemon
:command.SRVRFTP
:runtime.PASCAL
:diskwarn.YES
:anonymous.NO
:ESM_Enable.NO
:ESM_Validate.RPIVAL
:ESM_Racroute.RPIUCMS
```
To customize these parameters for your installation, you can create a file called SYSTEM DTCPARMS, nodeid DTCPARMS, or userid DTCPARMS. nodeid DTCPARMS is useful when you are using shared DASD among several VM systems; userid DTCPARMS is useful for testing purposes.

When a server is being initialized, the DTCPARMS files are searched in this order:

- userid DTCPARMS
- nodeid DTCPARMS
- SYSTEM DTCPARMS
- IBM DTCPARMS

The first file found will be used. When a server is being initialized, a search is made for its server definition. All parameters specified on the server definition are saved as override values. Then a search is made for the class definition specified in the server definition. The search ends when the first class definition is found. If you change either the server or class definition, it must be a complete replacement since parameters are not merged from various files. The recommended process is to copy the definition you want to change from IBM DTCPARMS to SYSTEM DTCPARMS or nodeid DTCPARMS and then make your changes. We recommend that you add the parameters you wish to change to the server definition. This leaves you less vulnerable to changes made by IBM with maintenance or a new release since IBM ships the default environments as part of the class definitions.

Another difference between making changes to the system definitions and the class definitions is the scope of the changes.

If you specify a parameter under the class type definition, it will apply to all instances of this class. If you specify a parameter under the server type definition, it applies only for that particular server.

For example, the following defines the parameter anonymous in the server definition. This setting applies only to the particular ftp server with the user ID FTPSERVE.

```
:nick.ftpserve type:server :class.ftp :anonymous.yes
```

In the following example, the anonymous parameter is defined for the class. Therefore, this setting will apply to any server that is associated with class ftp.

```
:nick.ftp :type.class :anonymous.yes
```

5.1.1.1 Our changes to DTCPARMS

Our work group created a file called SYSTEM DTCPARMS on TCPMAINT 198.

We added the :vctc tag to the stack server definition. The virtual channel-to-channel devices are defined and coupled to the specified virtual machine and address. The virtual machine is the TCP/IP server. Here is the stack definition in SYSTEM DTCPARMS:

```
:nick.TCPIP :type.server :class.stack
:vctc.812 tcpip2 812, 813 tcpip2 813
```
The changes made to SYSTEM DTCPARMS for other servers will be explained in the chapter that describes the server.

5.1.1.2 Using exits
There are also exits that can be used to provide configuration information.

You can use a server profile exit if you require more customization than is provided by the DTCPARMS file. You specify the name of the exit on the :Exit tag in DTCPARMS.

Also there is the global profile exit, TCPRUNXT EXEC, which can be used if you have processing common to all servers. A sample, TCPRUNXT SEXP, is provided on TCPMAINT 591.

For more information about these exits, see *TCP/IP Function Level 320 Planning and Customization*, SC24-5847.

5.1.2 TCP/IP PROFILE

When the TCPIP virtual machine is started, operation and configuration parameters are read from an initial configuration file. A sample initial configuration file is provided as PROFILE STCPIP. You can copy it from TCPMAINT 591 to TCPMAINT 198 and rename it PROFILE TCPIP or nodeid TCPIP. Then you can modify it to fit your requirements.

You should review this file to determine what changes need to be made for your installation. There are some parameters that may not need changing. For example, the defaults for the initial allocation of control blocks and data buffers used by TCP/IP may be satisfactory. However the following aspects of the configuration would likely need to be changed:

- Network device number
- IP address
- Subnet mask
- Default gateway

5.1.2.1 Our changes to the profile
We copied PROFILE STCPIP from TCPMAINT 591 as PROFILE TCPIP on TCPMAINT 198. Figure 57 on page 108 shows our system’s connection to the network. We made changes to PROFILE TCPIP to reflect this configuration.
We made the following changes:

1. Device

   DEVICE VMXACTC CTC 812

   gives the name and device address for the virtual channel-to-channel (CTC) we are using as our network interface device. TCP/IP uses device address and device address + 1, in this case, 812 and 813.

2. Link

   LINK WTSCVMXA CTC 1 VMXACTC

   specifies that the link is a CTC adapter and associates the name WTSCVMXA with it. It gives the name of the device defined in step 1. It specifies the adapter number to indicate whether 812 is the read or write address. We have specified 1 for this side of the connection indicating that 812 is the write address. The LINK statement for the other host would use adapter 0.

3. Home

   Home 9.12.13.62 WTSCVMXA

   specifies the IP address of our host, TOTVM1, and the link associated with it. The link was defined in step 2.

4. Gateways

<table>
<thead>
<tr>
<th>Network</th>
<th>First</th>
<th>Link</th>
<th>Max. Packet</th>
<th>Subnet</th>
<th>Subnet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address</td>
<td>Hop</td>
<td>Name</td>
<td>Size (MTU)</td>
<td>Mask</td>
<td>Value</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
<td>------</td>
<td>-------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>9.12.13.63</td>
<td>=</td>
<td>WTSCVMXA</td>
<td>1500</td>
<td>HOST</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>=</td>
<td>WTSCVMXA</td>
<td>1500</td>
<td>0.255.255.0 0.12.14.0</td>
<td></td>
</tr>
</tbody>
</table>

Figure 57. System configuration
; THIS IS THE DEFAULT GATEWAY THAT IS BEING USED TODAY
DEFAULTNET = WTSCVMXA 1500 0

The first gateway statement specifies that any packet destined for 9.12.13.63 should be sent over link WTSCVMXA. It also specifies that this destination is a host system (as opposed to a router).

The second gateway statement specifies that any packet destined for subnetwork 9.12.14 should be sent over link WTSCVMXA.

The defaultnet statement indicates that any destination with an address not covered by the first 2 statements should be sent over link WTSCVMXA.

5. Start statement

START VMXACTC

specifies that the device VMXACTC defined in step 1 should be started.

6. Syscontact and Syslocation

SYSCONTACT
    Main Operator (555-2150)
    Susquehanna Hat Company
ENDSYSCONTACT

SYSLOCATION
    First Floor Computer Room
ENDSYSLOCATION

This indicates the contact person for our host, including information about how to reach them. This is used by the Simple Network Management Protocol.

7. Translate

The translate statements were deleted because we were not using any protocols that required them.

5.1.2.2 Dynamic changes to the configuration

It is possible to make temporary changes to the system operation and network configuration without stopping and restarting the TCPIP virtual machine. You can maintain different files that contain TCP/IP configuration statements and use the OBEYFILE command to activate them. You must be listed in the obey list in the configuration file in order to issue this command. See TCP/IP Function Level 320 Planning and Customization, SC24-5847 for more details about using the OBEYFILE command.

5.1.3 TCPIP DATA file

The TCPIP DATA file defines system parameters used by TCP/IP clients. A sample TCPIP DATA file is provided as TCPIP SDATA on TCPMAINT 592. It should be copied to the same disk as TCPIP DATA and then modified.

You can provide configuration information for a single system or for multiple systems. If you want to specify different values for different systems for some parameters, you can prefix the configuration statement with the system name. If the statement is not prefixed then the value applies to all systems. If a statement is prefixed with a system name, the statement applies only to that system.

In the following example:

TOTVM1: TCPIPUSERID TCPIP
WTSCVMXA: TCPIPUSERID TCPIP2

the user ID of the TCPIP virtual machine (TCPIPUSERID) is TCPIP if TCPIP DATA was accessed from TOTVM1. If it is accessed from WTSCVMXA, the TCPIP virtual machine user ID is TCPIP2. The CMS IDENTIFY command is used to determine the system name.

The configuration statements and their meanings in TCP/IP data are:

• **ATSIGN**
  specifies an alternate hexadecimal value for @. Use this if your installation uses code pages in which the EBCDIC @ is not defined as X'7C'.

• **DOMAINLOOKUP**
  specifies what methods should be used to resolve host names and in what order. You can specify that both the name servers and the HOSTS SITEINFO file should be used and in what order. Or you can indicate that only one or the other should be used.

• **DOMAINORIGIN**
  specifies the domain origin that is appended to the host name when a complete local host name is formed; for example, ITSO.IBM.COM.

• **DOMAINSEARCH**
  specifies a list of domains to be searched during host name resolution. This allows resolution of a host name that is not in the domain specified in DOMAINORIGIN. For example, if DOMAINORIGIN is specified as ITSO.IBM.COM and you wanted to telnet to GDLVM7, you would need to specify the fully qualified name GDLVM7.ENDICOTT.IBM.COM. However if you specify ENDICOTT.IBM.COM in DOMAINSEARCH then the host name GDLVM7 can be resolved.

• **HOSTNAME**
  specifies the TCP host name of your VM host.

• **NSINTERADDR**
  defines the internet address of a name server.

• **NSPORTADDR**
  specifies the port of the name server.

• **RESOLVERTIMEOUT**
  indicates the number of seconds the resolver waits until a response is received.

• **RESOLVERUDPRETRIES**
  specifies the number of times the resolver should try to connect to the name server when using UDP datagrams.

• **RESOLVEVIA**
  specifies whether UDP or TCP should be used to communicate with the name server.

• **SMTPSERVERID**
  identifies the virtual machine that provides SMTP services if one exists.
• TCPIPUSERID
  specifies the user ID of the TCPIP virtual machine.

• TRACE RESOLVER
  causes a complete trace of all queries to and responses from the name server to be written to the user’s console.

• UFTSERVERID
  identifies the virtual machine that provides UFT services, if one exists.

• USERDATA
  specifies user-defined parameters.

5.1.3.1 Our changes to TCPIP DATA

Host name of our system:

TOTVM1: HOSTNAME TOTVM1

The domain to append to our host name to form the full host name:

DOMAINORIGIN ITSO.IBM.COM

Other domains to search when resolving a host name:

DOMAINSEARCH ENDICOTT.IBM.COM
DOMAINSEARCH POK.IBM.COM

Network address of our name server:


User ID of the UFT server:

UFTSERVERID UFTD

5.1.4 HOSTS LOCAL file

The information in the HOSTS LOCAL file, is used to create the site table which enables name resolution without using a domain name server. The site table is needed in order for the NETSTAT command to work correctly. If this table is not available you will see HOSTS ADDRINFO * not found in the NETSTAT output.

The HOSTS LOCAL file can contain three types of statements

• Host
• Network
• Gateway

The NET and GATEWAY statements are not used by the TCP/IP for VM/ESA Function Level 320 applications. Some socket calls require the NET entries for added performance. If your programs do not need these statements, delete them.

A sample file, HOSTS SLOCAL, is provided on TCPMAINT 592. You can copy it to TCPMAINT 592 as HOSTS LOCAL and modify it. In our HOSTS LOCAL file we had a single HOST definition and deleted the NET and GATEWAY statements. The following is our HOST definition:

The next step is to create the site table. From the TCPMAINT user ID, issue the command MAKESITE.

This produces the HOSTS SITEINFO and HOSTS ADDRINFO files on the A disk. Copy these files to TCPMAINT 592 so that they are available to all clients. You can use the TESTSITE command to test the correctness of the HOSTS SITEINFO file as follows:

```
testsite
VM TCP/IP Testsite
To quit, type 'quit'
Name:
totvml
TOTVML: Host address: 9.12.13.62
No gateway named 'totvml'
No net named 'totvml'
Name:
quit
Ready; T=0.01/0.02 17:52:01 his is screen.
```

5.2 Starting and stopping TCP/IP

This is information about starting and stopping the TCPIP virtual machine (the stack) and the other TCP/IP servers.

5.2.1 Starting TCP/IP

Add the CP XAUTOLOG TCPIP command to the PROFILE EXEC of the AUTOLOG1 virtual machine in order to start TCP/IP automatically when VM is started. The TCPIP virtual machine will then start each TCP/IP server identified on the AUTOLOG statement in the configuration file.

You can also start TCP/IP by issuing the CP XAUTOLOG TCPIP command from any authorized user, such as TCPMAINT.

5.2.2 Stopping TCP/IP

Issue NETSTAT CP EXTERNAL from a user listed in the obey list to stop TCP/IP in an orderly manner. The stack will begin to shut down. New connection requests will be rejected and outstanding socket requests will be cancelled with shutdown indications. After a 30-second delay, the TCP/IP stack will stop its network interfaces, terminate, and the TCPIP virtual machine will log off.

You may need to bring down a particular server; for example, if you want to retrieve a file from the server's 191 disk. To stop an individual TCP/IP server, do one of the following:

- Log on to the server to be stopped.
- Enter #CP IPL CMS.
- Use SMSG to send the command
- Use HX
Chapter 6. Terminal emulation

The Teletypewriter Network (TELNET) facility is a standard part of TCP/IP for VM, and includes support for both client and server functions. TELNET provides interactive access to local and remote systems, VM based or not, that are TCP/IP enabled. It supports TN3270 and TN3270E datastreams, but not graphics.

6.1 Concepts

TELNET is built on the services of TCP in the transport layer of the TCP/IP protocol suite. It uses the client-server model to support sessions between different terminals and hosts that may be running different operating systems.

The TELNET client opens a TCP connection with the TELNET server using well known port 23. The session is established using a protocol common to all TCP/IP enabled systems, called Network Virtual Terminal (NVT). This allows both sides of the connection to negotiate the terminal type to be emulated. Commands are exchanged and agreement is reached on the session characteristics that decide a native terminal type. Otherwise, NVT is confirmed, and this operates in half-duplex mode, line by line, using ASCII characters.

6.2 Implementation

The TELNET command is part of the TCP/IP client code that resides on the 592 disk owned by the TCPMAINT userid. End users can access this disk by using the VMLINK TCPIP command.

The TELNET server code is included in the TCP/IP stack and does not run as a daemon in a separate virtual machine.

6.3 Operations

TCP/IP for VM supplies a client function to allow a VM/CMS terminal to logon to a foreign host, which can be the local VM system or another remote operating system. The TELNET command initiates the connection. You must be logged on to VM/CMS in 3270 fullscreen mode when using this command, since there is no support for the TELNET client to use a linemode terminal.

By default, the session will operate in fullscreen (TN3270) mode. This is supported by the TELNET server function in TCP/IP for VM, as well as other TCP/IP enabled S/390 operating systems. Optionally, you may specify linemode operation, which is useful when accessing UNIX or Windows based systems where the TELNET server does not support 3270 datastreams.

For code translation purposes, TELNET TCPXLBIN is included because the STANDARD TCPXLBIN is not satisfactory. In addition, the TELNET command allows you to specify an alternate translate table, whose file type must be TCPXLBIN.

There are no operator commands to query or control the TELNET server function. Users can invoke the NETSTAT TELNET command to list active TELNET sessions.
For more information on the TELNET command and its subcommands, refer to the *TCP/IP Function Level 320 User's Guide*, SC24-5848.

### 6.4 Security

The TELNET implementation in TCP/IP for VM does not support user authentication or session encryption. Secured access to a foreign host by a VM TELNET client is limited to the logon userid and password facility provided by the host operating system.

### 6.5 Printing

TCP/IP for VM includes TN3270E support for sending 3270-based print files to TCP-connected printers. TN3270E statements are used to describe printer sessions to the TCP/IP server. Each definition links the IP address of a TELNET client with the logical unit name supplied by the client that represents the printer. A TN3270E printer exit is required to establish the session by connecting the printer to the virtual machine that wants to send the print files.

For more information on TELNET print support, refer to the *TCP/IP Function Level 320 Planning and Customization*, SC24-5847.
Chapter 7. Electronic mail

Electronic mail (e-mail) is one of the most widely used TCP/IP applications. The basic Internet mail protocols provide mail (note) and message exchange between TCP/IP hosts. This combined set of protocols is frequently referred to as the Simple Mail Transfer Protocol (SMTP). SMTP is limited to 7-bit ASCII text with a maximum line length of 1000 characters. Multipurpose Internet Mail Extensions (MIME) includes mechanisms to remove these and other restrictions in a manner that is highly compatible with the SMTP protocols. An SMTP server with MIME support is included in TCP/IP for VM/ESA.

The Post Office Protocol, Version 3 (POP3) is an electronic mail protocol with both client (sender/receiver) and server (storage) functions. POP3 supports basic functions (download and delete) for electronic mail retrieval.

The Internet Message Access Protocol, Version 4 (IMAP4) is an electronic messaging protocol with both client and server functions. Similar to POP3, IMAP4 servers store messages for multiple users to be retrieved upon client request, but IMAP4 clients have more capabilities for doing so than POP3 clients. IMAP4 allows clients to have multiple remote mailboxes to hold messages and to choose any of those at any time. IMAP4 clients can specify criteria for downloading messages, such as not to transfer large messages over slow links. Also, IMAP4 always keeps messages on the server and replicates copies to clients. Transactions performed by disconnected clients are effected on server mailboxes by periodic re synchronization of client and server. IBM has an IMAP project for VM/ESA for which more information may be found on the Web at:


For additional information on the POP, IMAP and MIME protocols see the chapter titled “Application Protocols” in the IBM Redbook TCP/IP Tutorial and Technical Overview, GG24-3376. For additional information on mail servers and clients available for VM/ESA see Appendix B.3, “Mail servers and clients” on page 198.

7.1 Simple Mail Transfer Protocol

Simple Mail Transfer Protocol (SMTP) is an electronic mail protocol with both client (sender) and server (receiver) functions. SMTP is implemented with the CMS NOTE and CMS SENDFILE EXEs in a VM/ESA environment. You do not interface directly with SMTP. Instead, electronic mail software is used to create mail, which in turn uses SMTP to send the mail to its destination.

7.1.1 Operation

The SMTP virtual machine operates as a mail gateway between TCP/IP network sites and RSCS network sites, also as native VM/ESA mail like CMS NOTE. This means, for example, OfficeVision users can exchange mail with UNIX workstations through the TCP/IP for VM/ESA SMTP gateway. The SMTP server allows you to create custom mail headers, provides an interface that allows the querying of SMTP mail delivery queues and provides a set of privileged commands for system administrator tasks. For more information about SMTP, see RFCs 821, 822, 974, 1413, and 1440.
7.1.2 SMTP server configuration

SMTP can be configured to use either a domain name server (see Chapter 2.8.5, “Domain name system (DNS)” on page 36 and Chapter 11.4.1, “Configuring the Domain Name Server” on page 176) or the local host name (see Chapter 5.1.4, “HOSTS LOCAL file” on page 111) tables. You can also specify unique startup parameters and customize the server environment by modifying the DTCPARMS file.

You should include SMTP in the AUTOLOG statement of the PROFILE TCPIP configuration file. The SMTP virtual machine is then started automatically when TCPIP is initialized. The SMTP server requires port TCP 25 to be reserved.

For a detailed discussion about SMTP configuration refer to Chapter 20 of TCP/IP Function Level 320 Programmer’s Reference, SC24-5849.

7.1.2.1 Using the SMSG Interface to SMTP

The VM Special Message Facility (SMSG) provides an interactive interface to the SMTP virtual machine to:

- Perform general user tasks such as querying the SMTP mail queues and build statistics of the SMTP virtual machine.
- Perform privileged system administration tasks, such as rebooting or shutting down the SMTP virtual machine and enabling or disabling various tracing and debugging options.

Note: Responses to commands are sent back to the originator of the command using CP MSG commands (or CP MSGNOH commands if SMTP is running with CP privilege class B).
Chapter 8. Transferring files

This chapter describes how you can use TCP/IP for VM/ESA Function Level 320 to copy data files between systems, view directories, and perform file maintenance. TCP/IP for VM supports File Transfer Protocol (FTP) and Trivial File Transfer Protocol (TFTP).

FTP provides access to data residing on a VM minidisk, the SFS, and the BFS.

TFTP provides access to data residing in the BFS.

TCP/IP for VM/ESA Function Level 320 provides support for FTP and TFTP clients and servers.

8.1 Concepts

FTP enables users to transfer files between two hosts across the network and is available on almost all computing platforms. The person retrieving files from another system does not need to have any familiarity with the remote host operating system. Users do need to be familiar with the FTP subcommands that allow them to connect to a remote host and transfer the files they want.

Whenever an FTP client is invoked, a TCP connection is established with the FTP server on the target host. Transfer of files is immediate. There is no use of spoolers or queues.

In addition to data transfer, FTP has subcommands to list and work with directories belonging to the local or foreign system, delete or rename files, communicate with the operating system and obtain status information about the FTP server on the foreign host. If the client is on a VM host, you can issue CMS commands from the client.

FTP also provides for authentication of the user. When a client opens a connection to a host system, a user ID and password are requested. There is also the ability to allow anonymous access, in which no password is requested.

TFTP is a simple way of transferring files between local and foreign hosts. Unlike FTP, TFTP uses the User Datagram Protocol (UDP), has no commands to list or manipulate directories and provides no user authentication. TFTP is generally used to provide access to public files needed by network computers, such as the IBM Network Station.

The remainder of this chapter focuses on FTP as it is the standard method of transferring files.

8.2 Configuration

The following is a description of the various configuration files that can be used to configure FTP for your installation.

8.2.1 CHKIPADR EXEC

CHKIPADR EXEC is an exit that can be used for security checking and to establish a default working directory on a per user basis.
When the FTP server is initialized, it checks if the CHKIPADR EXEC exists. If it does, it is called each time a user logs on. The IP address of the foreign host and the VM user ID are passed as input. A sample CHKIPADR SEXEC is shipped on TCPMAINT 591. If you want to use it, copy it to TCPMAINT 198 as CHKIPADR EXEC and modify it to suit your installation.

Ordinarily, the FTP server establishes a user's 191 minidisk as their default working directory. CHKIPADR EXEC can establish a different default working directory for a user. The default working directory can be any minidisk, SFS directory, or BFS directory.

If you wish to have a single welcome banner presented to all users connecting to the FTP server, place the desired banner text in the file FTP BANNER on TCPMAINT 198.

To display an additional, more personalized, banner for specific users, place the text in filename BANNER on TCPMAINT 198 and update CHKIPADR EXEC with:

```
queue "BANNER filename"
```

Since the VM user ID is passed to CHKIPADR, you may find it convenient to use the VM user ID as the file name. This banner is displayed in addition to FTP BANNER (if FTP BANNER exists).

Banners are displayed exactly as entered in the file.

### 8.2.1.1 Our changes to CHKIPADR EXEC

We copied the sample exec to TCPMAINT 198 and modified it to do the following:

- For user ID ANONYMOUS or ANONYMOU:
  - Set up an SFS directory as the default working directory.
  - Display a banner that gives information about the working directory and how to transfer files.
  - Return a value of 0.

  **Note:** Password requests are always suppressed for ANONYMOU or ANONYMOUS if the tag :ANONYMOUS.YES has been specified or ANONYMOUS has been specified on the :Parms tag (See 8.2.3, “DTCPARMS” on page 119).

- For user ID ANONFTP:
  - Set up an SFS directory as the default working directory.
  - Display a banner that gives information about the working directory and how to transfer files.
  - Return a value of 20 to suppress a password request.

- For other users:
  - Display a banner that gives a welcome message.
  - Return a value of 0.

We created the two banner files to be used: one for the anonymous user and one for everyone else.

### 8.2.2 PROFILE TCPIP

The sample profile TCPIP shipped with TCP/IP for VM has an entry to autolog the FTP server, FTPSERVE, and reserves ports 20 (data connection) and 21 (control
connection). Verify that these statements are in the PROFILE TCPIP on the configuration disk, TCPIBMINT 198. If you change the port FTP uses, you must change it in PROFILE TCPIP. In addition, you need to update the DTCPARMS entry for the FTP server. This is described in detail in the next section.

8.2.2.1 Our changes to PROFILE TCPIP
We did not need to make any changes to PROFILE TCPIP.

8.2.3 DTCPARMS

The parameters in the DTCPARMS file that apply to the FTP server are:

- **:Anonymous**
  
  A tag value of YES indicates that a user can FTP into your system and specify ANONYMOUS or ANONYMOU when prompted for a user ID. No password is requested. The setting for :ANONYMOUS in IBM DTCPARMS is NO.

- **:ESM_Enable**
  
  Indicates whether an external security manager is being used on your system. The setting in IBM DTCPARMS is NO.

- **:ESMValidate**
  
  If you set :ESM_Enable to YES and are using RACF, this parameter is already properly set in IBM DTCPARMS. If you are using another external security manager, specify the name of the module to be used to validate user IDs and passwords or NO if validation is not done by your external security manager.

- **:ESM_Racroute**
  
  If you set :ESM_Enable to YES and are using RACF, this parameter is already properly set in IBM DTCPARMS. If you are using another external security manager, specify the command to be used for initialization and termination of a RACROUTE environment or NO to indicate that no initialization of RACROUTE should be done.

- **:Parms**
  
  Use this tag if you want to:
  - Use a port number other than the default port 21.
    
    The tag value is Port *port_number*. This must match the port specified in PROFILE TCPIP. You might use this if you want to test a new version of the FTP server or a new exit.
  - Change the inactivity time-out from the default of 300 seconds.
    
    The tag value is INACTIVE *number_seconds* where *number_seconds* is a decimal number from 1 to 1 048 576. The connection is closed if there is no activity for the amount of time specified.
  - Trace FTP activity.
    
    The tag value is TRACE. Trace information is written to FILE DEBUGTRA on the FTPSERVE 191.
  - Restrict a foreign FTP client to using the same IP address for the control and data connections.
The tag value is DONTREDIRECT. If you consider using this, read the warnings in Chapter 11 of TCP/IP Function Level 320 Planning and Customization, SC24-5847.

- Load the FTP audit exit.
  
  The tag value is FTAUDIT. This causes the FTP server exit, FTPEXIT, to be loaded during initialization and enables audit processing. See 8.6, “FTP exit” on page 123 for more details.

- Load the FTP subcommand processing exit.
  
  The tag value is FTCHKCMD. This causes the FTP server exit, FTPEXIT, to be loaded during initialization and enables general command exit processing. See 8.6, “FTP exit” on page 123 for more details.

- Load the FTP change directory (CD) processing exit.
  
  The tag value is FTCHKDIR. This causes the FTP server exit, FTPEXIT, to be loaded during initialization and enables Change Directory (CD) command exit processing. See 8.6, “FTP exit” on page 123 for more details.

- Enable FTP reader file support.
  
  The tag value is RDR fm where fm specifies the file mode of the resource that can be used as a temporary storage before the file is put in the reader. This can be a minidisk, a temporary disk, a virtual disk or an SFS directory.

  To use the reader support, the FTPSERVE virtual machine must have Class D authority.

**Note:** The :Parms tag can also be used to specify Anonymous and RACF. We recommend, however, that the tags :Anonymous and :ESM_Enable be used instead of the :Parms tag.

Tag values for the :Parms tag should be separated by blanks. Any subsequent :Parms tag replaces the previous values, so you should not code multiple :Parms tags. If your tag values do not fit on one line, just continue them on the next line.

### 8.2.3.1 Our changes to the DTCPARMS file

We used the following server definition for the FTP server:

```plaintext
.* File Transfer Protocol (FTP) daemon
:nick.FTPSERVE :type.server :class.ftp
    :ESM_Enable.YES
    :Params.RDR D FTAUDIT
```

- :ESM_Enable.YES since our system was using RACF as the External Security Manager
- :Params.RDR D so that we could use the FTP reader support and FTAUDIT to load the FTP server exit, FTPEXIT, for audit purposes.

:FTP DATA. You can specify the following timeout values in the FTP DATA file to be used by the FTP client:

- Connection open timeout
- Data connection timeout for closing connection
- Control connection timeout for closing connection
- Inactive timeout value waiting for open
• Data connection timeout for Send/Receive

These values are used only by the client and are ignored by the server. There is one parameter which is used by both the client and server:

LOADDBSCTABLE

It is used to tell the client and server which DBCS translate tables should be loaded. Multiple tables may be loaded.

8.2.3.2 Our changes to FTP DATA

We copied the sample file FTP SDATA on the TCPMAINT 592 as FTP DATA on the TCPMAINT 592 but did not change any of the values.

8.3 VM considerations

When using FTP to access VM minidisks, all of the usual VM restrictions apply to the FTP environment. For example, multiple write access to the same minidisk is not permitted. This means, for example, that if you want to write a file to a VM minidisk (using the PUT command) that is accessed in write mode by another VM user ID, the FTP server will not link that minidisk in write mode and the PUT will fail.

Another VM consideration is that you will not see files with file mode 0 when a minidisk is linked in read-only mode.

8.4 Security

When a user connects to a system using FTP, the decision to allow the session is based on the user ID and password supplied. The user ID and password are verified by the External Security Manager (ESM) or by CP if no ESM is installed.

Once the session is established, all subsequent data access requests are performed using the VM user ID provided. The permission to access resources is based on this user ID. If an ESM is installed, it is used to determine if a user can access resources defined to it. If an ESM is not installed, access to minidisks and the reader (RDR) is determined by CP.

See Appendix A in TCP/IP Function Level 320 Planning and Customization, SC24-5847 for a detailed description of security issues.

8.4.0.1 CMS Shared File System

If the resource is a file in the SFS, the user must have write authority for the directory to put a file in it. To get a file from a directory, the user must have read authority for the file.

In order for FTP clients to access files in the SFS, the FTP server must have SFS file pool administrator authority. List all servers that will provide such access on the ADMIN statement in the SFS server's DMSPARM file.

8.4.0.2 Byte File System

If the BFS is being accessed, there are permission bits associated with each directory and file. These permission bits are used to determine what type of access is allowed. In order to put a file in a directory, the user must have write
permission for the directory. To get a file from a directory, the user must have read
permission for the file.

In order for FTP clients to access files and directories in the BFS, the FTP server
must be defined as a POSIX superuser. The FTPSERVE CP directory must
contain the following statement:

POSIXINFO UID 0 GID 0

8.4.0.3 Additional security
The FTP exit, FTPEXIT, can be used to provide further security checking. See
8.6, “FTP exit” on page 123 for information on this exit.

8.5 SMSG interface
You can control the FTP server from another VM user ID by using the CP SMSG
command. This user ID must be specified in the obey list in PROFILE TCPIP.
Here is an excerpt from that file showing the obey list:

; -----------------------------------------------------------------------------
; Define user IDs that are allowed to issue OBEYFILE and other
; restricted commands.
; -----------------------------------------------------------------------------
OBEY
   OPERATOR TCPMAINT SNMPD SNMPQE ROUTED REXECD DHCPD
ENDOBEY

Using the SMSG interface, you can:

- Drop an FTP connection
- Enable or disable an FTP exit
- Reload an exit
- Query user IDs or exits
- Release a file mode
- Turn trace on or off

In the following example, we issued SMSG to obtain information about all current
FTP users. We then used SMSG drop to sever the connection for the user who
was connected.

```
smsg ftpserve query *
Ready; T=0.01/0.01 17:48:19
17:48:19 FTPSERVE: IP Address: 9.117.32.29, Port: 38046
17:48:19 FTPSERVE: Login Date & Time: 1999-07-01 17:46:05
17:48:19 FTPSERVE: Total Data Bytes Transferred: 0.
17:48:19 FTPSERVE: Last Command: PWD
smsg ftpserve drop 2
Ready; T=0.01/0.01 17:48:35
17:48:35 FTPSERVE: Drop of connection 2 user LSA004MR is in progress
```
In the following example, we determined which exits were enabled, disabled FTAUDIT, and enabled FTCHKDIR.

```
smeg ftpserve query ftpexit
Ready; T=0.01/0.01 18:22:02
18:22:02 FTPSERVE: FTP CD command exit FTCHKDIR is disabled
18:22:02 FTPSERVE: FTP general command exit FTCHKCMD is disabled
18:22:02 FTPSERVE: FTP audit exit FTAUDIT is enabled
18:22:02 FTPSERVE: FTPEXit routine is active at '000716B8'X
smeg ftpserve ftaudit off
Ready; T=0.01/0.01 18:22:15
18:22:15 FTPSERVE: FTP audit exit FTAUDIT has been disabled
smeg ftpserve ftchkdir on
Ready; T=0.01/0.01 18:22:23
18:22:23 FTPSERVE: FTP CD command exit FTCHKDIR has been enabled
```

The changes made by SMSG remain in effect until the next time the server is IPLed or changed by another SMSG.

For more details on the SMSG interface, see TCP/IP Function Level 320 User’s Guide, SC24-5848.

### 8.6 FTP exit

The FTP exit, FTPEXIT, can be enabled for any or all of the following:

- **Audit processing (FTAUDIT)**
  
  The exit is called for every transfer of data over a connection, including data associated with files, as well as data returned in response to LIST, DIR or similar subcommands. Events such as login and logout are also audited.

- **FTP subcommand validation (FTCHKCMD)**
  
  The exit is called to perform command validation for every FTP subcommand received. The exit can be used to perform additional validation of a supplied user ID, IP address or the subcommand itself. The exit can indicate if the subcommand should be rejected with an exit-defined message returned to the user.

- **FTP directory change validation (FTCHKCD)**
  
  The exit is called to validate FTP directory changes, allowing greater control over access to system resources by providing the capability to selectively honor or refuse a client CD (change directory) request.

FTPEXIT can be loaded and initialized by specifying FTAUDIT, FTCHKCMD, or FTCHKCD on the :Parms tag in the definition for the FTP server (see 8.2.3, “DTCPARMS” on page 119) or by using the SMSG interface (see 8.5, “SMSG interface” on page 122).

A sample FTPEXIT ASSEMBLE routine and sample FTPEXIT EXEC are shipped on TCPMAINT 591. FTPEXIT ASSEMBLE is called by the FTP server. It calls FTPEXIT EXEC and passes the results back to the server. Assemble it by issuing VMFHLASM FTPEXIT DMSVM and place the resulting text deck on TCPMAINT 198. You can copy the FTPEXIT sample files to TCPMAINT 198 and modify them to suit your installation. A complete description of the FTP exit parameter list can
be found in Appendix C in *TCP/IP Function Level 320 Programmer’s Reference*, SC24-5849.

We used the sample FTPEXIT EXEC and enabled audit processing. A file, AUDIT FTPLOG, was created on FTPSERVE 191. Information about the login, logout and data transfer activity for each user was recorded. The following is an excerpt from this file:

```
```

An installation can modify the FTPEXIT EXEC to record information that is useful to it and optionally write programs to use the data; for example, to produce usage reports or billing records.

### 8.7 Multiple FTP servers

We did two variations of multiple servers:

- 2 servers listening on different ports
  - This could be used to test a new level of the FTP server or perhaps a new exit.
- 2 servers listening on the same port
  - Multiple FTP servers listening on port 21 can be configured to increase the throughput for concurrent FTP usage. It should be considered when throughput rates become slow due to FTP server congestion.

Multiple FTP servers is a solution even if you are not concerned about throughput. In VM when a minidisk is accessed, it is given a file mode that is a letter. Obviously the number of letters available is limited. A single FTP server may not be able to access more than 20 minidisks for file transfer purposes because it already has an A minidisk (its own), accesses some system minidisks (Y and S minidisks), and accesses TCPMAINT minidisks. So if you expect to have more than about 20 FTP sessions concurrently, you should define another FTP server in your VM system.

### 8.7.1 Configuration

Here are the steps we took to define a second FTP server with a VM user ID FTPSERV2:

- Create a VM user ID, FTPSERV2, using the FTPSERVE CP directory entry as a model.
- Copy TCPROFIL EXEC from TCPMAINT 191 to FTPSERV2 191 disk.
• Make the necessary RACF changes for the new server. See Appendix A in *TCP/IP Function Level 320 Planning and Customization*, SC24-5847 for information about these changes.

• Add FTPSERV2 to the Autolog list in PROFILE TCPIP.

  
  
  FTPSERV2 0 ; FTP SERVER

• Add a server definition for FTPSERV2 to SYSTEM DTCPARMS:

  
  We wanted the second server to handle RDR requests. We also wanted the auditing exit enabled.

  
  .* File Transfer Protocol (FTP) daemon
  :nick.FTPSERV2 :type.server :class.ftp
  :ESM_Enable.YES
     :Parms.RDR A PTAUDIT

  
  If this server will listen on a different port, add a Port parameter to the :Parms tag:

  
  .* File Transfer Protocol (FTP) daemon
  :nick.FTPSERV2 :type.server :class.ftp
  :ESM_Enable.YES
     :Parms.RDR A PTAUDIT Port 921

  
  It was not necessary to change anything in the class definition, ftp.

• Reserve the ports for FTPSERV2 in PROFILE TCPIP.

  The following indicates that the FTP servers listen on different ports:

  
  PORT
  20 TCP FTPSERVE NOAUTOLOG ; FTP SERVER
  21 TCP FTPSERVE ; FTP SERVER
  920 TCP FTPSERV2 NOAUTOLOG ; FTP SERVER 2
  921 TCP FTPSERV2 ; FTP SERVER 2

  
  The following indicates that both servers listen on the same port:

  
  PORT
  20 TCP FTPSERVE NOAUTOLOG ; FTP SERVER
  21 TCP FTPSERVE ; FTP SERVER
  20 TCP FTPSERV2 NOAUTOLOG ; FTP SERVER 2
  21 TCP FTPSERV2 ; FTP SERVER 2

• Log on to FTPSERV2 to start the server.

  
  8.7.2 Invocation

  By default, port 21 is always used for FTP. If a user wants to use server FTPSERV2, port 921 must be specified on the FTP command.

  
  If you are using the VM FTP client, you would issue:

  
  FTP TOTVM1 921

  
  8.8 FTP operations

  The following is a discussion of the usage of FTP, including use of the VM FTP command and sample sessions showing the FTP subcommands.
8.8.1 VM FTP command

You use the VM FTP command to enter the FTP environment where you can then transfer files between your local VM host and a foreign host. If you simply issue the command FTP you will be prompted for the name of the foreign host, the user ID for the connection and the password. The user ID is an ID known to the foreign host. The password is the password for that user ID. On some systems, you can specify ANONYMOUS rather than a user ID. See 8.10, “Anonymous access” on page 133.

Optionally you can specify the foreign host when you issue the FTP command; for example FTP TOTVM1. You will then be prompted for the user ID and password.

You can also specify parameters on the FTP command that control termination, tracing, timeout, and translation. If you are interested in any of these options, see TCP/IP Function Level 320 User’s Guide, SC24-5848 for more information.

8.8.1.1 NETRC DATA

You can specify a user ID and password for a remote host connection in the NETRC DATA file. We created a NETRC DATA file on our 191 disk with the following information (the password is given in this file but not shown here):

```
machine totvm1.itso.ibm.com login lsa004mr password password
```

When we connected to totvm1.itso.ibm.com, the user ID and password specified in the file were used. You can also specify an IP address instead of host name.

8.8.2 FTP sessions

There are many FTP subcommands that a user can issue during an FTP session. The following examples show the use of the most common subcommands.

```
* * Open a connection with the foreign host.
*

ftp wtscpok.itso.ibm.com
VM TCP/IP FTP Level 320
Connecting to WTSCPOK.ITSO.IBM.COM 9.12.14.1, port 21
220-FTPSERVE IBM VM Level at WTSCPOK.ITSO.IBM.COM
220 Connection will close if idle for more than 5 minutes.
USER (identify yourself to the host):
LSA004KH
>>>USER LSA004KH
331 Send password please.
Password:
[password entered here is not displayed]
>>>PASS ********
230 LSA004KH logged in; working directory = LSA004KH 191
Command:
*
* Ask the foreign host to describe itself.
*

system
>>>SYST
```
215-VM/ESA Version 2 Release 3.0, service level 9901
VM/CMS Level 14, Service Level 901
215 VM is the operating system of this server.

Command:

* Display the foreign host working directory.

pwd

>>>PWD
257 "LSA004KH.191" is working directory

Command:

* List entries on this foreign host minidisk.

ls

>>>PORT 9,12,13,62,4,97
200 Port request OK.

>>>NLST
125 List started OK
PRODUCT.CUR
PRODUCT.OLD
ACTIVITY.LOG
250 List completed successfully.

Command:

* Pass CMS command to the local host.


cms q accessed

Mode Stat Files Vdev Label/Directory
A R/W 16 191 LSA191
B R/W 2 192 LSA192
S R/O 706 190 MNT490
X R/O 1068 120 ALQ492
Y/S R/O 2048 19E Y-DISK
Z R/O 8255 19F Z-DISK

Command:

* Display the local host working directory.

lpwd

Local directory is 'A'

Command:

* Change to another minidisk on the local host.


cmp b

Local directory mode is 'B'

Command:
Set the transfer type between VM hosts to EBCDIC.

```
type e
>>>TYPE e
200 Representation type is EBCDIC.
```

Copy files from the foreign host to the local host.

```
mget product.* (rep
>>>PORT 9,12,13,62,4,111
200 Port request OK.
>>>NLST product.*
125 List started OK
250 List completed successfully.
>>>PORT 9,12,13,62,4,112
200 Port request OK.
>>>RETR PRODUCT.CUR
150 Sending file 'PRODUCT.CUR'
250 Transfer completed successfully.
328 bytes transferred in 0.041 seconds.
>>>PORT 9,12,13,62,4,113
200 Port request OK.
>>>RETR PRODUCT.OLD
150 Sending file 'PRODUCT.OLD'
250 Transfer completed successfully.
166 bytes transferred in 0.033 seconds.
```

Pass CMS command to the local host.

```
cms listfile * * b
PRODUCT CUR   B1
PRODUCT OLD   B1
```

Delete a file on the foreign host.

```
delete product.old
>>>DELE product.old
250 'product old' deleted
```
* Rename a file on the foreign host.

```
rename product.cur product.old
>>>RNFR product.cur
350 RNFR accepted. Please supply new name for RNTO.
>>>RNTO product.old
250 product.cur renamed to product.old
```

Command:

* Copy a file from the local host to the foreign host.

```
put product.cur product.cur
>>>SITE FIX recfm 80
200 Site command was accepted.
>>>PORT 9,12,13,62,4,133
200 Port request OK.
>>>STOR product.cur.a
150 Storing file 'product.cur'
250 Transfer completed successfully.
82 bytes transferred in 0.084 seconds.
```

Command:

* Access an SFS directory on the foreign host.

```
cwd sfstemp:lsa004kh.
>>>CWD sfstemp:lsa004kh.
250 SFS working directory is SFSTEMP:LSA004KH.
```

Command:

* List entries in the directory on the foreign host.

```
ls
>>>PORT 9,12,13,62,5,2
200 Port request OK.
>>>NLST
125 List started OK
PRODUCT.LOG
250 List completed successfully.
```

Command:

* Pass CMS command to the local host.

```
cms x product new a
```

Command:

* Append a local file to a file on the foreign host.

```
append product.newproduct.log
>>>SITE FIX recfm 80
200 Site command was accepted.
>>>PORT 9, 12, 13, 62, 5, 3
200 Port request OK.
>>>APPE activity.log
150 Appending to file 'product.log'
250 Transfer completed successfully.
82 bytes transferred in 0.056 seconds.
Command:

* * * Access a BFS directory on the foreign host. * * *
cwd .././VMBFS:.BFS:.ROOT//
>>>CWD .././VMBFS:.BFS:.ROOT//
250 BFS working directory is .././VMBFS:.BFS:.ROOT
Command:

* * * Create a new directory on the foreign host. * * *
mkdir prod
>>>MKD prod
257 New directory/..:/VMBFS:.BFS:.ROOT/prod/ has been created.
Command:

* * * Change the foreign host working directory. * * *
cwd prod
>>>CWD products
250 BFS working directory is .././VMBFS:.BFS:.ROOT/prod/
Command:

* * * Copy a file from the local to the foreign host. * * *
put product.cur
>>>SITE FIX recfm 80
200 Site command was accepted.
>>>PORT 9, 12, 14, 1, 199, 144
200 Port request OK.
>>>STOR product.cur
150 Storing file 'product.cur'
250 Transfer completed successfully.
82 bytes transferred in 0.64 seconds.
Command:

* * * Put a file in the reader on the foreign host. * * *
Transfering files

cd lsa004kh.rdr
>>>CWD lsa004kh.rdr
250 Working directory is LSA004KH.RDR
Command:
put product.cur
>>>SITE VARrecfm
200 Site command was accepted.
>>>PORT 9,12,14,1,199,144
200 Port request OK.
>>>STOR product.cur
150 Storing file 'product.cur'
250 Transfer completed successfully.
82 bytes transferred in 0.64 seconds.
Command:
*
* Terminate the FTP session.
*
quit
>>>QUIT
221 Quit command received. Goodbye.
Ready; T=0.12/0.26 09:46:40

Notes:

- You can use a plus sign (+) preceded by a blank at the end of a line to indicate continuation of input for an FTP subcommand.
- The VM FTP client sends a SITE subcommand containing record format information when you issue a PUT or MPUT subcommand. However, this information is only meaningful if the remote host is S/390. You can issue the SENDSITE subcommand (a toggle) to prevent the record format information from being sent.

8.8.3 Naming files

On the GET, MGET, PUT, and MPUT subcommands, filenames are specified. The format of these names is dependent on the system. For CMS, the format is filename.filetype.filemode. Filemode is optional. Filemode usually defaults to the current working directory. You can use the FTP subcommand LPWD to determine the current local working directory and PWD to determine the current remote working directory.

Some hosts do not have filetypes and when files from these systems are stored in CMS, filetype $DEFAULT is used.

The rules for naming files depends on the subcommand.

8.8.3.1 File names on the GET subcommand

The format for GET is:

GET foreignfile [localfile] (REPLACE

If a file exists on your local system with the same name as the file on the remote system, you can specify a local name or use the replace option.
**Foreign file**
- The name of the foreign file is required.
- We recommend that you do not specify `filemode`. You cannot specify a `filemode` other than the `filemode` of the current remote directory so there is no point in specifying it and we observed some interesting side effects when it was specified.
- You cannot use an asterisk (*) in the `filename` or `filetype`. You can use MGET if you want to use an asterisk.

**Local file**
- The name of a local file is optional.
- You can specify `filemode` to direct where the file should be stored. If `filemode` is not specified, the current local directory is used.
- You cannot use an asterisk (*) in the `filename`, `filetype`, or `filemode`.

### 8.8.3.2 File names on the PUT subcommand

The format for PUT is:

```
PUT localfile [foreignfile]
```

If a file exists on the remote system with the same name as foreignfile (or localfile if you don’t specify foreignfile), the file will be replaced.

**Local file**
- The name of a local file is required.
- You can specify `filemode` to indicate where the file can be found. If `filemode` is not specified, asterisk (*) is the default resulting in a search of all accessed disks.
- You cannot use an asterisk (*) in the `filename` or `filetype`. You can use MPUT if you want to use an asterisk.

**Foreign file**
- The name of a foreign file is optional.
- A `filemode` is accepted, including an asterisk (*), but it is ignored. The file is stored in the current directory on the foreign host.
- You cannot use an asterisk (*) in the `filename` or `filetype`.

### 8.8.3.3 File names on the MGET and MPUT subcommands

The format for MGET is:

```
MGET foreignfile (REPLACE
```

The format for MPUT is:

```
MPUT localfile
```

**File names**
- Multiple file names should be separated by a blank.
- You can use an asterisk (*) in the `filename` and/or `filetype` for pattern matching.
- You cannot specify a `filemode`. The current directories on both the local and remote systems are used.
8.9 Using FTP within an EXEC

FTP commands can be issued from an EXEC. FTP subcommands can be supplied by either the program stack or a CMS disk file.

Likewise, FTP commands results can be directed to either the console or to a CMS disk file.

There are two examples of how FTP subcommands could be issued from within a REXX exec in TCP/IP Function Level 320 User's Guide, SC24-5848.

8.9.1 Input

If the subcommand input is to be obtained from a disk file, that file must be identified with the CMS FILEDEF command. The ddname INPUT is recognized and used by FTP for subcommand input. Program stack data is ignored. You must provide the logon password on the same line as the user ID.

The program stack is used as the input source for FTP subcommands if the input device has not been defined with a FILEDEF, or if the input device has been defined as a terminal. The logon password must be queued with the logon name. You could specify the logon name and password in NETRC DATA instead of supplying them in the program. See 8.8.1.1, “NETRC DATA” on page 126 for more information.

8.9.2 Output

By default, output generated during FTP command processing is directed to your terminal. However, the output can be directed to a disk file. The file must be identified with the CMS FILEDEF command. Only the ddname OUTPUT is recognized and used by FTP.

8.10 Anonymous access

A popular feature of FTP is anonymous access. This allows public access to some file directories. The remote user uses the login name ANONYMOUS or ANONYMOU. No password is requested if anonymous access has been enabled for the server.

ANONYMOU does not have to be defined as a userid on your system unless your system uses an external security manager, in which case user ANONYMOU must be defined to it.

Also ANONYMOU must be enrolled in any SFS file pool where anonymous access is allowed. For example, if you want to allow anonymous access to an SFS directory where sample code resides, you might set up an SFS directory called SAMPLES in a filepool called TEST. Your SFS administrator would need to enroll ANONYMOU in the filepool:

```
ENROLL USER ANONYMOU TEST:
```

Either the SFS administrator or the owner of the samples directory would then grant ANONYMOU read authority to the directory and its files:

```
GRANT AUTHORITY TEST:SAMPLES. TO ANONYMOU (READ NEWREAD
GRANT AUTHORITY * * TEST:SAMPLES. TO ANONYMOU (READ
```
You might consider adding a banner that is displayed when the anonymous user logs on to give information about the system. Here is a sample banner:

```
*******************************************************************
** WELCOME to the FTP Center at the ITSO Center in Poughkeepsie **
**
** Your current directory contains the sample files.  **
**
** You can issue the command dir to see a list of the samples **
** available. Then you can issue get fn.ft fn.ft where **
** the first fn.ft represents the name on this system and the **
** second represents the name under which they should be stored **
** on your system.  **
*******************************************************************
```

See 8.2.1, “CHKIPADR EXEC” on page 117 for information on providing banners.

### 8.10.1 Configuration

In order to allow anonymous access to the FTP server, add the :ANONYMOUS tag to the server definition for FTPSERVE. The following shows the change we made to our SYSTEM DTCPARMS file.

```
.* File Transfer Protocol (FTP) daemon
:nick.FTPSERVE :type.server :class.ftp
:ESM_Enable.YES
:Parms.RDR A FTAUDIT
:Anonymous.YES
```
Chapter 9. Accessing VM file systems from other platforms

The Network File System (NFS) support in TCP/IP for VM/ESA Function Level 320 allows you to keep your data on VM but use your BFS, SFS or minidisk files transparently from any platform that has an NFS client.

With NFS, data is available across many platforms without duplication and you get all the advantages of keeping your data on VM:

- Automatic, secure facilities to back up and recover data
- System managed storage
- Host access control
- Centralized space administration

9.1 Concepts

The network file system (NFS) allows your system to access files on another computer as if they were on your own computer. NFS supports a hierarchical file structure. The directory and subdirectory structure can be different for individual client systems. NFS makes remote objects stored in file systems appear to be local, as if they reside in the local host. With NFS, all the systems in a network can share a single set of files, eliminating the need for duplicate copies of files on every system. NFS gives users and administrators the ability to distribute data across a network by exporting local file systems from a local server for access by remote clients and mounting remote server file systems over local client directories. NFS uses the Remote Procedure Call (RPC) protocol to communicate between the client and the server.

The VMNFS server implements the Network File System (NFS). The VMNFS server also implements the PCNFSD user ID authentication function.

VMNFS allows users to execute programs stored in remote file systems and to create and edit files. The VMNFS virtual machine requires access to the IBM Language Environment runtime library during execution.

In addition to file serving, the NFS server provides support for the PC-NFS protocol used by nearly all of the NFS clients available today.

For more information about NFS, see RFC 1094 and RFC 1813.

9.2 Enabling

NFS is an optional feature of VM/ESA Version 2 Release 4 that must be enabled before you use it. See Program Directory for TCP/IP Feature for VM/ESA and Features Function Level 320 Program Number 5654-030, GI10-4695, for information on enabling the NFS feature.

9.3 Configuration

The following is a description of the various files that can be used to configure NFS for your installation.
9.3.1 PROFILE TCPIP

The sample PROFILE TCPIP shipped with TCP/IP for VM/ESA Function Level 320 has an entry to autolog the NFS server, VMNFS, and reserves port 2049 for it. Verify that these statements are in the profile TCPIP on the configuration disk, TCPMAINT 198. The NFS server requires port 2049.

Our changes to PROFILE TCPIP

We did not need to make any changes to the PROFILE TCPIP for NFS.

9.3.2 DTCPARMS

The parameters in the DTCPARMS file that apply to the NFS server are:

- :Anonymous

  A tag value of YES indicates that a user can access the server without a VM user ID and password. The value in IBM DTCPARMS for Anonymous is NO. ANONYMOU does not have to be defined as a userid on your system.

  If your system uses an external security manager, then user ANONYMOU should be defined to it if you want an anonymous user to have access to resources controlled by the ESM.

  User ANONYMOU also must be enrolled in any SFS file pool where anonymous access is allowed. This is not necessary if PUBLIC is enrolled.

- :ESM_Enable

  Indicates whether an external security manager is being used on your system. The setting in IBM DTCPARMS is NO.

- :ESM_Validate

  If you set :ESM_Enable to YES and you are using RACF, this parameter is already properly set in IBM DTCPARMS. If you are using another external security manager, specify the name of the module to be used to validate user IDs and passwords or NO if validation is not done by your external security manager.

- :ESM_Racroute

  If you set :ESM_Enable to YES and you are using RACF, this parameter is already properly set in IBM DTCPARMS. If you are using another external security manager, specify the command to be used for initialization and termination of a RACROUTE environment or NO to indicate that no initialization of RACROUTE should be done.

- :Parms

  The following provides the tag values and their corresponding actions:

  V Force the server to accept only Version 2 requests.
  U Force the server to accept only UDP connections.
  M nnn Sets a trace mask. This is the preferred way to set up for debugging, rather than using the D or G parms.

Notes:

- The :Parms tag can also be used to specify Anonymous and RACF. We recommend, however, that the tags :Anonymous and :ESM_Enable be used instead of the :Parms tag.
You could use U or V if you wanted to force the server to use a lower level of support in order to debug a problem. Generally the level of support is negotiated by the client and server.

It is recommended that you specify M 504 as a permanent debug setting as it provides valuable information about why things are not acting as you might expect for SFS and BFS file access.

Tag values for the :Parms tag should be separated by blanks; for example, :Parms.M 504.

9.3.2.1 Our changes to the DTCPARMS file

We used the following server definition for the NFS server:

```
.* Network File System daemon
:nick.VMNFS :type.server :class.nfs
:ESM_Enable.YES
:Anonymous.YES
```

- :ESM_Enable.YES since our system was using RACF as the External Security Manager
- :Anonymous.YES so that users could access the server without specifying a VM user ID and password.

9.3.3 VMNFS CONFIG

This file contains configuration information for the NFS server. A sample file VMNFS SCONFIG is provided on TCPMAINT 591. You can copy it to TCPMAINT 198 and modify it.

The following parameters are available:

- **DUMPMOUNT**
  
  Controls whether the server will send a list of mounted file systems when requested by the NFS client. This would include SFS and BFS directories that are considered active (used within 15 minutes) and any minidisks the server has linked. The NFS client request for a list of mounted file systems is anonymous. Specify DUMPMOUNT NO if the names of mounted file systems should be hidden from those who are not authorized to use them. The default is YES.

- **EXPORT**
  
  Defines a symbolic name to be used for a mount and the corresponding mount operands. All the export statements form an export list which can be sent to an NFS client. There are no items in the list by default.

- **EXPORTONLY**
  
  Indicates whether clients may mount file systems not in the export list. The default is NO which means that clients have no restrictions on what can be mounted.

- **MAXTCPUSERS**
  
  Defines the maximum number of NFS clients using the TCP transport protocol that can be concurrently supported by the VM NFS server. The default is 50.

- **PCNFSD**
Indicates whether the user ID authentication protocol, PCNFSD, is available on the server. The default is YES.

- **VMfiletype**

  Defines how translation will be done for each file extension. If a client specifies lines=ext or trans=ext, the line end handling and translation are based on the file extension.

  The defaults are lines=CMS and trans=NO when no lines, trans or record options are specified on the mount.

### 9.3.3.1 Our changes to VMNFS CONFIG

We ran with a default VMNFS CONFIG file. However, we made use of the Export List to simplify mounting from the NFS client. See 9.9, “Mounting file systems using the export list” on page 143.

### 9.4 SMSG interface

You can control the NFS server from another VM user ID by using the CP SMSG command.

Using the SMSG interface you can:

- Detach a minidisk (usually one linked read-write) to establish your own link.
- Refresh a minidisk by deleting cached data.
- Write trace tables to the file TRACEV FILE A1.
- Refresh the VM NFS server configuration parameters.
- Request a summary of VM NFS server activity.
- Display the server level and configuration parameters.
- Execute a CMS command.

As shipped, all of the commands, except CMS, are available to all users. You can use the NFSSMSG and VMNFSSMG exits to restrict access.

Permission to pass a CMS command to the NFS server is controlled by VMNFSCMS EXEC. If this EXEC is not found on TCPMAINT 198, all CMS command requests are rejected. The output from the CMS command is written to the server's console.

Samples of NFSSMSG, VMNFSSMG, and VMNFSCMS are shipped on TCPMAINT 591 and can be copied to TCPMAINT 198 and modified.

We used the SMSG command when the NFS server had our 191 minidisk linked read-write because of an implicit mount (see 9.8, “Mounting file systems implicitly” on page 143). The following screen shows part of the response we received from query link. It verified that our 191 minidisk was linked read-write by the server. We then used the SMSG command to detach the minidisk.
9.5 Security

The NFS server can use an external security manager (ESM) to authenticate NFS clients and to control access to minidisks. See Appendix A in *TCP/IP Function Level 320 Planning and Customization*, SC24-5847 for a detailed discussion of security issues.

9.5.1 VM NFS Server Machine Authorization

The VM NFS server machine must have OPTION DIAG88 and privilege class B specified in the CP directory entry.

DIAG88 specifies that the virtual machine is authorized to use DIAGNOSE code X’88’ to validate user authorizations and link minidisks.

9.5.2 CMS Shared File System

In order for NFS clients to access files in the Shared File System (SFS), the NFS server must have SFS file pool administrator authority. List all servers that will provide such access on the ADMIN statement in all SFS server DMSPARMS files.

9.5.3 Byte File System

In order for NFS clients to access files and directories in the Byte File System (BFS), the NFS server must be defined as a POSIX superuser. The VMNFS CP directory must contain the following statements:

```
POSIXINFO UID 0 GID 0
POSIXOPT QUERYDB ALLOW
```

9.5.4 Additional security

NFS provides a number of exit routines which can assist you in providing additional control over the operation of the NFS server. The C, Assemble, and EXEC files are shipped on TCPMAINT 591. See *TCP/IP Function Level 320 Planning and Customization*, SC24-5847 for information on these exits.

```
18:06:53 * MSG FROM VMNFS : M 00000620 rw LSA004MR.191
smsg vmnfs M query link
18:06:53 * MSG FROM VMNFS : M 00000620 rw LSA004MR.191
smsg vmnfs M detach LSA004MR.191
18:34:41 * MSG FROM VMNFS : M DTCNFS1360I RC=0 from detaching LSA004MR
```
9.6 Operations


TCP/IP for VM/ESA Function Level 320 does not provide an NFS client. We used InterDrive 2.6, the NFS client for Windows 95/NT from FTP Software on a Windows NT platform. The following describes the steps we took to mount our VM system's minidisks, SFS files and BFS files from our VM system on our Window NT platform.

9.6.1 Defining the NFS servers on remote systems
On your NFS client, you identify the remote systems that contain the file systems you want to mount by specifying their fully qualified IP addresses. In our case, we had 2 host systems: TOTVM1.ITSO.IBM.COM and GDLVM7.ENDICOTT.IBM.COM. The following instructions describe how to add a host:

1. Double-click Network Neighborhood.
2. Double-click Entire Network.
3. Double-click InterDrive NT.
4. Select NFS Servers I Have Configured.
5. Click File.
7. Fill in your host name and click Add.
   
   NOTE: this step takes some time to complete. On our system, the cursor did not change to an hourglass but the window did not respond to further selections right away.

8. Click OK.

9.6.2 Creating aliases
We found it easiest to define an alias for the file systems that we wanted to mount. An alias is just a user-friendly name for the path. Some clients refer to aliases as sharenames or shares. To create an alias we did the following:

1. Double-click Network Neighborhood.
2. Double-click Entire Network.
3. Double-click InterDrive NT.
4. Double-click NFS Servers I Have Configured
5. Select the appropriate remote host.
7. Select Create Alias.
8. Fill in the alias name.
9. Fill in the Path specifying the path and the mount operands.

   We created the following aliases:
   - my191rw
This alias can be used to mount LSA004MR 191 when you want to work with text files that can be used from both CMS and Windows. The minidisk is mounted read-write.

Multiple write access to the same minidisk is not permitted. If a minidisk is already linked read-write by a user, the NFS server cannot mount it read-write.

- my191ro

This alias can be used to mount LSA004MR 191 read-only.

- totsfs

This alias can be used to mount the samples directory in filepool sfstest.

- totbfs

This alias can be used to mount the samples directory in filepool BFS.

Our system was RACF-protected. If your minidisk is protected by a link password, you can use the mdiskpw operand to specify the minidisk’s password.

10. Select the **Security tab**.

On this panel you can specify a VM user ID and password to be used when you request that the file system be mounted. See 9.6.3, “Specifying passwords”.

### 9.6.3 Specifying passwords

You can specify the user ID and password to use on the VM host. The user ID and password can be specified on the server definition (see 9.6.5, “Modifying the NFS Server definition” on page 142), the alias definition (see 9.6.2, “Creating aliases” on page 140 or 9.6.4, “Modifying aliases” on page 142), or both.

The user ID and password in the alias definition is used if provided. Otherwise, the user ID and password from the server definition is used. If no user ID or password is provided, you will be prompted.

We found it convenient to specify the user ID and password on the server definition. It eliminated the prompts for the information. By specifying the information on the server definition, we only had to change the password in one place when it changed. In our case, we always used the same user ID and password when we connected to a particular remote system. If you have multiple user IDs on a system or your password changes frequently, you may wish to provide the information each time you connect.

Be careful with password specification! It is easy to have your VM user ID revoked because of multiple attempts to connect to the system using the wrong password. The NFS server does multiple retries each time a connection fails so you can get into this situation quite easily even though you think you have only tried the connection a couple of times.
If the NFS server allows anonymous access (see 9.3.2, “DTCPARMS” on page 136), you do not need a user ID or password for the system where the file system resides. On our client we specified NOBODY as the user ID in the Connecting As box which appears when you request that the file system be mounted. We were not prompted for a password and the file system was mounted.

9.6.4 Modifying aliases

1. Double-click Network Neighborhood.
2. Double-click Entire Network.
3. Double-click InterDrive NT.
4. Double-click NFS Servers I Have Configured
5. Double-click the appropriate remote host.
6. Select the alias.
7. Select File.
8. Select Modify Alias.

You can then make your changes to the path. You can also select the Properties box and then the Security tab to provide or change user ID and password information (see 9.6.3, “Specifying passwords” on page 141).

9.6.5 Modifying the NFS Server definition

To modify a server definition, do the following:

1. Double-click Network Neighborhood.
2. Double-click Entire Network.
3. Double-click InterDrive NT.
4. Double-click on NFS Servers I Have Configured.
5. Select a remote system.
6. Click File.
7. Choose Properties.

Choosing the Security tab allowed us to specify the user ID and password to use when requesting a mount (see 9.6.3, “Specifying passwords” on page 141).

9.7 Mounting file systems

To mount one of the file systems, we did the following:

1. Double-click Network Neighborhood.
2. Double-click Entire Network.
3. Double-click InterDrive NT.
4. Double-click NFS Servers I Have Configured.
5. Double-click host that you want.
6. Select the alias you want to use.
7. Right mouse click and select Map Drive.
8. Fill in the user ID that should be used on the host system in the **Connecting As** box.

Even if you entered the user ID on the security panel when defining the alias, we recommend that you always specify the user ID here. If there is a check in the **Reconnect at Logon** box, the client will reconnect file systems that were mounted when you logged off the system next time you logon. During this reconnect processing, the client uses the user ID given in the **Connected As** box on the last connection. If there was no ID specified in the **Connected As** box on the last connection, it uses your Windows NT user ID. Since this may not be the same as your ID on the host system, the reconnect may fail.

9. Click on **OK**.

10. You may be prompted for a password if you did not specify one in the security section of the alias definition.

11. If successful, you will see the file system listed under **My Computer**.

**NOTE:** Error messages did not always present an accurate picture of the problem. Often we received an error message indicating that the password was incorrect but the real problem was that the mount operands on the alias definition were incorrect. It is best when you get any error to check your alias definition to make sure that the mount statement is correct.

---

### 9.8 Mounting file systems implicitly

There is a way to access the file system defined in an alias without actually mounting it. We used Windows NT Explorer and kept expanding the directories under Entire Neighborhood until we had a list of our aliases. If we left-mouse clicked on an alias, the files in that file system would be listed and we could work with them just as if they had been explicitly mounted.

The problem with this method is there is no way (that we found) to unmount it except for rebooting. If the alias mounts your 191 disk read-write, this can be a problem. We used the SMSG interface in order to unmount our 191 disk when we had the disk mounted read-write implicitly and were done with it. See 9.4, “SMSG interface” on page 138 for information on how to do this.

### 9.9 Mounting file systems using the export list

An installation can choose to pre-define aliases for its users. This is done by adding export statements to the VMNFS CONFIG file on TCPMAINT 198. There are variables that can be used in these statements that will be resolved by the NFS server. These variables are:

- `%USERID` the user ID specified for the mount is substituted.
- `%FSROOT` the FSROOT (root directory) parameter from the POSIXINFO CP directory statement is substituted.
- `%IWDIR` the IWDIR (working directory) parameter from the POSIXINFO CP directory statement is substituted.

We added the following statement to the VMNFS CONFIG file:

```
EXPORT /PC/Your191 %USERID.191,ro,record=nl,nlvalue=0D0A
```
We already had the host defined to the NFS client. We selected the host on Windows NT, right mouse clicked on it, and chose Explore. Information was exchanged between the NFS client and our NFS server and Your191 showed up in the list under the host. When we did a Map Directory for it, it prompted us for a user ID and password. The 191 disk of the user ID specified was mounted by the client.

9.9.1 Using the mounted file systems

Once the file system is mounted, you can work with the files just as if you were accessing them on the host. If you have write access to the files, any changes you make to the file are made to the host copy. Permission to read or write is the same as if on the host. The access permissions are based on the user ID used for the mount.
Chapter 10. Enterprise printing

The combination of TCP/IP for VM/ESA and RSCS provide facilities for enterprise printing in a TCP/IP environment. Enterprise printing can be defined as providing consolidated print services for all users in an enterprise on both host and network printers.

VM/ESA users can print on host-attached printers, network printers, printers attached to LAN servers and printers attached to workstations with print serving software. Workstation users can print on host-attached printers, network printers, printers attached to LAN servers and printers attached to workstations with print serving software using VM/ESA as an enterprise print server.

10.1 Print server consolidation

Print server consolidation using VM/ESA can simplify your print management.

10.1.1 Before consolidation

In Figure 58, a sample LAN environment has four network print servers, each controlling six cut sheet printers. There are a several workstations, some of which have attached printers. The cut sheet printers are depicted as logically attached to the print server although physically they attach directly to the network with their own network adapters. They each have TCP/IP-based Line Printer Daemon (LPD) support. Each print server has its own hard disk drive which holds the print requests until printing can be completed. The capacity on the hard drives must be monitored and managed individually. If additional printing capacity is needed, not only are new printers purchased, but additional servers may be needed as well. The operating systems of the workstation servers might be from different vendors. Those that are the same are often at different maintenance levels. Should a user need to print at a location different from his default, the results can be troublesome.

Like your environment perhaps, this sample configuration includes a S/390 with a high speed printer — but it is only being used for S/390 applications. Workstation users with 3270 sessions need to transfer (often large) files to the host before submitting them to the high speed printer. To print host files on their locally...
attached printer or even on one of the LAN printers, they first need to receive them onto their own workstation hard drive (assuming there is enough spare room) prior to selecting a destination and printing. Host files may also require conversion or reformatting.

10.1.2 The decision to consolidate

Realizing there are significant administration, support, operational, and user challenges in a purely distributed print server environment, you might search for a better solution. Certainly you will have noted the numerous press articles about the “drive to consolidation” which is underway in many organizations. Under all the hype is a basic truth that still applies: there are economies of scale. By leveraging your large systems to help with the management of print queues, you eliminate numerous points of administration and many management tasks.

10.1.3 Print server environment — after consolidation

As shown in Figure 59, after consolidation the network print servers are eliminated, the queue management function is conducted by VM/ESA on the mainframe, and the print queues themselves reside on large, reliable, S/390-attached disk. Any TCP/IP user, on a workstation or the host, anywhere in the enterprise can now print on any desired printer. All printers can be managed centrally by a single, reliable server: VM/ESA.

![Figure 59. Print serving after consolidation](image_url)

10.2 Concepts

Implementation of enterprise printing requires configuring of both TCP/IP for VM/ESA and RSCS. While TCP/IP does provide an LPD print server through the LPSERVE virtual machine, we recommend that the LPD server in RSCS be used instead as it provides more functionality, particularly in the area of PostScript and PCL printer support.

The LPD server provided by RSCS serves workstation requests to print a file. The RSCS server looks for the printer specified by the Line Printer Requestor (LPR) client and attempts to print the document. The printer itself is not dedicated to TCP/IP and it may receive output from other sources as well. The LPD server supports three types of printer connections:
• Local printers

Local printers are physically attached to the VM/ESA system. These are typically high capacity printers such as the IBM Infoprint 4000 Advanced Function Printing System.

• Remote Network Job Entry (NJE) printers

Any printer available through the NJE network can be accessed, with the remote RSCS printer support. This printer could be attached to any system in an NJE network (such as OS/390, VM/ESA or VSE/ESA) without a TCP/IP connection.

• Remote TCP/IP printers

Any printer controlled by another LPD server can be made available to a user through the LPR driver in RSCS. This includes printers which contain an internal LPD server, such as the IBM Network Printer family, printers attached to a workstation with an LPD server function and printers attached to a network interface, such as the Hewlett-Packard JetDirect external print server. This support is provided by the RSCS LPR driver. RSCS will redirect inbound LPR requests to the desired printer, end users do not need to know the IP address or host name and queue name of the real printer.

VM/ESA users have three options for printing to TCP/IP printers. A detailed comparison of these may be found at:


Briefly, the available printing options can be described as follows:

• Printing synchronously using the LPR command. The LPR command, running in the CMS user’s virtual machine, connects directly to the TCP/IP printer’s LPD server. Using this option ties up the CMS user’s virtual machine for the duration of the time it takes the TCP/IP printer to process the request.

• Printing asynchronously using the LPR command. The LPR command sends the print file to RSCS for printing. This option is described further in 10.7, “Printing from VM/ESA to a network printer” on page 168.

• Printing asynchronously using the print PostScript (PPS) command to send the print file to RSCS for printing. PPS is a sample command provided with RSCS for which more information may be found at:


The LPR driver provided by RSCS sends workstation print files received by the RSCS LPD driver and host print files to remote TCP/IP printers. You have two non-exclusive options in how you can set up the LPR driver. One option uses a pool of links for PostScript printers (the LPRP link pool) and a pool of printers for non-PostScript printers (the LPR link pool). The other option uses dedicated links for each printer. With the first option a form identifier is used in conjunction with a set of control files to direct a file to a specific printer; with the second option the form identifier is only used to specify printer control information (such as the separator). We will describe both options in this document.

The primary advantage to the option with defined links is that you can define a pool of printers to which RSCS can direct output. We describe how we used this capability later in this chapter.
10.2.1 Required documentation

Before attempting to implement enterprise printing on VM/ESA, obtain the current publications and related material. Some of this material is not available in printed form and must be downloaded from the Internet. Table 10 on page 148 lists the material that may be required and gives Internet sites and other sources for the information.

Table 10. Publications which may be required for enterprise printing

<table>
<thead>
<tr>
<th>Title</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Hardcopy available from IBM.</td>
</tr>
<tr>
<td></td>
<td>Softcopy only.</td>
</tr>
<tr>
<td></td>
<td>Softcopy only.</td>
</tr>
<tr>
<td></td>
<td>Hardcopy available from IBM.</td>
</tr>
<tr>
<td></td>
<td>Hardcopy available from IBM.</td>
</tr>
<tr>
<td>Technical Reference Bundle, by Hewlett-Packard, Inc. part number 5021-0377</td>
<td>Hewlett-Packard, Inc.</td>
</tr>
</tbody>
</table>

While all PCL and PostScript implementations have much in common, every printer vendor has differences in implementation. It is very important that you have available the appropriate PCL and/or PostScript reference publications for your printers. The publication PCL5e/PostScript Technical Reference, S544-5344 provides information on differences in PCL and PostScript implementations for IBM Network Printers and IBM InfoPrint LAN printers. The PostScript Language Reference, by Adobe Systems, Inc. ISBN 0-201-37922-8 provides the definition of the PostScript language and the Technical Reference Bundle, by Hewlett-Packard, Inc. part number 5021-0377 provides the definition of the PCL language.

A short reference to PCL Commands, Basic Page Formatting, and Font Selection is available on the Hewlett-Packard Web site at:
http://www.interactive.hp.com/cgi-bin/cspt/ljxxxx/dyndocwrap.pl?lid=general&fid=bpl02705

10.3 Security

There are no security services provided in the IP printing environment. If required, an installation could provide control over which printers a user could direct print to through the use of RSCS exits.
10.4 Configuring Windows

In this section we describe how to configure Windows to receive print from VM/ESA for printing on a workstation-attached printer and how to configure Windows to send print output to a printer managed by VM/ESA.

10.4.1 Configuring Windows to receive print from VM/ESA

If you wish to print to a printer that is attached directly to a Windows workstation from the VM/ESA host or from another workstation, you will need to set up an LPD on that workstation. Windows NT 4 does include an LPD server as a service called “TCP/IP Print Server”. However, we found that this was not very robust. Windows 95/98 do not include an LPD server.

There are many third-party LPD servers available for the Windows platforms and we chose NIPrint from Network Instruments, Inc. for this residency. More information and a demonstration version of NIPrint can be found at the Network Instruments Web site at:

http://www.netinst.com/

NIPrint also provides an LPR function; however, it does not operate as a port monitor. We therefore recommend other solutions, which are described in 10.4.2, “Configuring Windows NT 4 as a print client” on page 151 and 10.4.3, “Configuring Windows 95/98 as a print client” on page 153.

10.4.1.1 NIPrint configuration for local print

Once you have installed NIPrint, using the instructions on the Network Instruments Web site, perform the following steps to configure NIPrint as a local print server.

1. Click the Start menu, select Programs, select NIPrint, then NIPrint32.
2. On the NIPrint32 window, select Configuration from the menu, then select General Settings from the pull-down list.
3. On the General Settings window, shown in Figure 60 on page 150, select ENABLE LOCAL PRINT SERVER and Confirm on close, then click Ok.
4. On the NIPrint32 window select **Configuration**, then **Local Print Configuration** in the pull down list.

5. On the LOCAL Print Configuration window select **Add Printer**.

6. On the Add/Edit LOCAL Printer window, shown in Figure 61, enter a **Printer Name** that you wish the printer to be known as on the network and select the printer in the **Installed Printers** list. The printer name you enter should contain no blanks or special characters.
7. Click **Advanced Print Settings**. On the LOCAL Print Advanced Settings window click the **Wait for remote LPR to close connection** radio button, then click **Ok**.

8. On the Add/Edit LOCAL Printer window, click **Ok**.

9. On the LOCAL Print Configuration window, click **Save**.

You should now be able to use the local printer you have defined from any LPR client, including RSCS on VM/ESA, by specifying the domain name or IP address of the Windows workstation as the LPD server and the printer name (IBM4029 in this example) as the printer or queue name to LPR.

**10.4.2 Configuring Windows NT 4 as a print client**

Windows NT 4 includes an LPR port monitor, which provides a TCP/IP print monitor port driver for seamless Windows NT network printing. This port driver can be used on Windows NT to automatically direct print jobs from Windows NT to any printer or print server that runs a TCP/IP LPD print server, such as VM/ESA.

1. Click **Start->Settings->Printers**.

2. Double-click **Add Printer**. The Add Printer wizard leads you through the installation process of setting up and configuring a printer.

3. Select **My Computer**, then click **Next**.

4. If the port you are going to print to is listed on under **Available ports**, select the check box beside it, otherwise click **Add Port**.

5. In the list of **Available Printer Ports**, select **LPR Port** and click **New Port**.
6. On the Add LPR compatible printer window, shown in Figure 64, enter the domain name or IP address of the VM/ESA system that provides the print services and the name of the printer queue as defined in LPDXMANY CONFIG (see 10.6.2, “Setting up the LPD server in RSCS” on page 157).

Windows NT also includes an LPR command for printing jobs directly. The LPR command provides standard RFC 1179 interfaces. Figure 65 on page 153 shows the options available on the LPR command. When using the LPR command to send a job to a printer managed by VM/ESA, the -s option specifies the host name or host IP address of the VM/ESA system and the -P option specifies the printer queue name as specified in the LPDXMANY CONFIG file. The -C and -J options can be used to specify the class and jobname to VM/ESA. The -O option is generally ignored, but you should not specify it as the results are unpredictable. You should not specify the -d option as the output is also unpredictable. For more information on the options supported by VM/ESA, see Chapter 9 in VM RSCS Exit Customization, SH24-5222. The options are case sensitive with the Windows NT LPR command.
An example of using the LPR command is:

```
  lpr -Stotvm1.itso.ibm.com -Pnp24ps output.ps
```

where “totvm1.itso.ibm.com” is the server and “np24ps” is the name of the print queue.

### 10.4.3 Configuring Windows 95/98 as a print client

Windows 95/98 does not include a LPR port monitor. However, IBM does provide an LPR port monitor for Windows 95/98, known as the **IBM LPR Remote Printing Client**, at no charge. This IBM LPR Remote Printing Client is restricted by its license to use with IBM network printers.

The IBM LPR Remote Printing Client is available from the IBM Printing Systems Web site at:

http://www.ibm.com/printers/

On the IBM Printing Systems Web site select **Support** in the menu on the left side; on the next page select **Fixes, printer drivers and updates** on the left side and then select **Network Printers Utilities and Microcode**. Follow the instructions on this Web page under the heading IBM LPR Remote Printing Client Version 2.30 for Windows 95 to retrieve the IBM LPR Remote Printing Client.

For printers other than IBM network printers, the University of Texas at Austin provides the **ACITS LPR Remote Printing Windows 95/98 and NT 4.0** product, with equivalent function. Additional information about this product is available at:

http://www.utexas.edu/academic/otl/software/lpr/

There are also many other products available for Windows and other platforms which provide LPR function.

The IBM LPR Remote Printing Client utility provides a TCP/IP print monitor port driver for seamless Window 95 network printing. This port driver can be used on Windows 95/98 to direct print jobs from Windows 95/98 to the IBM network printers or any other printer or printer server that runs a TCP/IP LPD print server. The port driver can be used on mobile systems for remote printing or it can be used on department print servers in an unattended print server mode.
This package also includes a command line LPR program for printing files directly. The IBM LPR program, called IBMLPR, provides standard RFC 1179 interfaces and allows the -S and -P parameters to be set by way of environment variables so that drag-and-drop printing can be enabled for a default printer.

Figure 66 on page 154 shows the options available on the IBMLPR command. When using the LPR command to send a job to a printer managed by VM/ESA, the -S option specifies the host name or host IP address of the VM/ESA system and the -P option specifies the printer queue name as specified in the LPDXMANY CONFIG file. The -C and -J options can be used to set the class and jobname to VM/ESA. The -O option is generally ignored, but you should not specify it as the results are unpredictable. You should not specify the -D option as the output is also unpredictable. For more information on the options supported by VM/ESA, see Chapter 9 in VM RSCS Exit Customization, SH24-5222. The options are case sensitive with the Windows NT LPR command.

An example of using the IBMLPR command is:

```
ibmlpr -Stotvm1.itso.ibm.com -Pnp24ps output.ps
```

where “totvm1.itso.ibm.com” is the server and “np24ps” is the name of the print queue.

---

Usage: ibmlpr [-S <server>] [-P <printer>] [-C <class>]
[-J <job name>] [-L <option>] [-O <filter>]
[-R <remote port>] [-T <timeout>] [-D] [-B]
[-#<copies>] filename1 filename2 ... filename(###)

Options:

S -- Print server or printer hostname or ip address
(The environment variable LPRSERVER may be used instead)
P -- Printer or printer queue name
(The environment variable LPRPRINTER may be used instead)
C -- Class name on banner page
J -- Job name on banner page
L -- Option to use for local port -- [S]trict, [D]efault, [E]xtended
(The default option is [D])
O -- Filter option: Ascii [f], HPGL, PCL, Postscript, etc... [l]
The default option is [l]
R -- Remote port to attempt connection on (default is 515)
T -- Timeout value (default is 120 seconds, 0 = INFINITE)
D -- Send data file before control file
B -- Disable the banner control flag (L) By default this flag is sent.
# -- Number of copies (1 to 10) to print (default is 1 copy)
filename(s) -- file(s) to print - may include wildcards (*,'?')

---

10.4.3.1 Installation of the IBM LPR Remote Printing Client

Double-click the program named INSTLPR.EXE to begin installing the program. After the installation process has completed, your system is ready to use the IBM LPR Remote Printing Client. Refer to the online documentation located in the IBM
LPR Remote Printing Client group for information about configuring and using an IBM LPR Remote Printer.

10.4.3.2 Adding an IBM LPR Remote Printer
1. Click **Start**, select **Settings**, then select **Printers**.
2. Double-click **Add Printer**. The Add Printer wizard leads you through the installation process of setting up and configuring a printer.
3. Add a Local printer and select the print driver for the IBM LPR Remote Printer you are installing.
4. Select LPT1: for the printer port or an IBM LPR Remote Printer if you have already created a remote printer.
5. Type a name for the printer. The name can be anything you find convenient to remember.
6. Click the **NO** radio button when prompted about **Printing a Test Page**, then click **Finish** to install any required print drivers and create the printer.
7. Right click the newly created printer icon in the **Printers Folder** and select **Properties**.
8. Click the **Details** menu tab, then the **Add Port** push button. Click the **Other** radio button and select **IBM LPR Remote Printing Client**.
9. Fill out the printer information. If you are unsure about some of the settings, consult your network administrator. Click **OK** when you are finished. You may create up to 32 different IBM LPR printer entries: IBMLPR_1 through IBMLPR_32.
10. Click **OK** to return to the **Properties** dialog and then click **OK** to save your printer settings. The printer is now configured for the IBM LPR Remote Printing Client.

**Note:** In order for the IBM LPR Remote Printing Client to work correctly, the spooler settings for the associated printer must be set to the following values:

- Start printing after last page is spooled
- Disable bidirectional support for this printer

The IBM LPR Remote Printing Client sets these values correctly when you send the first file to the printer. You can change the spooler settings manually as follows:

1. Right click the **Printer** icon.
2. Click **Properties**.
3. Click **Details**.
4. Select **Spooler settings**.

**Note:** On some Windows systems, you may need to set the spooler data format to RAW in order to print correctly. If you have a problem when you first try to print, try setting this value to RAW as described above.
10.5 Configuration of TCP/IP for VM/ESA for network printing

In the PROFILE TCPIP on TCPMAINT's 198 disk, comment out the LPSERVE statement in the AUTOLOG section:

; LPSERVE PASSWORD ; LP Server

In the PROFILE TCPIP on TCPMAINT's 198 disk, change the statement for port 515 in the PORT section from LPSERVE to RSCS:

; 515 TCP LPSERVE ; LP Server
515 TCP RSCS ; LP Server

Changes were made to RSCS support of LPR printing as a result of this residency. The changes are documented in APAR PQ29745 and the fix is available as PTF UQ33885 for TCP/IP for VM/ESA Function Level 320. We strongly suggest that users install the PTF for this APAR and any others related to LPR available for TCP/IP.

10.6 Configuration of RSCS for network printing

There are several steps to take in configuring RSCS for IP printing. A new service virtual machine (RSCSDNS) must be defined, changes must be made to the RSCS CONFIG file, the PROFILE GCS for RSCS must be updated, and new configuration files (LPDXMANY CONFIG and LPRXFORM CONFIG) must be created.

Issue the following to make sure the RSCS server is running at the 9803 (or higher) RSU service level:

SMSG RSCS QUERY SYSTEM LEVEL

A response from RSCS indicating it is at the appropriate level would be:

RSCS Networking Version 3, Release 2.0-9803

If RSCS is not at or beyond this level, we recommend that you install the latest RSU before proceeding with these instructions, otherwise some steps may not produce the desired results otherwise.

Several changes were made to RSCS support of LPR printing as a result of this residency. The changes are documented in APAR VM62277 and the fix is available as PTF UV60455. We strongly suggest that users install the PTF for this APAR and any others related to LPR or LPD available for RSCS.

10.6.1 Setting up the RSCSDNS server

The RSCSDNS server should already be defined for you if you are installing VM/ESA Version 2 Release 4. As delivered, this server will be autologged for you automatically at GCS startup. If you do not have the RSCSDNS server in your directory, you will need to define one as shown in Figure 67 on page 157.
Figure 67. RSCSDNS server virtual machine directory entry

On the RSCSDNS virtual machine’s 191 disk you will need to have a PROFILE EXEC as shown in Figure 68.

```plaintext
/* RSCSDNS Profile Exec */
'ACCESS 403 B' /* access mdisk with GETHOSTC module */
'ACCESS 592 C' /* access tcpmaint disk */
'GLOBAL LOADLIB SCEERUN' /* SCEERUN is part of RUNTIME LIBRARY */
'GETHOSTC' /* invoke the Domain Name Server code */
```

Figure 68. PROFILE EXEC for RSCSDNS

10.6.2 Setting up the LPD server in RSCS

1. Change RSCS’s PROFILE GCS so it can load the LPD-type link exit routine. On the RSCS install ID’s (P684096K) 401 minidisk (also the RSCS ID’s 191 minidisk), locate the line:

   ```plaintext
   GLOBAL LOADLIB RSCS
   ```

   and change it (if required) to:

   ```plaintext
   GLOBAL LOADLIB RSCS RSCSEXIT
   ```

2. Change RSCS’s PROFILE GCS so it can load a configuration file for the LPD-type link exit routine. First locate a line containing a FILEDEF statement, then add the following line:

   ```plaintext
   FILEDEF LPD DISK LPDXMANY CONFIG
   ```

3. Copy the LPDXMANY CONFSAMP file from the RSCS install ID’s 406 minidisk to its 401 minidisk as LPDXMANY CONFIG.

4. Define the LPD-type links in the RSCS CONFIG file (located again on the install ID’s 401 minidisk) by adding the lines in Figure 69 on page 158. We have defined four links; you may need to define more to ensure that a link is always available to your workstation users. We have also increased the default timeout from 60 seconds to 300 seconds to allow for network congestion.
5. Define printer queue names within the LPD configuration file (LPDXMANY CONFIG). We have defined four printers in our configuration and a default printer definition. Two of the printers (HP890C and NP17PS) will use the LPR/LPRP pool and two will use groups of dedicated printers (NP24PS and POK3130).

Table 11 contains more information on each of the tokens in the LPDXMANY CONFIG file.

<table>
<thead>
<tr>
<th>Token</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A printer queue name up to 32 characters. A queue name of DEFAULT can be used to define parameters for any printer queue not defined by a configuration file record. When a printer queue name arrives, LPDXMANY first looks for a configuration record of that name; if not found it will look for a configuration record using DEFAULT; if not found it will use existing LPDXMANY defaults. We recommend that the printer queue name not exceed 8 characters (see token 5).</td>
</tr>
<tr>
<td>2</td>
<td>The logical record length 1-1280; default is 255. We recommend the maximum setting of 1280.</td>
</tr>
<tr>
<td>3</td>
<td>The number of lines per page 1-99; default is 66. We recommend that this be specified as &quot;*&quot; which gives the default.</td>
</tr>
<tr>
<td>4</td>
<td>The one character spool file class; default is blank. We recommend that this be specified as &quot;*&quot; which gives the default.</td>
</tr>
<tr>
<td>Token</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| 5     | The one- to eight-character spool file form; default is blank.  
This name is used to associate a printer with an entry in the LPRXFORM CONFIG file by the RSCS LPR function.  
If you are using the LPR/LPRP pool option, we recommend that this value be the same as the printer queue name in token 1.  
If you are using dedicated links for your printers, this value should be set to a form which defines the desired print processing. |
| 6     | The one- to eight-character job name; default is SYSTEM.  
We recommend that this be specified as "**" which gives the default. |
| 7     | The one- to eight-character PSF destination; default is blank. When using a PSF destination, LPDXMANY will set the user ID field of the tag to SYSTEM. When using a PSF destination, the printer queue name can be in the form SYSTEM@nodeid. Alternatively, the printer queue name can be anything unique as long as the user ID field is SYSTEM or * and the node ID field is specified.  
We recommend that this be specified as "**" for non-PSF printers. |
| 8     | Either PAGE or NOPAGE; default is NOPAGE. This parameter determines how pagination will be performed as follows:  
PAGE will cause LPDXMANY to always paginate regardless of the control file print filter.  
NOPAGE will cause LPDXMANY to paginate only as defined by the 'f' or 'p' control file print filters. To be effective, the control file must be received prior to the data file.  
We recommend that this be specified as "**" which gives the default. |
| 9     | Either TRAN, NOTRAN, or ASISCC; default is TRAN. Translation specifies whether or not to translate the data file prior to spooling as follows:  
TRAN: Translate data received into EBCDIC removing any CR/LF/FF (carriage return, line feed, form feed) control characters.  
NOTRAN : Do not translate data received into EBCDIC, but remove any CR/LF/FF control characters.  
ASISCC: Leave the data received as is, do not translate nor remove any control characters. ASISCC would be desired when data received is destined to be printed by a LPR-type link.  
We recommend that “ASISCC” be specified if the final destination is a workstation or network printer. |
| 10    | The one- to eight-character user ID to which the file should be spooled; default is derived from the printer queue name. **Note:** If a PSF destination is provided (see token 7), then the userid field can only be set to SYSTEM.  
We recommend that “ASCII” be specified if the final destination is a workstation or network printer. |
| 11    | The one- to eight-character destination node ID to which the file should be spooled; default is derived from the printer queue name.  
If you are using the LPR/LPRP pool option we recommend that this be specified as “LPRP” for PostScript printers and “LPR” for all other workstation or network printers. If you are using dedicated links for your printers, this option should be specified as "**". |
6. Start the LPD-type links in RSCS’s PROFILE GCS by adding the following lines after the ‘RSCS INIT’ statement (one for each link defined in Figure 69 on page 158):

   START LPD1
   START LPD2
   START LPD3
   START LPD4

7. Next time RSCS is recycled (IPLed), the LPD links will be available for use. Note that RSCS can be quickly recycled by entering the command:

   RSCS SHUTDOWN QUICK CP IPL GCS

   from the RSCS console.

10.6.3 Setting up the LPR function in RSCS

1. Change RSCS’s PROFILE GCS so it can load the LPD-type link exit routine. On the RSCS install ID’s (P684096K) 401 minidisk (also the RSCS ID’s 191 minidisk), locate the line:

   GLOBAL LOADLIB RSCS

   and change it (if required) to:

   GLOBAL LOADLIB RSCS RSCSEXIT

2. Change RSCS’s PROFILE GCS so it can load the configuration files for the LPR-type link exit routine. First locate a line containing a FILEDEF statement, then add the following lines:

   FILEDEF LPR DISK LPR CONFIG
   FILEDEF LPRP DISK LPRP CONFIG
   FILEDEF LPRXFORM DISK LPRXFORM CONFIG

3. Define the LPRXFORM exit in the RSCS CONFIG file (located on the install ID’s 401 minidisk) by adding the following lines.

   EXIT 0 ON LPRXFM00
   EXIT 1 ON LPRXFM01

4. Create an LPRXFORM CONFIG file on the install ID’s 401 minidisk. There should be an entry for each form specified in the LPDXMANY CONFIG file. Figure 71 shows the LPRXFORM CONFIG lines for a shared Hewlett-Packard DeskJet 890C (attached to a Windows NT workstation); Figure 72 shows the LPRXFORM CONFIG lines which can be used by any IBM Network Printer 24s which meet a standard configuration and are LAN attached; Figure 73 shows the LPRXFORM CONFIG lines which can be used by an IBM Network Printer 17 (attached to a Windows workstation); Figure 74 on page 161 shows the LPRXFORM CONFIG lines which can be used by an IBM 3130 printer which meets a standard configuration. We would also suggest a definition of an entry for a STANDARD form to be used for printing EBCDIC files from VM/ESA. In

<table>
<thead>
<tr>
<th>Token</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>The filename of a TCP/IP TCPXLBIN translate table to be used for this printer queue name. The file must exist on any accessed disk. If not specified, the translation table used will either be the default one within LPDXMANY or the override specified within the LPDXMANY configuration file. We recommend that this be specified as &quot;.*&quot; which gives the default.</td>
</tr>
</tbody>
</table>
our case we used the NPPS entry with the specification of PostScript output
(FORMSUB=P+ASCII) removed (Figure 75 on page 162).

For more information on the content of the LPRXFORM CONFIG file see
10.6.4, “Setting up the LPRXFORM CONFIG file” on page 163.

Figure 71. LPRXFORM CONFIG for HP DeskJet HP890C

| FORM=HP890C |
| HOSTNAME=TOT48.ITSO.IBM.COM |
| PRINTER=HP890C |
| SEP=YES |
| PREFIX=1B45 |
| PREFIX=1B28733050 |
| PREFIX=1B2873313248 |
| PREFIX=FF04 |
| PREFIX=1B45 |
| SUFFIX=1B45 |

Figure 72. LPRXFORM CONFIG for IBM Network Printer 24 printing PostScript files

| FORM=NPPS |
| FORMSUB=P+ASCII |
| SEP=YES |
| PREFIX=252150532D41646F62650D0A |
| PREFIX=737461745736469637420626567696E20666616C73652073657464675706C |
| PREFIX=65786D6F646520322073657464675706C55 |
| PREFIX=786D6F646520332073657464675706C55 |
| PREFIX=FF04 |

Figure 73. LPRXFORM CONFIG for an IBM Network Printer 17

| FORM=IPPS |
| FORMSUB=P+ASCII |
| SEP=YES |
| PREFIX=252150532D41646F62650D0A |
| PREFIX=737461745736469637420626567696E20666616C73652073657464675706C |
| PREFIX=65786D6F646520322073657464675706C55 |
| PREFIX=786D6F646520332073657464675706C55 |

Figure 74. LPRXFORM CONFIG for an IBM 3130 for use in printing PostScript files

| FORM=IPPS |
| FORMSUB=P+ASCII |
| SEP=YES |
| PREFIX=252150532D41646F62650D0A |
| PREFIX=737461745736469637420626567696E20666616C73652073657464675706C |
| PREFIX=65786D6F646520322073657464675706C55 |
| PREFIX=786D6F646520332073657464675706C55 |
Figure 75. LPRXFORM CONFIG for IBM Network Printer printing non-PostScript files

5. Define the LPR-type links in the RSCS CONFIG file (located on the install ID’s 401 minidisk). The LINK and PARM statements are used to define eight PostScript (Figure 78 on page 163) and eight non-PostScript (Figure 79 on page 163) LPR links that are pooled on the ROUTE GROUP statements for improved performance. As shown here, on each of these links a separator page will be printed for each print request. This can be suppressed in LPRXFORM CONFIG file for specific printers. Figure 76 shows the definitions for a group of three IBM Network Printer 24s; Figure 77 on page 162 shows the definitions for a group of three IBM 3130 printers. The printers in these two groups can either be addressed directly by the linkid or by the groupid. RSCS will direct print output to any one of the printers in the group when the groupid is used. This allows the user to set up a pool of like printers for use in a printer room, for example.

```
FORM=STANDARD
SEP=YES
PREFIX=252150532D41646F62650D0A
PREFIX=7374617475736469637420626567696E20661C2D0D0A
PREFIX=65786D6F646520322073657470617065727461792065640D0A
PREFIX=F04
PREFIX=252150532D41646F62650D0A
PREFIX=7374617475736469637420626567696E2074727565207365746475706C65
PREFIX=786D6F646520320D0D0A
```

Figure 76. RSCS CONFIG changes for a group of IBM Network Printer 24s

```
LINKDEFINE NP24PSA TYPE LPR FORM * AST
LINKDEFINE NP24PSB TYPE LPR FORM * AST
LINKDEFINE NP24PSC TYPE LPR FORM * AST
PARM NP24PSA EXIT=LPRXPSE ITO=0 US=Y SYS=Y EP='S=Y EH=N C=LPRP HOST=9.12.2.5 PR=PASS
PARM NP24PSB EXIT=LPRXPSE ITO=0 US=Y SYS=Y EP='S=Y EH=N C=LPRP HOST=9.12.2.6 PR=PASS
PARM NP24PSC EXIT=LPRXPSE ITO=0 US=Y SYS=Y EP='S=Y EH=N C=LPRP HOST=9.12.2.8 PR=PASS
ROUTE GROUP NP24PS TO LINK NP24PSA NP24PSB NP24PSC
```

Figure 77. RSCS CONFIG changes for a group of IBM 3130 printers

```
LINKDEFINE C16 TYPE LPR FORM * AST
LINKDEFINE POK3130D TYPE LPR FORM * AST
LINKDEFINE POK3130E TYPE LPR FORM * AST
PARM C16 EXIT=LPRXPSE ITO=0 US=Y SYS=Y EP='S=Y EH=N C=LPRP HOST=9.12.2.2 PR=afccu2
PARM POK3130D EXIT=LPRXPSE ITO=0 US=Y SYS=Y EP='S=Y EH=N C=LPRP HOST=9.12.2.3 PR=afccu2
PARM POK3130E EXIT=LPRXPSE ITO=0 US=Y SYS=Y EP='S=Y EH=N C=LPRP HOST=9.12.2.4 PR=afccu2
ROUTE GROUP POK3130 TO LINK C16 POK3130D POK3130E
```
10.6.4 Setting up the LPRXFORM CONFIG file

The structure of the LPRXFORM CONFIG file is documented in *VM RSCS Operation and Use*, SH24-5220. While the basic structure is quite simple, the specification of the PREFIX and SUFFIX tag values is very dependent on the
specific printer being defined. There should be no blank lines in the file: use a comment line (a line starting with "") to separate printer definitions.

Table 12. LPRXFORM CONFIG description

<table>
<thead>
<tr>
<th>Tag name</th>
<th>Tag description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORM</td>
<td>This value is used to relate this printer definition to an entry in the LPDXMANY CONFIG file and is also used by the VM/ESA LPR command to specify a printer. The FORM parameter on the VM/ESA LPR command correlates with the name specified here. If you will be supporting workstation LPR client printing via VM, this value should match the fifth token of an entry in LPDXMANY CONFIG.</td>
</tr>
<tr>
<td>HOSTNAME</td>
<td>This value is used to specify the fully qualified host name of either a workstation running an LPD or a network printer, and is mutually exclusive with the HOSTID keyword.</td>
</tr>
<tr>
<td>HOSTID</td>
<td>This value is used to specify the host IP address of either a workstation running an LPD or a network printer, and is mutually exclusive with the HOSTNAME keyword. If HOSTNAME and HOSTID are both specified, the HOSTID is used.</td>
</tr>
<tr>
<td>PRINTER</td>
<td>This is either the name of the printer on the LPD server or the queue name in a network printer. The queue name in a network printer is printer-dependent and should be provided in the documentation that came with the printer.</td>
</tr>
<tr>
<td>SEP</td>
<td>A value of YES indicates a VM/ESA separator page; NO indicates no VM/ESA separator page; 2P indicates 2 separator pages (for duplex printing). In general, we suggest a value of YES for shared printers and a value of NO for personal workstation attached printers.</td>
</tr>
<tr>
<td>FORMSUB</td>
<td>For PostScript printers, specify this as P+ASCII, which indicates to the LPRXPSE exit that the file is ASCII PostScript data. For all other printers, do not specify this tag.</td>
</tr>
<tr>
<td>PREFIX</td>
<td>ASCII control sequences (in hex) that are sent to the printer before the print data. If there is prefix data that should be sent after the separator page, a prefix string of &quot;FF04&quot; should be used to indicate the separator page location. &quot;FF04&quot; is recognized by the LPR exits which insert the separator page at the specified location. If more than one PREFIX tag is present, they are concatenated.</td>
</tr>
<tr>
<td>SUFFIX</td>
<td>ASCII control sequences (in hex) that are sent to the printer after the print data. If more than one SUFFIX tag is present, their values are concatenated.</td>
</tr>
</tbody>
</table>

10.6.4.1 PCL prefix and suffix strings

All PCL controls must be preceded by the Escape (esc) character (hex 1B). Each set of PCL controls should be preceded by a reset character “E” (hex 1B45). The prefix and suffix strings in Figure 71 on page 161 for the Hewlett-Packard DeskJet 890C are explained in Table 13 on page 164. The prefix strings will result in a monospace font at 12 characters per inch for the separator page.

Table 13. PCL prefix and suffix strings for HP890C

<table>
<thead>
<tr>
<th>ASCII representation</th>
<th>Text representation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1B45</td>
<td>escE</td>
<td>Reset</td>
</tr>
<tr>
<td>1B28733050</td>
<td>esc(s0P)</td>
<td>Monospace font</td>
</tr>
</tbody>
</table>
The sample exec in Figure 80 will convert PCL commands in a file to strings for use on PREFIX and SUFFIX tags in LPRXFORM CONFIG. The exec adds the escape sequence (esc above) to the beginning of each line. The separator string of FF04 is inserted by having an input line containing the string “separator”.

```c
/* PCPREFIX - Change PCL strings to ASCII HEX for LPRXFORM */

Arg ifn ift ifm . /* Get arguments */
If ifm = "" then ifm = "*" /* Filenmode default to "*" */
If ift = "" then ift = "PCL" /* Filetype default to "PCL" */
If ifn = "" then Exit 4 /* No filename, exit! */

'PIPE state' ifn ift ifm , /* Get file info */
  'PIPE (endchar ?) <' ifile , /* Read input */
  'strip', /* Remove blanks */
  'console', /* Display on console */
  'a: casei nlocate /separator/ ', /* Find separators */
  'xlate from 037', /* Translate to CP 500 */
  'xlate e2a', /* Translate to ASCII */
  'specs l-* c2x 1', /* Convert to hex */
  'specs /1B/ 1 l-* n', /* Prefix with escape x1B */
  'deblock fixed 60', /* Split to shorter records */
  'b: faninany', /* Join streams together */
  'buffer',/*
  '?> ofile ', /* Save in "fn" PCPREFIX A */
  'literal PCPREFIX: Output to' ofile , /* Build header */
  'console', /* Display on console */
  '? a:', /* Process separator */
  'specs /FF04/ 1', /* Insert "hex" separator */
  'b:' /* Go to join streams */
```

*Figure 80. PCPREFIX exec to convert PCL strings for LPRXFORM CONFIG*

The following is an example of use of the PCPREFIX exec.
PostScript control sequences are in reverse Polish notation. Between the start sequence of “statusdict begin” and the end sequence “end”, the control information is in the format “value keyword”. The sample exec in Figure 81 on page 167 will convert PostScript commands in a file to strings for use on PREFIX and SUFFIX tags in LPRXFORM CONFIG. The exec adds the carriage return and line feed (CRLF above) to the end of each line. The separator string of FF04 is inserted by having an input line containing the string “separator”.

---

### Table 14. PostScript prefix and suffix strings for IBM Network Printer 24

<table>
<thead>
<tr>
<th>ASCII representation</th>
<th>Text representation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2512102165 62656F64652030207365747061727472617920656E640D0A</td>
<td>statusdict begin false setduplexmode 0 setpapertray endCRLF</td>
<td>Paper tray 0, duplex off</td>
</tr>
<tr>
<td>737461747573646963742062656F64652031207365747061727472617920656E640D0A</td>
<td>statusdict begin true setduplexmode 1 setpapertray endCRLF</td>
<td>Paper tray 1, duplex on</td>
</tr>
<tr>
<td>FF04</td>
<td></td>
<td>Separator placement</td>
</tr>
</tbody>
</table>
Figure 81. PSPREFIX exec to convert PostScript strings for LPRXFORM CONFIG

The following is an example of the use of the PSPREFIX exec.

```bash
psprefix np24ps
PSPREFIX: Input from NP24PS PS A1
%!PS-Adobe
statusdict begin false setduplexmode 0 setpapertray end
separador
statusdict begin true setduplexmode 1 setpapertray end
PSPREFIX: Output to NP24PS PSPREFIX A
252150532D41646F62650D0A
7374617475736469637420626567696E2066616C7365207365746475706C65786D6F646520302073657470617065727472617920656E640D0A
FF04
7374617475736469637420626567696E2074727565207365746475706C65786D6F646520312073657470617065727472617920656E640D0A
```
10.7 Printing from VM/ESA to a network printer

Printing from VM/ESA CMS to a network printer uses the LPRSET and LPR commands to send the print file to the network printer through RSCS and TCP/IP. A detailed discussion of the LPRSET and LPR commands is available in Chapter 10 of the *TCP/IP Function Level 320 User's Guide*, SC24-5848.

The LPRSET command can be used to set defaults for further processing by the LPR command. We recommend that permanent defaults be set to simplify end user printing. These permanent defaults could be set in SYSPROF EXEC, if you wish to set site level defaults.

Initially the default values are as shown in the following screen.

```
lprset (query
Printer and host values have not been defined
Server is defined as: RSCS (Default)
Printer is connected on link: LPR(P) (Default)
Printer is connected on node: TOTVM1 (Default)
LPR mode: SYNCHRONOUS (Default)
Ready; T=0.03/0.03 11:31:41
```

If you will be printing files through the RSCS LPR support, then this LPRSET command could be used:

```
LPRSET (PERM ASYNCH
```

This will result in the following defaults:

```
lprset (perm asynch
Ready; T=0.03/0.04 11:32:10
lprset (query
Printer and host values have not been defined
Server is defined as: RSCS (Default)
Printer is connected on link: LPR(P) (Default)
Printer is connected on node: TOTVM1 (Default)
LPR mode: ASYNCHRONOUS
Ready; T=0.03/0.03 11:32:12
```

With these defaults set, you could print an EBCDIC file to our pool of IBM Network Printers using one of the following commands:

```
PPS filename filetype (PRINTER NP24PS FORM NP
PPS filename filetype (PRINTER NP24PS
LPR filename filetype (RSCS NP24PS,TOTVM1 FORM NP
LPR filename filetype (RSCS NP24PS,TOTVM1
```

Using the LPR command, you must specify the RSCS node of the system on which the printer is defined (in our case TOTVM1). Form NP is defined in LPRXFORM CONFIG identical to form STANDARD.

With the above defaults set, you could print an ASCII PostScript file to our pool of IBM Network printers using one of the following commands:

```
PPS filename filetype (PRINTER NP24PS FORM NPPS
LPR filename filetype (RSCS ASCII,NP24PS FORM NPPS
LPR filename filetype (RSCS NP24PS,TOTVM1 FORM NPPS
```
The ASCII PostScript file may have been created on VM/ESA using the Document Composition Facility (SCRIPT) or may have been created on a workstation and uploaded to VM/ESA.

It is not possible to print an ASCII file to a PostScript printer using RSCS as the LPRXPSE exit assumes a text file is in EBCDIC and translates it. The installation could create a duplicate definition of the PostScript printer as a PCL printer and print the ASCII file to the PCL printer.

Printing to a PCL printer is similar to PostScript, but can be simpler. If you have a PCL printer, such as the Hewlett-Packard 890C we used in our environment, and you have a FORM set up for that printer in LPRXFORM CONFIG, you can use the following commands. To print an EBCDIC file use this command:

```
LPR filename filetype (FORM HP890C
```

And to print an ASCII file use this command:

```
LPR filename filetype (FORM HP890C RSCS ASCII,LPR
```

In general, if a printer is explicitly defined in LPRXFORM CONFIG, you use the FORM parameter on LPR or PPS to direct your output to that printer. If your printer is explicitly defined in RSCS CONFIG, then the entry in LPRXFORM CONFIG is just used to specify control information and you would use specify the printer name using the PRINTER parameter on the PPS command or as a value on the RSCS parameter of the LPR command.

---

### 10.8 Printing from Windows to a network printer

Once you have completed the steps in 10.4.2, “Configuring Windows NT 4 as a print client” on page 151 or 10.4.3, “Configuring Windows 95/98 as a print client” on page 153, then printing is quite simple. From a Windows application simply select as your printer destination the printer you defined on a port in Windows. The print output will be routed to the VM/ESA system through the LPD driver in RSCS and then to the selected printer through an LPR driver.

For printing from a command prompt, enter the LPR command (or equivalent) with a `-s` parameter of the VM/ESA system domain name and a `-p` parameter of the printer you defined on a port in Windows.

### 10.9 Printing from Windows to a VM/ESA printer

While we have not described in this chapter how to print from Windows workstation to a VM/ESA host printer, this is quite simple to accomplish for ASCII files to non-AFP printers or AFP files to AFP printers.

For ASCII files, define the host printer in the LPDXMANY CONFIG file, specifying that the file be translated into EBCDIC. You should also specify class, form, and node ID if the printer is on a remote VM/ESA system.

For AFP files (using the IBM AFP driver for Windows), define the host printer in the LPDXMANY CONFIG file and specify the class, form, PSF destination and node if the printer is on a remote VM/ESA system.
10.9.1 IBM AFP printer drivers for Windows

IBM has made available printer drivers for the various Windows platforms which can be used to create an AFP data stream file.

These IBM AFP Printer Drivers are available from the IBM Printing Systems Web site at:

http://www.ibm.com/printers/

On the IBM Printing Systems Web site select Support in the menu on the left side, on the next page select Fixes, printer drivers and updates on the left side, and then select IBM AFP printer drivers for Windows. Follow the instructions on this Web page for the appropriate Windows version to download and install the driver.

10.10 Printing from UNIX workstation

Printing from a UNIX workstation is very dependent on the specific version involved. UNIX variants have commands such as QPRT, ENQ, LP and LPR. Refer to your operating system documentation for more information.
Chapter 11. Application daemons and solutions

This chapter discusses the Service Daemons functions. These service machines run as application servers (daemons), providing shared services to clients over a network.

11.1 Dynamic Host Configuration Protocol Daemon (DHCP)

The DHCP daemon (DHCPD server) responds to client requests for boot information using information contained in a DHCP machine file. This information file includes the IP address of the client, the IP address of the TFTP daemon and information about the files to request from the TFTP daemon.

Unlike the BOOTP daemon, each client that is serviced by the DHCP daemon does not need to be listed in a machine file. IP addresses may be allocated in three different ways:

- Dynamically
  - An IP address is temporarily assigned to the client for a specified period of time, called a lease. The lease on the address must be renewed prior to the end of the specified lease time if the client wishes to keep using the IP address.

- Automatically
  - An IP address is permanently assigned to a client from the list of available addresses.

- Manually
  - A specific IP address (specified in the machine file) is assigned to the client.

11.1.1 Configuring the DHCP server

To configure the DHCPD virtual machine, you must perform the following steps:

- Update the TCP/IP server configuration file.
- Update the DTCPARMS file for DHCPD.
- Configure the ETC DHCPTAB file.
- Update PROFILE TCPIP.
- Include DHCPD in the AUTOLOG statement of the TCP/IP server configuration file. The DHCPD virtual machine is automatically started when TCP/IP is initialized. The DHCPD server listens on port 67.

For a detailed discussion about DHCPD configuration refer to Chapter 10 of TCP/IP Function Level 320 Planning and Customization, SC24-5847.

11.1.2 Allocating a new network address

This section describes the client/server interaction if the client does not know its network address. Assume that the DHCP server has a block of network addresses from which it can satisfy requests for new addresses. Each server also
maintains a database of allocated addresses and leases in permanent local storage.

The following is a description of the DHCP client/server interaction steps illustrated in Figure 82.

1. The client broadcasts a DHCPDISCOVER message on its local physical subnet. At this point, the client is in the INIT state. The DHCPDISCOVER message may include some options such as network address suggestion or lease duration.

2. Each server may respond with a DHCPOFFER message that includes an available network address and other configuration options. The servers may record the address as offered to the client to prevent the same address being offered to other clients in the event of further DHCPDISCOVER messages being received before the first client has completed its configuration.

3. The client receives one or more DHCPOFFER messages from one or more servers. The client chooses one based on the configuration parameters offered and broadcasts a DHCPREQUEST message that includes the server identifier option to indicate which message it has selected.

   In the case no offers are received, the client may reuse that address if its lease is still valid.

4. The servers receive the DHCPREQUEST broadcast from the client. Those servers not selected by the DHCPREQUEST message use the message as notification that the client has declined their server's offer. The server selected in the DHCPREQUEST message commits the binding for the client to persistent storage and responds with a DHCPACK message containing the...
configuration parameters for the requesting client. The combination of client hardware and assigned network address constitutes a unique identifier for the client's lease and is used by both the client and server to identify a lease referred to in any DHCP messages. The “Your IP Address” field in the DHCPACK messages is filled in with the selected network address.

5. The client receives the DHCPACK message with configuration parameters. The client performs a final check on the parameters, for example with ARP for the allocated network address, and notes the duration of the lease and the lease identification cookie specified in the DHCPACK message. At this point, the client is configured.

If the client detects a problem with the parameters in the DHCPACK message, it sends a DHCPDECLINE message to the server and restarts the configuration process. The client should wait before restarting the configuration process to avoid excessive network traffic in case of looping. On receipt of a DHCPDECLINE, the server must mark the offered address as unavailable (and possibly inform the system administrator that there is a configuration problem).

**Note:** If the client receives a DHCPNAK message, the client restarts the configuration process.

6. The client may choose to relinquish its lease on a network address by sending a DHCPRELEASE message to the server. The client identifies the lease to be released by including its network address and its hardware address.

**Note:** Responses from the DHCP server to the DHCP client may be either broadcast or unicast, depending on whether the client is able to receive a unicast message before the TCP/IP stack is fully configured; this varies among TCP/IP implementations.

### 11.1.3 Reusing a previously allocated network address

If the client remembers and wishes to reuse a previously allocated network address, the following steps are carried out:

1. The client broadcasts a DHCPREQUEST message on its local subnet. The DHCPREQUEST message includes the client's network address.

2. A server with knowledge of the client's configuration parameters responds with a DHCPACK message to the client (provided the lease is still current), renewing the lease at the same time.

3. If the client's lease has expired, the server with knowledge of the client, responds with DHCPNACK.

4. The client receives the DHCPACK message with configuration parameters. The client performs a final check on the parameters and notes the duration of the lease and the lease identification cookie specified in the DHCPACK message. At this point, the client is configured and its T1 and T2 timers are reset.

If the client detects a problem with the parameters in the DHCPACK message, the client sends a DHCPDECLINE message to the server and restarts the configuration process by requesting a new network address. If the client receives a DHCPNAK it must request a new address by restarting the configuration process as described in 11.1.2, “Allocating a new network address” on page 171.
Note: A host should use DHCP to verify its IP address and network parameters whenever the local network parameters have changed (for example, at system boot time), or the local network configuration may change without the host's or user's knowledge. If a client has multiple IP interfaces, each of them must be configured by DHCP separately.

For further information refer to RFC 2131 - Dynamic Host Configuration Protocol and to RFC 2132 - DHCP Options and BOOTP Vendor Extensions. The method for accessing RFCs is described in Appendix C, “RFCs related to protocol specifications” on page 203.

Additional DHCP Considerations are described in Chapter 7 of TCP/IP Tutorial and Technical Overview, GG24-3376.

11.2 Bootstrap Protocol daemon (BOOTPD)

The BOOTP virtual machine (daemon) responds to client requests for boot information using information contained in a BOOTP machine file. This information includes the IP address of the client, the IP address of the TFTP daemon and information about the files to request from the TFTP daemon.

For example, the IBM Network Station establishes a communication session with the BOOTP server. The system code will be loaded into the random access memory of the Network Station. The download is done using trivial file transfer protocol (TFTP). For further discussion on TFTP refer to 11.3.1, “Configuring the TFTP Server” on page 175.

11.2.1 Configuring the BOOTP server

To configure the BOOTP virtual machine, you must perform the following steps:

- Update the TCP/IP server configuration file.
- Update the DTCPARMS file for BOOTP.
- Configure the ETC BOOTPTAB file.

Include BOOTP in the AUTOLOG statement of the TCP/IP server configuration file. The BOOTP virtual machine will then be automatically started when TCP/IP is initialized. The BOOTP server listens on port 67.

For a detailed discussion about BOOTP configuration refer to Chapter 9 of TCP/IP Function Level 320 Planning and Customization, SC24-5847.

11.2.2 Operation of BOOTP

- The client defines its own hardware address; this is normally in a Read Only Memory.

A BOOTP client sends the hardware address in a UDP datagram to the server. If the client knows his IP address and/or the address of the server, it will use them, but in general BOOTP clients have no IP configuration data at all. If the client does not know its own IP address, it uses 0.0.0.0. If the client does not know the server's IP address, it uses the limited broadcast address (255.255.255.255). The UDP port number is 67.
The server receives the default datagram and looks up the hardware address of the client in his configuration file, which contains the client’s IP address. The server fills in the remaining fields in the UDP datagram and returns it to the client using UDP port 68. One of three methods may be used to do this:

- If the client knows its own IP address (it was included in the BOOTP request), then the server returns the datagram directly to this address. It is likely that the Address Resolution Protocol (ARP) cache in the server’s protocol stack will not know the hardware address matching the IP address.

- If the client does not know its own IP address (it was 0.0.0.0 in the BOOTP request), then the server must concern itself with its own ARP cache. ARP on the server cannot be used to find the hardware address of the client because the client does not know its IP address and so cannot reply to an ARP request. This is called the “chicken and egg” problem. There are two possible solutions:
  - If the server has a mechanism for directly updating its own ARP cache without using ARP itself, it does so and then sends the datagram directly.
  - If the server cannot update its own ARP cache, it must send a broadcast reply. When the BOOTP client receives the reply, it will record its own IP address (allowing it to respond to ARP requests) and begin the bootstrap process.

For further information refer to RFC 951. The method for accessing RFCs is described in Appendix C, “RFCs related to protocol specifications” on page 203.

**Note:** ARP is used to dynamically bind an IP address to a hardware address. ARP is implemented on a single physical network and is limited to networks that support broadcast addressing.

### 11.3 Trivial File Transfer Protocol daemon (TFTPD)

The TFTPD daemon (TFTPD server) transfers files between the BFS and TFTP clients. TFTPD supports access to files maintained in a BFS directory structure that is mounted.

The operating system code for the IBM Network Station will be loaded into the random access memory of the Network Station after a communication session with the BOOTP server is established.

TFTPD reads files into memory (cache) before they are sent to a client. Similarly, it receives a complete file before that file is written to the BFS. In addition, the permanent file list (TFTPD PERMLIST) identifies files that should be kept in memory. Due to the fact that files are maintained in memory, you may need to increase the virtual machine size to accommodate the number and size of files that your machine handles.

#### 11.3.1 Configuring the TFTPD Server

To configure the TFTPD virtual machine, perform the following steps:

- Update the TCP/IP server configuration file.
Update the DTCPARMS file for TFTPD.
- Review and address additional configuration considerations.
- Create the TFTPD PERMLIST data file.
- Create the TFTPD USERLIST data file.
- Update PROFILE TCPIP file.

Include TFTPD in the AUTOLOG statement of the TCP/IP server configuration file. The TFTPD virtual machine is then started automatically when TCP/IP is initialized. By default, the TFTPD server listens on port 69; however, this default can be changed through additional TFTPD customization.

For a detailed discussion about TFTP Server Configuration refer to Chapter 23 of *TCP/IP Function Level 320 Planning and Customization*, SC24-5847.

11.4 Domain Name Server

The Domain Name Server (DNS) maps a host name to an internet address or an internet address to a host name. DNS is like a telephone book that contains a person's name, address, and telephone number. For each host, the name server can maintain internet addresses, nicknames, mailing information, and available well-known services (for example, DNS, SMTP, FTP, or Telnet).

When a client needs to communicate with another host, the client uses either the internet address of the remote host or sends a query to the DNS for host name resolution.

11.4.1 Configuring the Domain Name Server

The TCP/IP name server is configured using the configuration file, NSMAIN DATA. A sample copy of this file is shipped with TCP/IP Function Level 320 as NSMAIN SDATA on the TCPMAINT 591 disk. If needed, this file should be copied to TCPMAINT 198, customized, and renamed to match what is specified in the NSMAIN command (default name is NSMAIN DATA).

The NSMAIN DATA configuration file contains information for the DNS virtual machine. This file needs to be configured only if this server will be used on your system. It allows you to specify:
- Internet addresses
- Nicknames for fully qualified domain names
- Mailing information
- Trace options

An example of the NSMAIN DATA configuration file is shown in E.2.1, “NSMAIN DATA” on page 216.

You should include NAMESRV in the AUTOLOG list as a virtual machine. The NAMESRV virtual machine is then started automatically when TCPIP is invoked. The name server requires that ports TCP 53 and UDP 53 are reserved for the server.
You can also specify unique startup parameters and customize the server environment by modifying the DTCPARMS file.

**Note:** You should modify the DTCPARMS file for the NAMESRV server if you:
- Use a configuration file other than NSMAIN DATA.
- Identify a DB2 for VM database if NAMESRV will not be running in a caching-only mode.

If you want to set up a name server that does not require DB2, you can configure the domain name server as a caching-only name server. The caching-only name server uses a CMS file to define the host names and internet addresses of remote name servers. For more information about caching-only configuration statements, see “Configuring the Domain Name Server” in Chapter 8 of *TCP/IP Function Level 320 Planning and Customization*, SC24-5847.

### 11.4.2 Operation

The Domain Name Server (NAMESRV) serves to resolve domain names and aliases for clients, and provides commands for querying name servers from a CMS user ID. Name servers manage two kinds of data which can be stored in either local data files (HOSTS LOCAL) or in DB2™ Server for VSE and VM (formerly known as SQL/DS™) databases. The first kind of data is held in sets called zones; each zone is the complete database. The second kind of data is cached data that is acquired by a local resolver. However, before the name server can resolve a query, it must be supplied with resource records that either define a zone or identify a different (remote) name server that can provide the information required for a response. Resource records are the data entries that were loaded into the SQL table after the database was set up. For information about Resource Record format, see “Configuring the Domain Name Server” in Chapter 8 of *TCP/IP Function Level 320 Planning and Customization*, SC24-5847.

If a name server maintains local data and has authority for the zone defined by this data, it is called a primary name server. If a name server zone transfers a zone from a remote primary name server, it is called a secondary name server. Once a secondary name server receives zone data, it has authority for that zone. The secondary name server then periodically refreshes this zone data based on values defined in the zone’s authority record. For more information about primary and secondary zones, see RFCs 1034 and 1035.

A name server that does not have authority for any zone is called a caching-only name server. To respond to queries, a caching-only name server must communicate with a remote name server in the network that has access to zone data. The TCP/IP domain name server can be configured to run as a primary, secondary, or caching-only server. Both primary and secondary name servers store zone data in DB2 tables.

For a detailed description of how to prepare and define the DB2 for VM Database, refer to Chapter 8 of *TCP/IP Function Level 320 Planning and Customization*, SC24-5847.

The name server has three main locations for retrieving data: cache memory, a DB2 for VM database, or other name servers.
The name server caches query answers in one of its four caches. All future queries are answered directly from the cache without referencing the DB2 for VM database or another name server. Records remain active in the cache, based on their time-to-live (TTL) fields. If a cache becomes full, the least-recently-used entry is deleted. For more information, see Chapter 8 of *TCP/IP Function Level 320 Planning and Customization*, SC24-5847.

If the name server has authority for the query’s zone, the name server consults its DB2 for VM database for the answer. This answer is then sent back to the requestor and cached for future queries.

The name server communicates with other name servers to query records outside its zone, to answer queries about zones for which it has authority, and to transfer zones both to and from other name servers.

Figure 83 illustrates the domain name resolution process.

*Figure 83. VM domain name resolution*
11.4.3 Resolvers

Programs that query a name server are called resolvers. Because many TCP/IP applications need to query the name server, a set of routines is usually provided for application programmers to perform queries. Under VM, these routines are available in the TCP/IP application programming interface (API) for each supported language.

Resolvers operate by sending query packets to a name server, either over the network or to the local name server. A query packet contains the following fields:

- A domain name
- A query type
- A query class


TCP/IP applications have a resolver routine compiled into their code that resolves host names into internet addresses. This code accesses a name server or site tables, or both, depending on your configuration. You can change the configuration in the TCPIP DATA file. For information about the TCPIP DATA file, see Chapter 5 of TCP/IP Function Level 320 Planning and Customization, SC24-5847.

11.5 Routing daemon (RouteD)

RouteD is a server that implements the Routing Information Protocol (RIP). It provides an alternative to static TCP/IP gateway definitions. The VM/ESA host running with RouteD becomes an active RIP router in a TCP/IP network. With TCP/IP for VM/ESA Function Level 320, RIP Version 2 (RFC 1723), an extension of RIP Version 1 (RFC 1058), is supported. RIP Version 2 provides improved configurability and diagnostic capabilities and Virtual IP Addressing (VIPA). For more information about RIP, see RFCs 1058 and 1723.

The RouteD server dynamically creates and maintains the network routing tables using RIP, which eliminates the need to manually configure and maintain a routing table. RIP defines how gateways and routers periodically broadcast their routing tables to adjacent nodes. Handling of subnets that have the same IP address is now available. Network topologies that consist of two subnets with the same IP subnet address for different internal networks cause routing ambiguity. The RouteD server recognize this topology and dynamically changes the routing tables entries.

For example, the RouteD server can determine if a new route was created, if a route is temporarily unavailable, or if a more efficient route exists for a given destination.

For a detailed description about static and dynamic routing refer to 2.10, “TCP/IP routing protocols and techniques” on page 43.

11.5.1 Configuration process

The steps to configure RouteD are as follows:
1. Create the RouteD configuration file.
2. Update PORT, BSDROUTINGPARMS, and GATEWAY statements in the TCP/IP profile.
3. Update the DTCPARMS file (Optional).
4. Configure the ETC GATEWAYS file (Optional).

   Note: If a default route is to be defined to a destination gateway or router, configure a default route in this ETC GATEWAYS file.

For a detailed description of the configuration process refer to Chapter 7 in TCP/IP Function Level 320 Planning and Customization, SC24-5847.

11.5.2 RIP routes
TCP/IP routes that are added and maintained by RouteD are RIP routes. Only RouteD can add or delete RIP routes. Furthermore, if RouteD is terminated, RIP routes are not deleted. To help maintain the integrity of the routing table, at initialization, RouteD deletes all RIP routes from the stack routing table and then adds RIP routes based on BSDROUTINGPARMS settings and the optional ETC GATEWAYS file. For detailed information about these settings refer to “Configuration Process” in Chapter 7, TCP/IP Function Level 320 Planning and Customization, SC24-5847.

11.5.2.1 RouteD RIP route types
RouteD supports four types of RIP routes:

- Passive

   Passive RIP routes are known by both TCP/IP and RouteD. Information about passive routes is put in TCP/IP's and RouteD's routing tables. A passive entry in RouteD's routing table is used as a place holder to prevent a route from being propagated and from being overwritten by a competing RIP route. With the exception of directly-connected passive routes, passive routes are not propagated; they are known only by this router. Passive routes should be used when adding routes where the target host or network is not running RIP. Passive routes should also be used when adding a default route, because this is the only way to prevent a route from timing out.

- Permanent

   Permanent routes are like passive routes except that they are propagated to other networks. This may be necessary in certain configurations, such as when a destination cannot propagate its own routing information. However, because it is possible to create routing loops and because recoverability is poor, permanent routes should be used with care. Passive routes should be used instead of permanent routes, whenever possible.

- External

   External RIP routes are known by RouteD but not by TCP/IP. External routes are managed by other protocols such as the Exterior Gateway Protocol (EGP). An external entry exists in RouteD's routing tables as a place holder to prevent a route from being overwritten by a competing RIP route. External routes are not propagated. RouteD does not manage external routes. Therefore, RouteD only knows that there is an existing route to a host or network and one that is not known to TCP/IP. External routes should be used when the local host is running with some type of non-RIP routing protocol that dynamically changes
the TCP/IP routing tables. The foreign host does not need to run any routing protocol, since the only concern is how to route traffic from the local host to the foreign host, and how to prevent multiple routing protocols from interfering with each other.

- **Active**

An active RIP route is treated as a network interface. Active gateways are routers that are running RIP but are reached by a medium that does not allow broadcasting and is not point-to-point. RouteD normally requires that routers be reachable via broadcast addresses or via a point-to-point link. If the interface is neither, then an active gateway entry can add the gateway to RouteD’s interface list. RouteD will treat the active gateway as a network interface. Note that the active gateway must be directly connected.

Active routes should be used when the foreign router is reachable over a non-broadcast, non-point-to-point network and is directly connected to the local host. RouteD will communicate with active routers using point-to-point transmissions to the gateway address.

### 11.5.3 Virtual IP addressing (VIPA) functions

VIPA provides the following functions:

- Fault-tolerant TCP connections providing automatic and transparent recovery from controller and interface failure
- Support for defining primary and alternate interfaces to the same (sub)net and support to switch dynamically between these interfaces in case one of them malfunctions
- Recovery from VM/ESA host failure where an alternate host has the necessary redundancy to back up a failing VM host
- Routing improvements using the customized interface metrics, route forwarding options and VIPA

For a detailed description of VIPA refer to 2.5.1.4, “Virtual IP addressing (VIPA)” on page 18.

#### 11.5.3.1 Purpose

The purpose of virtual IP addressing is to free other TCP/IP hosts from dependence on particular network attachments to VM/ESA. Prior to VIPA, other hosts were bound to one VM/ESA HOME IP address and, therefore, to a particular network attachment to VM/ESA. If the physical network attachment failed, the IP address became unreachable. Recovery required network reconfiguration or the ability of a client to retry alternate destination addresses that had been returned by the name server. Not all commercial applications attempt recovery by trying every address returned by a name server; most just try the first address.

VIPA provides a virtual network attachment with a virtual IP address that other TCP/IP hosts can use to select a TCP/IP stack without selecting a specific network attachment. If a specific physical network interface fails, the VIPA interface should still be reachable by another physical interface.
11.6 VSNMP virtual machine

Simple Network Management Protocol (SNMP) is an architecture that allows you to manage an internet environment. You can use SNMP to manage elements such as gateways, routers, and hosts on the network. Network elements act as servers and contain management agents, which perform the management functions required. Network management stations act as clients to run the management applications that monitor the network. SNMP provides the communication between these elements and stations to send and receive information about an internet's resources. The SNMP agent runs in the SNMPD virtual machine.

The Management Information Base (MIB) consists of information about these resources and elements. The MIB is referred to as a database, but it actually exists as counters and temporary storage areas on most of the servers. Data in the MIB is designed according to international standards for internet management and defined according to the International Standard Organization Abstract Syntax Notation 1 (ASN.1). For more information about the MIB, refer to RFC 1155. The method for accessing RFCs is described in Appendix C, “RFCs related to protocol specifications” on page 203.

More functions are available by using the SNMP daemon Distributed Program Interface (DPI). The DPI permits end users to dynamically add, delete, or replace management variables in the MIB without requiring the recompile of the local SNMP agent. For more information about the SNMP DPI, see TCP/IP Function Level 320 Programmer's Reference, SC24-5849.

You can manage TCP/IP for VM with NetView or any management product that supports SNMP, including TME 10 NetView for AIX® and equivalent vendor programs.

11.6.1 VM implementation of SNMP with NetView

A full SNMP implementation in TCP/IP for VM/ESA Function Level 320 requires the following components:

- TCP/IP stack
  Sends the Protocol Data Unit (PDU) between the TCP/IP hosts.
- Netview
  Implements the SNMP command interface and the monitor function.
- SNMP Query Engine (SNMPQE virtual machine)
  SNMPQE - Connects Netview SNMP clients to TCP/IP.
- SNMPD
  Implements the SNMP agent.

The SNMP support in NetView consists of a command processor and some sample CLISTs. A CLIST, or the NetView operator, may invoke the SNMP command in order to obtain an answer from the addressed remote SNMP agent. Information that can be obtained from SNMP agents is defined by the Management Information Base (MIB). The MIB defines objects, such as packet counts and routing tables, which are relevant to a TCP/IP environment. The objects defined by the MIB are divided into groups, with each group representing
a set of management data. For a detailed description of how to define the MIB data file, refer to “Defining the MIB Data File”, in Chapter 22, *TCP/IP Function Level 320 Planning and Customization*, SC24-5847.

The following figure illustrates the VM implementation of SNMP with Netview.

![Figure 84. Overview of NetView SNMP support](image)

The following steps describe how the SNMP command is processed:
1. The NetView operator or CLIST issues an SNMP command.
2. The SNMP command is validated by the SNMP command processor.
3. The command processor passes the request to the SNMPIUCV task.
4. The SNMPIUCV task passes the request to the SNMP Query Engine.
5. The SNMP Query Engine validates the request, converts the MIB variable to ASN.1 format if necessary, builds the SNMP request and sends it to the SNMP daemon.
6. The SNMP Query Engine receives a response from the SNMP daemon.
7. The SNMP Query Engine decodes the response and sends it to the NetView SNMPIUCV task.
8. The SNMPIUCV task sends the response as a multi-line message to the requesting operator or authorized receiver.

### 11.6.2 Configuration of SNMP servers

For detailed information about the configuration of SNMPD virtual machine (agent) and the SNMQE virtual machine, refer to Chapter 22, *TCP/IP Function Level 320 Planning and Customization*, SC24-5847.
11.7 Universal File Transfer daemon (UFTD)

The Universal File Transfer (UFT) server receives data streams from a TCP/IP UFT client for delivery to a network destination or printer, acting as a gateway for processing files from the TCP/IP world to the NJE world through RSCS.

The UFTD-type link driver will only handle a single data stream at a time. Since the driver was designed to allow multiple listens on the same TCP/IP port number, more than one UFTD-type link driver may be started to handle multiple concurrent incoming streams.

The UFTD-type link driver will bind to well-known port 608 and listen for TCP/IP UFT clients to connect and send a data stream. The port number that the UFTD-type link driver listens on may be overridden by defining a different port number on the UFTD-type link PARM statement or when starting a UFTD-type link. This allows for site installation testing and more flexibility. The maximum data length supported when receiving any individual file by the UFTD-type link driver is 2,147,483,647 bytes (2 gigabytes).

This link driver is an input-only driver. No data or messages will be transferred out over this link. Any files or messages queued to this link will remain so until explicitly transferred or purged.

You can use UFTD exits to specify where files will be delivered within the NJE network. The UFTD exits are called at six points in the operational cycle of the UFTD-type link drivers:

- Command processing
- Connect processing
- Data processing
- End-of-file processing
- Initialization
- Termination

RSCS provides sample UFTD exit routines, which you can use or modify as needed, for the UFTD-type links defined in your network.

11.7.1 Sending files to a TCP/IP UFT server

IBM supplies a sample exit routine called UFTXOUT to send data on a UFT-type link to a UFT daemon in a TCP/IP network. Use the CMS SENDFILE command with the asynchronous UFT option to send files to a UFT daemon in a TCP/IP network. For more information about UFTXOUT, see VM RSCS Exit Customization, SH24-5222. For more information about the CMS SENDFILE command, see VM/ESA: CMS Command Reference, SC24-5676.

11.7.2 Receiving files from a TCP/IP UFT client

IBM supplies a sample exit routine called UFTXIN to receive data on a UFTD-type link from a UFT client in a TCP/IP network. For more information about UFTXIN, see VM RSCS Exit Customization, SH24-5222.
11.7.3 Configuring the UFTserver

Chapter 24 of *TCP/IP Function Level 320 Planning and Customization*, SC24-5847 describes how to configure the Universal File Transfer daemon.

11.8 REXEC

The REXECD virtual machine implements the Remote Execution protocol (RExec) and the Remote Shell protocols (RSH) daemons. It will receive the REXEC requests on port 513 and the RSH requests on port 514.

11.8.1 Operation

The Remote Execution virtual machine sends a single command to a remote system for execution. The remote system name, user ID, password and command string can all be passed as parameters. The REXECD server can log on, execute the command and log off. REXECD can also be used for network management purposes. Procedures to start and stop TCP/IP devices (physical links) can be initialized using the REXEC command and the OBEYFILE command. You can also specify unique startup parameters and customize the server environment by modifying the REXECXIT EXEC procedure.

You should include REXECD in the AUTOLOG statement as a virtual machine to be started when TCP/IP is initialized.

Figure 85 shows the REXEC REXECD principle.

![Figure 85. REXEC REXECD principle](image-url)
An REXEC daemon must be running on the foreign host where your command is to be issued. For a detailed discussion about using the REXEC and RSH commands, refer to Chapter 9 of *TCP/IP Function Level 320 User's Guide*, SC24-5848.

For information about REXEC daemons on other platforms, see the appropriate documentation associated with the foreign host being used.
Appendix A. Using translation tables

TCP/IP uses translation tables to convert transmitted and received data between EBCDIC and ASCII. TCP/IP provides many different translation tables to meet the diverse needs of VM/ESA users.

Translation tables are stored in binary form on disk. TCP/IP provides more than 200 standard tables that are used as the default if you do not customize your own.

Two types of translation tables are used by TCP/IP of VM. SBCS translation tables are used for single-byte characters. DBCS translation tables are used for converting double-byte characters.

The following sections provide the information you need to understand what translation tables are, how they are used by TCP/IP applications, and how you can create your own custom translation table.

A.1 TCP/IP translation table euro support

There is just 1 year left to ensure your business and its IT systems are ready to cope with the euro.

A.1.1 Base VM TCP/IP support for the euro

Nearly 200 new translation tables (TCPXLBIN files) were added to VM TCP/IP. All of the tables have a file name of the form hhhhwwww, where hhhh is the 4-digit decimal code page number to be used on the host, and wwww is the 4-digit decimal code page number to be used on the client. Use leading zeros if necessary. For example, assume that you are using host code page 1047 (POSIX) and workstation code page 819 (ISO 8859-1). The file name would be 10470819.

The code page you choose depends on the assumptions made by the various operating system and application environments. For example, even though you are using a Windows operating system, you may have a file that was copied from a UNIX system that requires a UNIX-specific code page.

Translations are provided for all standard country code pages and their euro equivalents, code page 924 (ISO 8859-15, EBCDIC), and code page 1047 (IBM OpenEdition) to the following workstation code pages:

- 819 ISO 8859-1 (ASCII)
- 923 ISO 8859-15 (ASCII)
- 850 PC
- 858 Euro-ready PC
- 500 International EBCDIC
- 1148 International euro-ready EBCDIC
- 1047 IBM OpenEdition EBCDIC
- 1252 Microsoft® Windows®
In addition, code pages are available to convert between each standard country code page and its euro-ready equivalent.

The default translation table for all VM TCP/IP clients and servers is STANDARD TCPXLBIN. This is a 7-bit ASCII non-reversible translation. All extended ASCII characters with values greater than 0x7F are lost because the high-order bit of each octet (byte) is ignored. All other translation tables included with VM TCP/IP are 8-bit tables. The 8-bit tables provide the highest fidelity of transmitted data, but should not be used in situations where a client or server is known to treat the high-order bit in each octet as a parity bit.

**Note:** It is important to recognize that changing a default translation table for servers such as FTP and NFS can corrupt data in a file if that file is uploaded and downloaded using different translation tables. (This does not apply to binary transfers, of course.)

### A.1.2 CMS support for the euro

CMS was modified to correctly display all characters in code page 924 (ISO 8859-15), as well as other IBM euro-ready code pages. Support was added to recognize character set ID 1353. In order for you to see the euro symbol, your 3270 terminal emulator must use fonts that contain the euro symbol. Contact your emulator vendor for support information.

The OPENVM GETBFS and PUTBFS commands were updated to increase the number of code pages that may be specified. Any code page specification that is valid for CMS Pipelines can be used with GETBFS and PUTBFS.

A new CSL routine, DTCXLATE, was added to VMLIB CSLLIB. It provides an API so that programs may use the translation information contained in TCP/IP translation tables (TCPXLBIN files). For example, the output from this routine can be used with the CMS Pipelines XLA TE stage.

For more information about DTCXLATE, enter HELP ROUTINE DTCXLATE on your VM system after or refer to the DTCXLATE Web page:


### A.1.3 FTP support for the euro

The FTP server was updated to enable individual users to specify the name of a particular TCP/IP ASCII-EBCDIC translation table that they wish to use.

```
site xlate tablename

site translate tablename
```

where *tablename* is the name of any TCP/IP translation table (TCPXLBIN file) that is available to the FTP server on the VM host. The translate table specified takes precedence over default translate tables and becomes the specified translate table until you specify another one. To cancel specification, enter the “site xlate” command without any parameters specified.

**Note:** If the FTP client does not support the “site” command, then try “quote site”. If neither command is available, consult your FTP client documentation. If you are unable to enter the site command, translations occur using the default translation table established by the VM TCP/IP administrator.
The VM FTP and TFTP clients support the specification of the TRANSLATE \textit{tablename} option.

The FTP client \texttt{locstat} and \texttt{status} subcommands were updated to display the default translate table name and the user-specified translate table name, if one was specified.

See FTP Support for euro on the following Web page for more information on selecting a translation table:

\url{http://www.vm.ibm.com/euro/FTP.html}

A.1.4 NFS feature support for the euro

Many different translation tables can be made available to NFS clients. The VM/ESA Version 2 Release 3 TCP/IP FL320 NFS feature was updated so that NFS clients may specify the ASCII-EBCDIC translation of their choice using the new \texttt{xlate=tablename} optional parameter on a MOUNT request. If a file with the name \texttt{tablename TCPXLBIN} is not available to the VMNFS server, the mount request will fail. If \texttt{xlate} is not specified, the default translation table will be used.

The VM TCP/IP Version 2 Release 4 and FL310 NFS features were updated to accept a euro-ready translation table as the preferred translation table. To do this, copy the selected translation table to TCPMAINT 198 as VMNFS TCPXLBIN and reinitialize the NFS server.

For more information about NFS support for euro, see "VM/ESA TCP/IP NFS Server Support" on Web page:

\url{http://www.vm.ibm.com/nfs}

A.1.5 SMTP support for the euro

To use a euro-ready translation table with SMTP from the preferred translation table, copy it to TCPMAINT 198 with the name SMTP TCPXLBIN and reinitialize the SMTP server. If no preferred translation table exists, use the STANDARD TCPXLBIN.

See \textit{TCP/IP Translation Tables} on the following Web page for more information on selecting a translation table:

\url{http://www.vm.ibm.com/euro/tcpxlate.html}

A.1.6 LPR/LPD support for the euro

The LPR client and LPD server applications were enhanced to provide users with the ability to specify alternate translation tables.

\textbf{LPR command}

The new TRANSLATE \textit{tablename} option was added to the preferred translation table to allow the user to specify which translation table the application will use for EBCDIC-ASCII data translations.

If TRANSLATE \textit{tablename} is not specified, a default translation table is used. If the LPRSET command was used to set the TRANSLATE option, the specified translation table will be used. If the Nickname option of LPR is specified, the TRANSLATE tag associated with the nickname is used, if present. Otherwise, data translation is performed as follows:

\begin{itemize}
  \item \texttt{LPR set translate tablenam...} for LPR
  \item \texttt{LPR set translation tablenam...} for LPD
\end{itemize}
• Asynchronous processing
   Data translation is based on the configuration of the RSCS server and link to which the files were directed.

• Synchronous processing
   LPR searches for and uses the LPR TCPXLBIN file first, and then the STANDARD TCPXLBIN file. If neither is found, an internal translation table is used.

**LPRSET command**
The LPRSET command (for TCP/IP for VM Function Level 310 and 320) was changed to support a new option, TRANSLATE `tablename`, that allows the user to set the default translation table used for EBCDIC-to-ASCII data translations.

**LPD (LPSERVE)**
The LPD command was changed to support a new LPD configuration file statement, TRANSLATETABLE `tablename`; the synonym is XLATETABLE. When used, this statement identifies a specific translation table to be associated with a given print service. This statement must follow a SERVICE statement in the LPD CONFIG file.

See Base TCP/IP support on the following Web page for a description of the new code pages:


**A.1.7 RSCS LPR/LPD support for the euro**
RSCS LPR link drivers were updated to allow the submitting user to specify an EBCDIC-ASCII translation table. The "TRANSLATETABLE" statement in the LPD CONFIG file specifies this translation table. RSCS LPD link drivers were updated to allow the specification of a VM TCP/IP translation table name to be used for ASCII-EBCDIC conversions on that link.

RSCS internal translation tables were updated to support the euro.

For more information about this RSCS support, refer to APAR VM61635 and APAR VM61791 on the following Web page:

Appendix B. Additional TCP/IP solutions

A complete TCP/IP solution for VM/ESA will probably require the use of programs that are not delivered as part of VM/ESA, TCP/IP or RSCS. These include programs such as Web browsers, Web servers and mail servers. The following sections provide information on some of these programs. More information on these and other solutions for VM and VSE will be available on the VM and VSE Alliance Web site at:

http://www.ibm.com/s390/vmvse/

More information on Web browsers and Web servers is available in Chapter 8 of the IBM redbook TCP/IP Tutorial and Technical Overview, GG24-3376. For information on Web servers for VM/ESA see the IBM redbooks Web-Enabling VM Resources, SG24-5347 and Web Server Solutions for VM/ESA, SG24-4874.

B.1 Web browsers

Generally, a browser is referred to as an application that provides access to a Web server. Depending on the implementation, browser capabilities and hence structures may vary. A Web browser, at a minimum, consists of a Hypertext Markup Language (HTML) interpreter and a Hypertext Transfer Protocol (HTTP) client, which is used to access HTML Web pages. Besides this basic requirement, many browsers also support File Transfer Protocol (FTP), NetNews Transfer Protocol (NNTP), Post Office Protocol (POP) and Simple Mail Transfer Protocol (SMTP), among other features, with an easy-to-manage graphical interface.

As with many other Internet facilities, the Web uses a client/server processing model. The Web browser is the client component. Examples of Web browsers include Mosaic, Netscape Navigator, Microsoft Internet Explorer, and Sun HotJava. Web browsers are responsible for formatting and displaying information, interacting with the user, and invoking external functions, such as TELNET, or external viewers for data types that Web browsers do not directly support. Web browsers have become the "universal client" for the Graphical User Interface (GUI) workstation environment, in much the same way that the ability to emulate popular terminals such as the DEC VT100 or IBM 3270 allows connectivity and access to character-based applications on a wide variety of computers. Web browsers are widely available for all popular GUI workstation platforms and are inexpensive.

Text mode Web browsers are also available, including several for the 3270 environment. Charlotte is a Web browser available for VM/ESA at no charge. Enterprise View and EnterWEB are Web browser products available for a charge.

B.1.1 Charlotte from IBM and Province of British Columbia

Charlotte provides an interactive interface (text only) to assist finding, displaying and obtaining information on the internet. Using Charlotte you can jump from one item of interest to another by simply selecting highlighted text.

Charlotte lets you display documents managed by Web servers. It also provides support for the Gopher, FTP and NNTP protocols.
Enterprise View is a VM/ESA text browser that provides access to the Internet and your intranet from 3270 terminals and desktop systems running 3270 emulators.

3270 users can now access the same Web site information as workstation browser users, except graphics are not displayed. As a text browser, Enterprise View provides 3270 users with fast and easy access to the Web. Even Web pages with data displayed in tables and forms can be easily viewed on a 3270 terminal using Enterprise View.

Desktop systems running 3270 emulators and connected to your mainframe with SNA can also access the Web. Enterprise View eliminates the need for TCP/IP and browsers to be installed on every desktop.

Enterprise View provides 3270 users with consistent access to their mainframe information, the Web and an intranet. To ensure secure access, Enterprise View supports user authentication.

As a native VM/ESA application, Enterprise View is simple to install, administer and does not require the use of VTAM. It is centrally installed and can be easily integrated with existing security and application management systems. Existing network definitions and special user authorizations are fully supported.

Your 3270 users can easily access the Web to obtain critical information while you leverage your investment in 3270 terminals and training.

More information on Enterprise View can be obtained at:
http://www.beyond-software.com/

EnterWEB is a full screen, character based multi-user Web browser. It is a TN3270 client and supports mail servers with POP or Internet Message Access Protocol (IMAP), along with Lightweight Directory Access Protocol (LDAP).

EnterWEB gives your 3270 and 3270 emulation users access to the Internet or to an intranet. Once connected, EnterWEB users can access the following applications:

- **HTML documents**
  These are documents, or pages, that are available on all Web sites or from an intranet network.

- **TELNET - VT100/VT220**
  These are applications running on a server which is available via a TCP/IP network. This includes applications running on Unix machines.

- **Groupware**
  This includes corporate information systems such as Lotus Notes and Microsoft Exchange.
With EnterWEB, users can view this information on-line and can optionally download it. EnterWEB only provides access to text-based information. You cannot view any graphics or images, or hear any sound. EnterWEB offers a basic, but fast way of getting yourself onto the World Wide Web, utilizing your existing hardware and software.

EnterWEB means that you can now give your 3270 and 3270 emulation users access to any “Web-enabled” information, that is information that has been coded in HTML.

EnterWEB is a standalone product that does not require any additional servers. Users issue a Dial command to EnterWEB, which connects to the Internet using TCP/IP for VM/ESA.

EnterWEB supports HTML 3.2, which means that users can view Forms, Tables and Frames. However, users cannot view images, video or sounds clips. These are replaced by a text default (<IMAGE>, <VIDEO>, and so on), unless the document provides an overriding text alternative.

EnterWEB supports Telnet processing. This means that users running EnterWEB can access any application which is available to the TCP/IP network. VT220/VT100 support extends EnterWEB’s Telnet support to the Unix environment and gives your users “full screen” access to any application running under Unix. EnterWEB establishes sessions based on the protocols used by VT100 and VT220 terminals. This enables 3270-type devices to emulate VT100 and VT220 devices. EnterWEB handles the translation to give 3270-type devices access to an application that would normally only be accessed by VT100 or VT220 devices.

Using 3270 screens means that EnterWEB cannot display graphic images. However, in the USA, a number of EnterWEB customers have bought the product for that very reason. With no graphic images to display, the transmission time and cost of accessing data is dramatically reduced. Users spend less time browsing and misuse of the Internet is reduced as users concentrate on the information they require and do not spend time searching for graphics or other non-business related matters.

If there is a real need to view graphics, users can use EnterWEB to download the Web page to a workstation disk and view the document using existing workstation software.

In addition to giving you access to traditional Web browser services, EnterWEB also gives you access to both TN3270 and VT220 (UNIX) applications

More information on EnterWeb can be obtained at:

http://www.macro4.com/

B.2 Web servers

Web servers are responsible for servicing requests for information from Web browsers. The information can be a file retrieved from the server’s local disk, or it
can be generated by a program called by the server to perform a specific application function.

Apache, RSK-GA and Webshare are Web servers available for VM/ESA at no charge. EnterpriseWeb/VM, VM:Webgateway and Web390 are Web server products available for a charge.

B.2.1 Apache from Apache Group

Apache was originally based on code and ideas found in the most popular HTTP server of the time: NCSA HTTPD 1.3 (early 1995). It has since evolved into a far superior system which can rival (and probably surpass) almost any other UNIX-based HTTP server in terms of functionality, efficiency and speed. Since it began, it has been completely rewritten, and includes many new features.

Apache is, as of April 1999, the most popular WWW server on the Internet, according to the Netcraft Survey. You can view the complete results of the survey at:

http://www.netcraft.com/survey

Apache was ported to VM/ESA by Neale Ferguson and is available as part of the IBM redbook *Porting UNIX Applications to OpenEdition for VM/ESA*, SG24-5458, available from the IBM Redbooks Web site at:

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

More information about Apache can be obtained at:

http://www.apache.org/

or Neale Ferguson’s Web site at:

http://pucc.princeton.edu/~neale/vmoe.html

B.2.2 RSK-GA from IBM

VM/ESA 2.4.0 includes a new server development tool known as the Reusable Server Kernel (RSK), also available as service to VM/ESA 2.2.0 and 2.3.0. For more information on the RSK see the *Reusable Server Kernel Programmer’s Guide and Reference*, SC24-5852 or the RSK Web site at:

http://www.ibm.com/s390/vm/rsk/

A high performance Web server for serving static HTTP items (HTML, GIFs, JPGs, whatever) has been written using RSK technology. It has no support for Common Gateway Interface (CGI) routines or HTTP basic authentication. It can be configured for new types of content (new Multipurpose Internet Mail Extensions or MIME types) and can serve from SFS or BFS. The RSK-GA package can be obtained from the VM/ESA Web site at:

http://www.ibm.com/s390/vm/download/

B.2.3 Webshare from Beyond Software

Webshare is a freeware VM/ESA Web server from Beyond Software. It has the basic HTTP/1.0 functionality suitable for learning about the Web and integrating it into your VM/ESA system. Once you start doing real applications or putting a significant load on your Web servers, you will want to look into obtaining a commercial Web server.
Webshare is completely independent from UNIX HTTPD implementations. It is written entirely in REXX with CMS Pipelines. Enhancements are also available from Beyond Software, including support for server-side includes.

WebPac/1 is an add-on package for Webshare, available for a charge from Beyond Software. It provides additional utilities, functionality and performance to a basic Webshare installation. The components of WebPac/1 have been chosen to improve the performance of Webshare, making writing CGIs easier and to make Webshare more valuable to you company.

Webshare is available from Beyond Software at:
http://www.beyond-software.com/

For more information see the Webshare pages at:
http://www.beyond-software.com/Products/EWeb/Webshare/webshare.html

B.2.4 EnterpriseWeb/VM from Beyond Software

EnterpriseWeb/VM can convert your mainframe running VM/ESA into a powerful Web server that can be accessed by any standard Web browser anywhere on a TCP/IP network. EnterpriseWeb/VM lets users access VM/ESA mainframes from any standard browser, including Netscape Navigator and Microsoft Internet Explorer. These Web browsers can be used as client front ends for accessing data or applications. No special training, support, or terminal emulation programs are required.

Using Common Gateway Interface (CGI) scripts written using REXX and CMS Pipelines, information systems managers can utilize desktop graphical browser technology to provide an easy to use point-and-click interface to existing VM/ESA applications and information.

The demand for secure processing power is growing as organizations add complex applications to their environments. EnterpriseWeb/VM links into the existing mainframe security system and provides a level of security unmatched by typical Web server installations. EnterpriseWeb/VM is also upgradeable to use full encryption for sites requiring this function.

EnterpriseWeb/VM uses internet standards, specifically, HTTP, HTML, and MIME and can be used to serve any data stored by the mainframe. Data as diverse as text, binary files, images, sound clips and video can be accessed through HTML. Mainframe access is made easy to authorized users anywhere on the network.

EnterpriseWeb/VM makes it possible for organizations to continue to benefit from their existing investments in VM/ESA systems, the people that support them, and specially developed legacy applications that make up much of the business infrastructures in today’s global economy. Connectivity, reliability, scalability, and security of the mainframe running VM/ESA are leveraged, preserving the inherent strengths of these systems.

EnterpriseWeb/VM couples the powerful, reliable mainframe with the look, feel, and functionality of an easy-to-use Internet browser, providing a seamless interface across multiple platforms. Additional information on EnterpriseWeb/VM is available at:
http://www.beyond-software.com/
B.2.5 VM:Webgateway from Sterling Software

With the advent of e-business, bringing legacy applications to the Web has become imperative. Sterling Software's VM:Webgateway is a Web-to-Host solution that quickly and easily brings proven mainframe applications to Web browsers, while maintaining the strengths of the mainframe. It provides browser access to critical legacy applications and data through a modern graphical interface. All without rewriting code or deploying complex, multi-tiered Web servers.

VM:Webgateway is the most cost-effective way to bring all of your mainframe information to your users' desktops. By Web-enabling your legacy systems, you provide browser access to all of your VM/ESA, VSE/ESA, TPF, and OS/390 applications, including full-screen applications. By Web-enhancing, you can exploit the browser capabilities to dramatically improve the applications' graphical user interfaces - adding graphics, audio, and video elements to the screen.

This full-function, secure Web server hosted on VM/ESA leverages the strengths of the VM/ESA platform. These strengths include providing Reliability, Availability and Scalability (RAS) which is critical in e-business and serving guest applications so you can use a single Web server for all your legacy applications. VM:Webgateway leverages existing, proven mainframe security. It also exploits Secure Sockets Layer (SSL) technology to encrypt data transmitted between Web browsers and the mainframe. Plus, the product uses authentication procedures and access control rules to ensure that only authorized users have access to the information.

You can store, as well as serve, Java Applets and ActiveX controls with VM:Webgateway. When a Web browser accesses a Java applet or ActiveX control, the applet or control launches an application on the client. VM:Webgateway is an excellent repository for files that users download to their own platforms when they need them. Users can access word processing, spreadsheet, and other desktop files that are on the Web server. When VM:Webgateway serves a desktop file, you can configure the Web browser to automatically launch the appropriate desktop application for that file.

VM:Webgateway exploits multitasking to process multiple requests simultaneously, maximizing throughput on a single server and avoiding user lock out. Using HTTP 1.1 persistent connections, VM:Webgateway can receive multiple files through the same connection, improving performance. You can serve multiple home pages from the same server with virtual hosting.

You can re-configure VM:Webgateway from either a browser or VM without needing to bring down the server. You can publish HTML from your workstation directly to VM:Webgateway. VM:Webgateway supports special characters in many different languages.

VM:Webgateway documents resource consumption for Web access to any legacy application. Any accounting package can process this record and track and/or charge for use of your Web site.

Providing universal access to your legacy applications from a Web browser allows you to administer them from a centralized location, making software deployment and updates easier. Plus, the flexibility of the Web browser technology provides you platform independence.
Finally, the familiar Web browser interface with point-and-click navigation reduces end-user training, while also helping you leverage your existing skills since you maintain the data and applications where they already exist-on the mainframe.

With VM:Webgateway, Sterling Software also offers two optional components—OfficeVision Interface and CICS/VSE Interface. The VM:Webgateway OfficeVision Interface gives you all of the strengths of VM:Webgateway plus a complete GUI for the calendar and e-mail facilities of IBM's OfficeVision/VM product. With the VM:Webgateway CICS/VSE Interface, VSE developers can Web-enhance using the familiar CICS/VSE COBOL and other CICS command-level programming languages.

Additional information on Sterling Software's VM:Webgateway may be found at: http://www.vm.sterling.com/

B.2.6 Web390 from Information Builders

Web390 transforms an IBM S/390 into a Web application and data server capable of supporting global internet and intranet activities of new and existing S/390 users. S/390 offers an ideal Web server environment, designed from the ground up to provide secure, 24-hour simultaneous access to applications and vast stores of data. Web390 links the S/390's massive processing power and storage capacities with the popular Web browser, providing a convenient way for organizations to communicate and distribute services and information to audiences worldwide.

Web390 provides a secure gateway to existing 3270-based applications running under OS/390 (TSO and CICS) or VM/ESA, giving Web browser users immediate access to the legacy applications and data that support day-to-day commercial operations. Users have a choice of screen presentation modes, to customize the look and feel of the application to their specific tastes. Users can also be given protected access to all datasets allocated in the S/390 environment and application developers can write CGI (Common Gateway Interface) programs in traditional S/390 languages to accept and process input from Web browsers. With Web390, you can reuse your existing data access code, written in COBOL, PL/I, or Assembler, to publish operational data directly on the Web.

Since Web390 is a secure Web server, mainframes can serve as repositories for distributing information to an entire workforce or customer base, on a “need to know” basis, in a completely secure fashion. The Web390 server supports all of the popular Web file formats (MIME types), including HTML, VRML, GIF, and JPEG graphic files, as well as MPEG animation files and, of course, JavaScript and Java applets. With Web390, all of your Web-based content is hosted in a native S/390 environment: standard minidisk files or SFS directories on VM, and standard PDS, PDSE and flat files in the MVS environment. There is no need to implement UNIX services in order to host your Web sites and grant Web access to applications with Web390.

Web390's robust OpenCGI supports the use of Common Gateway Interface (CGI) scripts written in REXX, server-side JavaScripts, or in 3GL languages such as C++, Assembler, PL/I, and COBOL for accessing native S/390 data. Plus it supports more advanced Web-based data publishing tools like Information Builders' WebFOCUS product for building robust self-service applications and full ad-hoc reporting environments that access operational data. It's broad language
support facilitates rapid enhancement and exploitation of mainframe production systems on the Web.

Web390 requires no changes to mainframe applications, so sites can deploy them immediately on the Web. Users with Web browsers can share a single copy of mainframe-based applications, and sites can update source applications and data quickly and conveniently in one place, instantly deploying changes worldwide. The fact that only a Web browser is required on the desktop greatly simplifies workstation requirements, reducing application deployment costs.

In addition to full CGI support, which enables Web390 to host new Web-based applications on the S/390, Web390 also supports instant access to existing 3270 screen-driven applications from the Web. Two modes of application access are available in Web390: 3270 emulation mode, and html translation mode. Users or application developers can select access modes based on their individual preferences.

Users of 3270 terminals may prefer Web390 3270 emulation mode, which employs a Netscape Navigator plug-in to deliver 3270 screens within the browser window – complete with support for color and graphic characters. This mode is compatible with all recent releases of Netscape Navigator or Microsoft Internet Explorer, including release 3.0 or above. The emulation window looks and acts exactly like a 3270 screen, offering both “soft” (pop-up keypad window) and “hard” (PF and PA keys mapped to user keyboard) emulation. At logon to Web390 users can select the screen size and resolution.

Users familiar with the Web may prefer Web390 Translation Mode, which translates the 3270 data streams into html, providing access to mainframe applications through familiar browser controls such as radio buttons and data entry windows. Since Translation Mode runs on any browser that supports html version 2.0 or higher, Web390 applications can be accessed from Windows, OS/2, Macintosh, and UNIX.

While Web390 supports instant browser access to any application, it is particularly good at accessing applications built using Information Builders' FOCUS Report-Writer and 4GL for S/390. With the complementary Web Interface option for FOCUS, Web390 can be used to fully transform legacy FOCUS applications and reports using custom html and style sheets. Now you can add images, drill-down, drop-down lists, color graphs and other graphic and styling elements to existing FOCUS reporting applications without the need to migrate to another platform.

Additional information on Web390 can be found at:
http://www.ibi.com/

B.3 Mail servers and clients

Mail servers provide function in VM/ESA which allows workstation users to utilize VM/ESA as gateway for electronic mail (e-mail) in a TCP/IP environment. TCP/IP for VM/ESA provides an SMTP server which is the vehicle for mail transport, but does not provide a repository. SMTP requires a server called a mailer to be that repository. If the mailer is not present, incoming mail is placed in the VM/ESA spool. Mail clients provide access for CMS users to mail to and from the Internet.
B.3.1 VM Mail Server from VM Resources Ltd.

VM Resources Ltd. has a full function POP3 server available in its VM Mail Server 1.0.0 product. VM Mail Server 1.0.0 fully supports the POP3 standard and extension commands specified in RFCs 1725, 1939, and 2449.

The product uses standard CMS facilities including the Reusable Server Kernel (RSK), the Shared File System, and CMS Callable Services. It runs in one VM service machine with a standard XC configuration. Management and administration is performed through standard CMS facilities and interfaces.

The Reusable Server Kernel provides the multithreading task management required for a busy mail server. VM Mail Server declares tasks to the RSK for spool management (inbound receipt of mail items), TCP/IP listening services (for inbound client requests), and data posting for mail delivery (outbound data transfer).

Mail boxes and mail items are stored in a single SFS file space. In theory, an unlimited amount of mail boxes and mail items are supported. This is limited in practice by space on the SFS pool, and by a VM Mail Server configuration option of limiting the amount of items or blocks a user can have.

Mail items are stored as CMS files. Attachments are fully supported. The mail format is open. An API is provided which lets any CMS user with the proper SFS authority store a CMS file in a mail box, and the client will see this as mail the next time they request mail from the server.

VM Mail Server works with all standard POP3 mail clients, such as Netscape, Eudora, and Microsoft Outlook and Outlook Express.

For additional information on VM Mail Server, see the following Web site:

http://www.vm-resources.ca/

B.3.2 IBM OfficeVision/VM

As electronic mail (e-mail) and communication networks have grown over the years one of the most often asked questions is “How can users send e-mail to each other in an easy and consistent way?”. OfficeVision/VM (OV/VM) with ReaderThief and the OV/VM Enhanced Mail Addressing PRPQ provide support for internet e-mail to 3270 users. For additional information go to the OV/VM Web site at:

http://www.ibm.com/s390/vm/related/ovvm/

B.3.2.1 OV/VM with ReaderThief

ReaderThief is a functional replacement of the OV/VM component (OFSSRDR ASSEMBLE) that is responsible for moving mail from an OV/VM client's virtual reader to his OV/VM in-basket. ReaderThief is a server that polls the spool for reader files that contain mail. When it finds such reader files, it sends the mail they contain to the OV/VM master mailbox manager, making it look like the mail was sent directly there in the first place. ReaderThief has the following advantages over the original OV/VM implementation:

- Reader mail is made available for OV/VM's shared in-basket capabilities.
- The amount of time mail is left in the virtual reader is reduced.
• Restrictions on MIME mail are lifted, reducing the need for gateways for internet mail.
• Exit points are provided for manipulating reader mail.
• Response time for opening the mail is improved.
• System performance peaks at times of high openmail activity are reduced.
• The conversion of internet and CMS mail to OV/VM format is improved.

B.3.2.2 OV/VM Enhanced Mail Addressing
IBM OfficeVision Enhanced Mail Addressing for VM/ESA (5799-FPL) enables users to send notes, meeting notices, and documents to users on a TCP/IP network via the standard OV/VM mail facility.

Highlights of OV/VM Enhanced Mail Addressing include:
• Allow entering of RFC822 addresses in OV/VM notes, documents, and meeting notices.
• Provide a single interface facility to create OV/VM and RFC822 address nicknames.
• Support the automatic addressing of notes when replying to notes from RFC822 addresses the same as OV/VM addresses.
• Allow OV/VM documents to be distributed on a TCP/IP network.

B.3.3 RSCS Data Interchange Manager
The RSCS Data Interchange Manager Feature provides support for the exchange of electronic mail (e-mail) between VM/ESA CMS and SMTP users within all TCP/IP networks serving as a bridge for exchanging mail between networks using the Network Job Entry (NJE) facility such as RSCS and VM/ESA, and networks using the SMTP protocol such as the Internet.

The CMS mail can be OfficeVision or CMS Notes. It also provides the ability to customize the operation to support the message format that may be in use by your particular location. All e-mail from the network will be translated into that locally used message format so that your end users need only ever see what they are accustomed to viewing on their VM system.

This feature is part of the RSCS 3.2.0 base.

B.3.4 MailBook
MailBook provides an electronic mail (e-mail) solution for VM/ESA CMS users.

The MAIL command reads and sends e-mail. MAIL’s incoming mailbox (inbox) is the standard CMS virtual reader, not a separate virtual machine like the OfficeVision inbox server. If inbox email exists, MAIL scans reader files and builds a menu of inbox messages. Items can then be selected for display. The sortable and searchable menu can be limited in various ways, including by date ranges.

Inbox messages can be acted upon in several way: saved in NOTEBOOK files, deleted, printed, replied to, or forwarded.

The MAIL command also sends email. Users can easily send email to one or multiple recipients, either typing the email message to send or including one or
more CMS files. Email can be sent to local system users or, if connected to the Internet, to users of systems around the world.

The MAILBOOK command manages the NOTEBOOK files created by the MAIL command. MAILBOOK uses the same user interface as the MAIL command, but works with messages stored in NOTEBOOK files instead of the user’s inbox.

NOTEBOOK messages can be acted upon in several ways: copied or moved to other NOTEBOOKs, deleted, printed, replied to, or forwarded.

MIME (Multipurpose Internet Mail Extensions), an Internet standard for email, specifies how to encode information beyond standard text composed of the 95-character “US-ASCII” character set. This standard has been implemented for many different computing environments. MailBook provides this facility to VM/ESA CMS users.

MailBook interacts consistently with users. A command line is always available, as is a menu bar with pull-down menus for most actions. Comprehensive context-sensitive help information is always available. Pop-up windows simplify specifying or selecting options.

Depending on system connectivity, MailBook can use a transport agent, such as IBM’s SMTP server, for delivering outgoing email. MailBook can deliver email directly from CMS to RSCS networks. Other third-party email transport agents, such as the LMAIL program from L-Soft International, Inc. can also deliver email.

For additional information on MailBook, see the following Web site:

http://www.MailBookSoftware.com/
Appendix C. RFCs related to protocol specifications

IBM is committed to industry standards. The internet protocol suite is still evolving through Requests for Comments (RFC). New protocols are being designed and implemented by researchers, and are brought to the attention of the internet community in the form of RFCs. Some of these are so useful that they become a recommended protocol. That is, all future implementations for TCP/IP are recommended to implement this particular function or protocol. These become the de facto standards, on which the TCP/IP protocol suite is built.

RFCs can be viewed or obtained online from the IETF Web page using the following URL:

http://www.ietf.org/rfc

Many features of TCP/IP for VM are based on the following RFCs:

Table 15. Requests for Comments

<table>
<thead>
<tr>
<th>RFC</th>
<th>Title</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>768</td>
<td>User Datagram Protocol</td>
<td>J.B. Postel</td>
</tr>
<tr>
<td>791</td>
<td>Internet Protocol</td>
<td>J.B. Postel</td>
</tr>
<tr>
<td>792</td>
<td>Internet Control Message Protocol</td>
<td>J.B. Postel</td>
</tr>
<tr>
<td>793</td>
<td>Transmission Control Protocol</td>
<td>J.B. Postel</td>
</tr>
<tr>
<td>821</td>
<td>Simple Mail Transfer Protocol</td>
<td>J.B. Postel</td>
</tr>
<tr>
<td>822</td>
<td>ARPA Internet Text Message</td>
<td>D.H. Crocker</td>
</tr>
<tr>
<td>826</td>
<td>Ethernet Address Resolution Protocol: or Converting Network Protocol</td>
<td>D.C. Plummer</td>
</tr>
<tr>
<td></td>
<td>Addresses to 48.Bit Ethernet Address for Transmission on Ethernet</td>
<td></td>
</tr>
<tr>
<td>854</td>
<td>Telnet Protocol Specification</td>
<td>J.B. Postel, J.K. Reynolds</td>
</tr>
<tr>
<td>877</td>
<td>Standard for the Transmission of IP Datagram over Public Data Networks</td>
<td>J.T. Korb</td>
</tr>
<tr>
<td>919</td>
<td>Broadcasting Internet Datagrams</td>
<td>J.C. Mogul</td>
</tr>
<tr>
<td>922</td>
<td>Broadcasting Internet Datagrams in the Presence of Subnets</td>
<td>J.C. Mogul</td>
</tr>
<tr>
<td>950</td>
<td>Internet Standard Subnetting Procedure</td>
<td>J.C. Mogul, J.B. Postel</td>
</tr>
<tr>
<td>951</td>
<td>Bootstrap Protocol</td>
<td>Bill Croft, John Gilmore</td>
</tr>
<tr>
<td>959</td>
<td>File Transfer Protocol</td>
<td>J.B. Postel, J.K. Reynolds</td>
</tr>
<tr>
<td>974</td>
<td>Mail Routing and the Domain System</td>
<td>Craig Partridge</td>
</tr>
<tr>
<td>977</td>
<td>Network News Transfer Protocol (NNTP)</td>
<td>Brian Kantridge</td>
</tr>
<tr>
<td>1001</td>
<td>Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Concepts and Methods</td>
<td></td>
</tr>
<tr>
<td>1002</td>
<td>Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Detailed Specifications</td>
<td></td>
</tr>
<tr>
<td>RFC</td>
<td>Title</td>
<td>Author</td>
</tr>
<tr>
<td>------</td>
<td>-----------------------------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>1009</td>
<td>Requirements for Internet Gateways</td>
<td>R.T. Braden, J.B. Postel</td>
</tr>
<tr>
<td>1014</td>
<td>XDR: External Data Representation Standard</td>
<td>Sun Microsystems Incorporated</td>
</tr>
<tr>
<td>1027</td>
<td>Using ARP to Implement Transparent Subnet Gateways</td>
<td>S. Carl-Mitchell, J.S. Quarterman</td>
</tr>
<tr>
<td>1032</td>
<td>Domain Administrators Guide</td>
<td>M.K. Stahl</td>
</tr>
<tr>
<td>1034</td>
<td>Domain Names Implementation</td>
<td>P. Mockapetris</td>
</tr>
<tr>
<td>1035</td>
<td>Domain Names Concepts and Facilities</td>
<td>P. Mockapetris</td>
</tr>
<tr>
<td>1033</td>
<td>Domain Administrators Operations Guide</td>
<td>M. Lottor</td>
</tr>
<tr>
<td>1042</td>
<td>Standard for the Transmission of IP Datagrams over IEEE 802 Networks</td>
<td>J.B. Postel, J.K. Reynolds</td>
</tr>
<tr>
<td>1055</td>
<td>Nonstandard for Transmission of IP Datagrams over Serial Lines: SLIP</td>
<td>J.L. Romkey</td>
</tr>
<tr>
<td>1058</td>
<td>Routing Information Protocol</td>
<td>C.L. Hedrick</td>
</tr>
<tr>
<td>1094</td>
<td>NFS: Network File System Protocol Specification</td>
<td>Sun Microsystems Incorporated</td>
</tr>
<tr>
<td>1157</td>
<td>Simple Network Management Protocol (SNMP)</td>
<td>J.D. Case, M. Fedor, M.L. Schoffstall, C. Davin</td>
</tr>
<tr>
<td>1179</td>
<td>Line Printer Daemon Protocol</td>
<td>The Wollongong Group, L. McLaughlin III</td>
</tr>
<tr>
<td>1180</td>
<td>TCP/IP Tutorial</td>
<td>T. J. Socolofsky, C.J. Kale</td>
</tr>
<tr>
<td>1208</td>
<td>Glossary of Networking Terms</td>
<td>O.J. Jacobsen, D.C. Lynch</td>
</tr>
<tr>
<td>1230</td>
<td>IEEE 802.4 Token Bus MIB IEEE 802.4 Token Bus MIB</td>
<td>K. McCloghrie, R. Fox</td>
</tr>
<tr>
<td>1231</td>
<td>1231 IEEE 802.5 Token Ring MIB IEEE 802.5 Token Ring MIB</td>
<td>K. McCloghrie, R. Fox, E.Decker</td>
</tr>
<tr>
<td>1270</td>
<td>SNMP Communications Services</td>
<td>F. Kastenholz, ed.</td>
</tr>
<tr>
<td>1350</td>
<td>TFTP Protocol</td>
<td>K.R. Sollins</td>
</tr>
<tr>
<td>1354</td>
<td>IP Forwarding Table MIB</td>
<td>F. Baker</td>
</tr>
<tr>
<td>1356</td>
<td>Multiprotocol Interconnect on X.25 and ISDN in the Packet Mode</td>
<td>A. Malis, D. Robinson, R. Ullmann</td>
</tr>
<tr>
<td>1387</td>
<td>RIP Version 2 Protocol Analysis</td>
<td>G. Malkin</td>
</tr>
<tr>
<td>1389</td>
<td>RIP Version 2 MIB Extension</td>
<td>G. Malkin</td>
</tr>
<tr>
<td>RFC</td>
<td>Title</td>
<td>Author</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------------------------------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>1390</td>
<td>Transmission of IP and ARP over FDDI Networks</td>
<td>D. Katz</td>
</tr>
<tr>
<td>1393</td>
<td>Traceroute Using an IP Option</td>
<td>G. Malkin</td>
</tr>
<tr>
<td>1436</td>
<td>The Internet Gopher Protocol</td>
<td>F. Anklesaria</td>
</tr>
<tr>
<td>1583</td>
<td>OSPF Version 2</td>
<td>J. Moy</td>
</tr>
<tr>
<td>1647</td>
<td>TN3270 Enhancements</td>
<td>B. Kelly</td>
</tr>
<tr>
<td>1700</td>
<td>Assigned Numbers</td>
<td>J.K. Reynolds, J.B. Postel</td>
</tr>
<tr>
<td>1725</td>
<td>Post Office Protocol - Version 3</td>
<td>J. Myers</td>
</tr>
<tr>
<td>1733</td>
<td>Distributed Electronic Mail Models in IMAP4</td>
<td>M. Crispin</td>
</tr>
<tr>
<td>1777</td>
<td>Lightweight Directory Access Protocol (LDAP)</td>
<td>W. Yeong</td>
</tr>
<tr>
<td>1813</td>
<td>NFS Version 3 Protocol Specification</td>
<td>B. Pawlowski, P. Stauback, Sun Microsystems Incorporated</td>
</tr>
<tr>
<td>1928</td>
<td>SOCKS Protocol Version 5</td>
<td>M. Leech</td>
</tr>
<tr>
<td>1929</td>
<td>Username/Password Authentication for SOCKS V5</td>
<td>M. Leech</td>
</tr>
<tr>
<td>1961</td>
<td>GSS-API Authentication Method for SOCKS Version 5</td>
<td>P. McMahon</td>
</tr>
<tr>
<td>2116</td>
<td>X.500 Implementations Catalog-96</td>
<td>C. Apple</td>
</tr>
<tr>
<td>2045</td>
<td>Multipurpose Internet Mail Extensions (MIME)</td>
<td>N. Freed</td>
</tr>
<tr>
<td>2046</td>
<td>Multipurpose Internet Mail Extensions (MIME)</td>
<td>N. Freed</td>
</tr>
<tr>
<td>2047</td>
<td>Multipurpose Internet Mail Extensions (MIME)</td>
<td>K. Moore</td>
</tr>
<tr>
<td>2048</td>
<td>Multipurpose Internet Mail Extensions (MIME)</td>
<td>N. Freed</td>
</tr>
<tr>
<td>2049</td>
<td>Multipurpose Internet Mail Extensions (MIME)</td>
<td>N. Freed</td>
</tr>
<tr>
<td>2060</td>
<td>Internet Message Access Protocol - Version 4 rev1</td>
<td>M. Crispin</td>
</tr>
<tr>
<td>2068</td>
<td>Hypertext Transfer Protocol -- HTTP/1.1</td>
<td>R. Fielding</td>
</tr>
<tr>
<td>2131</td>
<td>Dynamic Host Configuration Protocol (DHCP)</td>
<td>R. Droms</td>
</tr>
<tr>
<td>2132</td>
<td>DHCP Options and BOOTP Vendor Extensions</td>
<td>S. Alexander</td>
</tr>
<tr>
<td>2205</td>
<td>Resource ReSerVation Protocol (RSVP) Version 1</td>
<td>R. Braden</td>
</tr>
<tr>
<td>2338</td>
<td>Virtual Route Redundancy Protocol</td>
<td>S. Knight</td>
</tr>
<tr>
<td>2355</td>
<td>TN3270 Enhancements</td>
<td>B. Kelly</td>
</tr>
<tr>
<td>2400</td>
<td>Internet Official Protocol Standards</td>
<td>J. Postel</td>
</tr>
</tbody>
</table>
Appendix D. Enable the TCP/IP feature

D.1 During installation

The TCP/IP feature for VM/ESA must be enabled prior to use. To enable the TCP/IP feature, use the following steps.

1. Log on the installation user ID, P735FALT.

2. Issue the CMS QUERY DISK command to verify the VM/SES/E code and Software Inventory minidisks are correctly linked and accessed.
   - Verify the MAINT 5E5 minidisk is accessed as file mode B, and is linked R/O.
   - Verify the MAINT 51D minidisk is accessed as file mode D, and is linked R/W.

   If necessary, establish the appropriate access to the VMSES/E minidisks by entering the following commands:
   - Establish read access to the VMSES/E code minidisk.
     
     ```
     link MAINT 5e5 5e5 rr
     access 5e5 b
     ```
   - Establish write access to the VMSES/E code minidisk.
     
     ```
     link MAINT 51d 51d mr
     access 5 d d
     ```

3. Set the TCP/IP for VM/ESA to the enabled state.
   
   ```
   vmfins enable ppf 573falt {tcpipItcipsfs}
   ```
   
   **Note:** Use `tcpip` if the TCP/IP feature default minidisk environment was maintained; Use `tcipsfs` if the service minidisks were moved to Shared File System directories.

   This command sets the TCP/IP Feature as ENABLED in VMSES/E and CP.

   The following figure shows the entry in the PRODUCT ENABLE/DISABLE INFORMATION section of the SYSTEM CONFIG file.

   ```
   /**********************************************************************/
   /* PRODUCT ENABLE/DISABLE INFORMATION */
   /**********************************************************************/
   
   PRODUCT PRODID 5735FALT STATE ENABLE DESCRIPTION '05/25/99.00:00:00.$BASEDDR T CP/IP LEVEL 320 - TCP/IP FEATURE (BASE)'
   
   /*****************************************************************************/
   
   Figure 86. PRODUCT ENABLE/DISABLE INFORMATION section
   ```

D.2 Dynamically

Use SET PRODUCT command to define the TCP/IP Feature to the VM/ESA system.

```set product ppf 573falt enable```

This allows the user to:

- Dynamically add PRODUCT statements to the running system.
• Change the state of an existing PRODUCT statement.
• Delete an existing PRODUCT statement.

For a detailed description of how to use the SET PRODUCT command refer to Chapter 2 of CP Command and Utility Reference, Version 2 Release 3.0, SC24-5773.
Appendix E. Configuration files

This appendix presents various TCP/IP configuration files that may be referenced by TCP/IP clients, or a specific TCP/IP server, or both.

E.1 Basic configuration files

The basic configuration files are necessary to provide basic TCP/IP services for most environments.

E.1.1 IBM DTCPARMS

IBM DTCPARMS file contains server machines configuration definitions.

*******************************************************************
* ATTACH DEVICES
nick.TCPIP :type.server :class.stack
:attach.812 AS 812,813 AS 813
* Use RACF for servers which interface with it

* Network File System daemon
nick.VMNFS :type.server :class.nfs
:ESM_Enable.YES

* File Transfer Protocol (FTP) daemon
nick.FTPSERVE :type.server :class.ftp
:ESM_Enable.YES
:Anonymous.YES
:Parms.RDR A FTAUDIT

* File Transfer Protocol (FTP) daemon 2
nick.FTPSERV2 :type.server :class.ftp
:ESM_Enable.YES
:Anonymous.YES
:Parms.RDR A FTAUDIT

* Remote Execution (REXEC) and Remote Shell (RSH) daemon
nick.REXEC :type.server :class.rexec
:ESM_Enable.YES

*******************************************************************

E.1.2 PROFILE TCP

PROFILE TCPIP and HOSTS LOCAL are configuration files for the TCP/IP server virtual machine.

; File Name: PROFILE TCPIP
; Description: Sample PROFILE TCPIP Configuration File
;
; This configuration file is used to define system operation
parameters, Telnet and network configuration information for your
TCP/IP environment. The definitions within this file are used by
the TCP/IP virtual machine to establish your TCP/IP services during
initialization, although some statements within this file can be
changed dynamically by using the OBEYFILE command.
; For detailed information about the use and syntax of configuration
; statements used within this file, see "Chapter 6. Configuring the
; TCPIP Virtual Machine" in the "IBM TCP/IP for VM Planning and
; Customization" manual.
;
; Note: Comments within this file must begin with a semicolon (;)
; followed by at least one blank.
;
;=====================================================================
;=====================================================================
;
;Free Pool Statements
;
;Use the statements below to define the allocation of control blocks
;and data buffers used by the TCP/IP stack (TCPIP) and other TCP/IP
;service virtual machines. Ensure the Free Pool statements precede
;all other statements within this file.
;=====================================================================

ACBPOOLSIZEx 1000
ADDRESSTRANSLATIONPOOLSIZEx 1500
CCBPOOLSIZEx 150
DATABUFFERPOOLSIZEx 160 8192
ENVELOPEPOOLSIZEx 750
IPROUTEPOOLSIZEx 600
LARGENVELPOOLSIZEx 50 8192
RCBPOOLSIZEx 50
SCBPOOLSIZEx 256
SCSKPOOLSIZEx 256
SMALLDATABUFFERPOOLSIZEx 0
TCBPOOLSIZEx 256
TINYDATABUFFERPOOLSIZEx 0
UCBPOOLSIZEx 100

;=====================================================================
;Monitorrecords statement
;
;Use the MONITORRECORDS statement to select the monitor data records
;that are produced by TCPIP, if any.
;=====================================================================

MONITORRECORDS

;=====================================================================
;Tracing Statements
;
;Define tracing statements below, to log events for diagnostic
;purposes. Tracing statements defined within this file will be
;enabled during TCPIP server initialization.
;
;Note: For most situations that require tracing, use of the OBEYFILE
;command to dynamically enable and disable tracing may be more
;advantageous than defining trace statements within this file.
;
;NOTRACE disables all TCP/IP tracing. For detailed information about
;tracing, see the description of the TRACE statement in the
;"IBM TCP/IP for VM Planning and Customization" manual.
;=====================================================================

NOTRACE ALL
TRACE MOST
LESSTRACE ALL
MORETRACE MOST
SCREEN
NOSCREEN
FILE TCPIP PROB0001 A

;=====================================================================
;Control trace file timestamps
;=====================================================================

TIMESTAMP 0

;=====================================================================
;Define user IDs to receive messages when serious errors occur.
;=====================================================================

INFORM
OPERATOR TCPMAINT
ENDINFORM
Configuration files

---

Various TCPIP initialization, operation and authorization parameters.

---

Assorted Parameters

---

ASSORTEDPARMS

NOFWD

VMDUMP

HDPOPQUEUELIMIT

RESTRICTLOWPORTS

PERMITTEDUSERSONLY

ENDASSORTEDPARMS

---

Define user IDs that are allowed to issue OBEYFILE and other restricted commands.

---

OBEY

OPERATOR TCPMAINT SNMPD SNMPQE ROUTED REXECD DHCPD ENDObey

---

Restrict specific user IDs (or a group of similarly named user IDs) from using TCP/IP services.

---

RESTRICT

VEND*

BADGUY

ENDRESTRICT

---

Permit specific user IDs to use TCP/IP services.

---

PERMIT

NETMAINT

ENDPERMIT

---

Keep-Alive Packet parameters

---

KEEPALIVEOPTIONS

INTERVAL 20

SENDGARBAGE FALSE

ENDKEEPALIVEOPTIONS

---

SYSCONTACT defines the contact person for this managed node, as well as how to contact this person. This definition supplies the value for the SNMP MIB variable, "sysContact".

---

SYSCONTACT

Main Operator (555-2150)

Susquahanna Hat Company

ENDSYSCONTACT

SYSLLOCATION defines the physical location of this node. This definition supplies the value for the SNMP MIB variable, "sysLocation".

---

SYSLLOCATION

First Floor Computer Room

ENDSYSLLOCATION

---

Server Virtual Machine-related Statements

---

AUTOLOG

BOOTPD PASSWORD ; BootP Server

DHCPD PASSWORD ; DHCP Server

FTPSERVE PASSWORD ; FTP SERVER
; TCP/IP Solutions for VM/ESA

; LPserv Password ; LP Server
; NAMEserv Password ; Domain Name Server
; NCSGLBD Password ; NCS GLBD Server
; NCSLLBD Password ; NCS LLBD Server
; PORTMAP Password ; PORTMAP Server
; REXEC Password ; REXEC Server
; ROUTED Password ; ROUTED Server
; SMTP Password ; SMTP Server
; SNMPD Password ; SNMP VM Agent Virtual Machine
; SNMPQE Password ; SNMP VM Client Virtual Machine
; TFTPd Password ; TFTPd (Trivial FTP) Server
; UFTD Password ; UFTD Server
; VMNFS Password ; NFS Server
ENDAUTOLOG

; Reserve ports for specific server machines. Port values used are
; those defined in RFC 1060, "Assigned Numbers"
; ----------------------------------------------------------------------
; PORT
; 20 TCP FTPSERVE NOAUTOLOG ; FTP SERVER
; 21 TCP FTPSERVE ; FTP SERVER
; 20 TCP FTPSERVE2 NOAUTOLOG ; FTP SERVER 2
; 21 TCP FTPSERVE2 ; FTP SERVER 2
; 23 TCP INTCLIENT ; TELNET Server
; 25 TCP SMTP ; SMTP Server
; 53 TCP NAMEserv ; Domain Name Server
; 53 UDP NAMEserv ; Domain Name Server
; 67 UDP BOOTP ; BOOTP Server
; 67 UDP DHCPP ; DHCP Server
; 69 UDP TFTP ; TFTP (Trivial FTP) Server
; 111 TCP PORTMAP ; PORTMAP Server
; 111 UDP PORTMAP ; PORTMAP Server
; 135 UDP NCSLLBD ; NCS LLBD Server
; 161 UDP SNMPD ; SNMP Agent
; 162 UDP SNMPQE ; SNMPQE Agent
; 512 TCP REXEC ; REXEC Server (REXEC)
; 514 TCP REXEC ; REXEC Server (RSH)
; 515 TCP LPSERVE ; LP Server
; 520 UDP ROUTED ; ROUTED Server
; 608 TCP UFTD ; UFTD Server
; 750 TCP VMKERB ; Kerberos Server
; 750 UDP VMKERB ; Kerberos Server
; 751 TCP AIMSserv ; Kerberos Database Server
; 751 UDP AIMSserv ; Kerberos Database Server
; 2049 UDP VMNFS ; NFS Server
; 2049 TCP VMNFS NOAUTOLOG ; NFS Server
;
; (End of PORT reservations)

; TELNET Server Configuration Statements.
; (The TELNET Server is an "internal client" of the TCP/IP server).
; ----------------------------------------------------------------------
; INTERNALCLIENTPARMS
; PORT 23
; INACTIVE 0
; TIMEMARK 600
; CCSTERMNAME TCPIP
; CONNECTEXIT TNEXIT1
; ENDINTERNALCLIENTPARMS

; DEVICE and LINK statements
;
; Define the network interfaces used in your environment.
; (Sample device and link statements are included below).
; ----------------------------------------------------------------------
;
; CTC CONNECTION FROM WTSCVMT TO WTSCVMXK
DEVICE VMXACTC CTC 812
LINK WTSCVMXK CTC 1 VMXACTC

; Routing Statements
;
; Define the routing information required for your environment via the
; HOME, GATEWAY, ARPAGE, BSDROUTIPARMS and PRIMARYINTERFACE statements.
Flush the ARP tables every nn minutes (a 5 minute interval is used below).

ARPAGE 5

Primary interface Definition

PRIMARYINTERFACE ETH1

Define the internet (IP) address(es) for this VM host

HOME

THESE ARE THE CURRENT HOME ADDRESSES THAT ARE BEING USED TODAY

9.12.13.62 WTSCVMXA

(Routing information)

Note:

* Routes defined via the GATEWAY statement are STATIC routes.

* Routes defined via the BSDROUTINGPARMS statement are DYNAMIC routes.

Static Routing Information

GATEWAY

<table>
<thead>
<tr>
<th>Network</th>
<th>First Address</th>
<th>Link Hop Name</th>
<th>Max. Packet Size (MTU)</th>
<th>Subnet Mask</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.12.13.63</td>
<td>WTSCVMXA 1500</td>
<td>HOST</td>
<td>1500</td>
<td>0.255.255.0</td>
<td>0.12.14.0</td>
</tr>
</tbody>
</table>

THESE ARE THE CURRENT GATEWAYS THAT ARE BEING USED TODAY IN PRODUCTION

Define The DEFAULT route used for any network not explicitly routed via the previous entries.

THIS THE CURRENT DEFAULT GATEWAY THAT IS BEING USED TODAY

DEFAULTNET = WTSCVMXA 1500 0

(End of GATEWAY Static Routing information)

TRANSLATE statements indicate relationships between internet addresses and network addresses. The TRANSLATE statement is most often used only in conjunction with HYPERchannel (HCH) and X.25 network connections.

TRANSLATE

(End of Translate information)

Start all network interface devices used in this environment.

THESE ARE THE CURRENT DEVICES THAT SHOULD BE STARTED

START VMXACTC

End of PROFILE STCPIP
E.1.3 HOSTS LOCAL

; HOSTS LOCAL
; -----------
; The format of this file is documented in RFC 952, "DoD Internet
; Host Table Specification".
; The format for entries is:
; NET : ADDR : NETNAME :
; GATEWAY : ADDR, ALT-ADDR : HOSTNAME : CP>Type : OPSYS : PROTOCOLS :
; HOST : ADDR, ALT-ADDR : HOSTNAME, NICKNAME : CP>Type : OPSYS : PROTOCOLS :
; Where:
; ADDR, ALT-ADDR = internet address in decimal, e.g., 26.0.0.73
; HOSTNAME, NICKNAME = the fully qualified hostname and any nicknames
; CP>Type = machine type (PDP-11/70, VAX-11/780, IBM-3090, C/30, etc.)
; OPSYS = operating system (UNIX, TOPS20, TENEX, VM/SP, etc.)
; PROTOCOLS = transport/service (TCP/TELNET,TCP/FTP, etc.)
; : (colon) = field delimiter
; :: (2 colons) = null field
; *** CP>Type, OPSYS, and PROTOCOLS are optional fields.
;
; Note: The NET and GATEWAY statements are not used by the TCP/IP for
; VM applications. However, some socket calls require the NET
; entries. For added performance, if your programs do not need
; the NET and GATEWAY statements, delete them before running
; the MAKESITE program.
;
HOST : 129.34.139.5 : SOFTWARE.WATSON.IBM.COM ::::
HOST : 127.0.0.1 : LOCALHOST ::::

E.1.4 TCPIP DATA

The TCPIP DATA file needs to be accessible to all TCP/IP servers and users; it
contains information used by all users.

;******************************************************************************
; Name of File: TCPIP DATA
; This file, TCPIP DATA, is used to specify configuration
; information required by TCP/IP client programs.
; Syntax Rules for the TCPIP DATA configuration file:
; (a) All characters to the right of and including a ';' will be
; treated as a comment.
; (b) Blanks and <end-of-line> are used to delimit tokens.
; (c) The format for each configuration statement is:
; <SystemName'':'> keyword value
; where <SystemName'':'> is an optional label which may be
; specified before a keyword; if present, then the keyword-
; value pair will only be recognized if the SystemName matches
; the node name of the system, determined by the CMS IDENTIFY
; command. This optional label permits configuration
; information for multiple systems to be specified in a single
; TCPIP DATA file.
;******************************************************************************
;TCPIPUSERID specifies the userid of the TCP/IP virtual machine.
;TCPIP is the default userid.
TCP/IPUSERID TCPIP

; TCPIPUSERID TCPIP
; HOSTNAME specifies the TCP host name of this VM host. If not
; specified, the default HOSTNAME will be the node name returned
; by the CMS IDENTIFY command.
;
; For example, if this TCP/IP DATA file is shared between two systems,
; OURVM and YOURVM, then the following two lines will define the
; HOSTNAME correctly on each system.
;
; OURVM: HOSTNAME OURVM
; YOURVM: HOSTNAME YOURVM
;
; WTSCVMT: HOSTNAME WTSCVMT
;
; DOMAINORIGIN specifies the domain origin that will be appended
; to host names passed to the resolver. If a host name contains
; any dots, then the DOMAINORIGIN will not be appended to the
; host name.
;
; DOMAINORIGIN WTSCVMT.IBM.COM
; DOMAINORIGIN ITSO.IBM.COM
;
; NSINTERADDR specifies the internet address of the name server.
; LOOPBACK (14.0.0.0) is the default value (your local name server).
; If a name server will not be used, then do not code an NSINTERADDR
; statement (Comment out the NSINTERADDR line below). This will cause
; all names to be resolved via site table lookup.
;
; NSINTERADDR 14.0.0.0
; NSINTERADDR 9.12.14.7
; 9.14.1.3 is corporate dns
; NSINTERADDR 9.14.1.3
; NSINTERADDR 9.114.151.254
;
; NSPORTADDR specifies the foreign port of the Name Server.
; 53 is the default value.
; NSPORTADDR 53
;
; RESOLVEVIA specifies how the Resolver is to communicate with the
; name server. TCP indicates use of TCP virtual circuits. UDP
; indicates use of UDP datagrams. The default is UDP.
; RESOLVEVIA UDP
;
; RESOLVERTIMEOUT specifies the time in seconds that the Resolver
; will wait to complete an open to the name server (either UDP or TCP).
; The default is 30 seconds.
; RESOLVERTIMEOUT 30
;
; RESOLVERUDPRETRIES specifies the number of times the resolver
; should try to connect to the name server when using UDP datagrams.
; The default is 1.
; RESOLVERUDPRETRIES 1
;
; TRACE RESOLVER will cause a complete trace of all queries to and
; responses from the name server or site tables to be written to
; the user's console. This command is for debugging purposes only.
; TRACE RESOLVER
;
; SMTPSERVERID specifies the userid of the local SMTP server.
; Optionally 'AT nodeid' can also be specified to define the
; location within the RSCS network of the SMTP gateway id.
; Multiple occurrences of SMTPSERVERID can be used to define
; multiple SMTP servers. The first instance is used by the CMS
; SENDFILE command with the SMTP or MIME option.
; SMTPSERVERID SMTP
;
; UFTSERVERID specifies the userid of this system's UFT server.
; An * for a value indicates use the network server value obtained
; from the IDENTIFY command. Multiple occurrences can be used to
; define multiple UFTD servers. The first instance is used by the
E.2 Configuration files for server machines

E.2.1 NSMAIN DATA

The NSMAIN DATA configuration file contains information for the Domain Name Server virtual machine. This file needs to be configured only if this server will be used on your system.

The PRIMARY and SECONDARY statements are used if you are using an SQL database.

PRIMARY watson.ibm.com WATSON ; Use authoritative tables WATSON0 & WATSON1
SECONDARY ibm.com IBM 129.34.128.245
; Transfer domain data for ibm.com into tables
; IBM0 and IBM1 from 129.34.128.245
;
CACHINGONLY NSMAIN CACHE * ; Sample CACHINGONLY data file
;
NEGATIVECACHING ; The NS will store negative query answers
STANDARDQUERYCACHE 100 ; The number of Standard queries to cache
INTERMEDIARYQUERYCACHE 50 ; The number of Intermediary queries to cache
INVERSEQUERYCACHE 5 ; The number of Inverse queries to cache
DATABASEQUERYCACHE 5 ; The number of Data Base queries to cache
;
LRUTIME 300 ; Time an entry must be in the cache before it can be replaced by the LRU algorithm
HOSTNAMECASE UPPER ; Hostnames in the SQL tables are in uppercase
DOMAINNAMEPORT 53 ; Listen on port N, default is 53
UDPONLY ; Use UDP only when contacting a remote NS
UDPRETRYINTERVAL 5 ; After N seconds, try next remote NS
NORECURSION ; Prevents NS from performing recursion
MSGNCH ; Specifies the NS use CP MSGNCH, not CP MSG
SMSGUSERFILE VALIDUSR ; Name of SMSG user id authorization exec
; TRACE QUEUE ; Display query origination and query name
E.2.2 VMNFS CONFIG

The VMNFS CONFIG file is used to specify values that the VMNFS server virtual machine should use for mount requests. It also is used to enable or disable PCNFSD operation.

;***********************************************************************
; * Name of File: VMNFS CONFIG
; * The VMNFS CONFIG file is used to specify the values that the VMNFS
; * server virtual machine should use for mount requests that
; * have specified options transsexual and/or linseeds.
; * It is also used to enable or disable PCNFSD operation. PCNFSD is
; * enabled by default.
; * Syntax Rules for the VMNFS configuration file:
; *(a) All characters to the right of and including a ';' will be
; * treated as a comment.
; *(b) Blanks and <end-of-line> are used to delimit tokens.
; *(c) All entries are of the form:
; * PCNFSD yes
; * where:
; * PCNFSD - Is a required keyword.
; * yes - Indicates whether or not PCNFSD operation is
; * enabled. Yes is the default.
; * or
; * VMfiletype filetype translate=yes lines=nl
; * where:
; * VMfiletype - Is a required keyword.
; * filetype - Is a required value. This is a file type
; * extension which may start or end with '*',
; * indicating wildcard.
; * Note: Case is ignored for filetype comparisons
; * (for example, BIN equals Bin).
; * translate=yes - Is an optional parameter. If specified, value
; * is yes or no. If not specified, the default of
; * translate=no is used.
; * lines=nl - Is an optional parameter. If specified, value
; * is nl, none, or cms. If not specified, the
; * default of lines=none is used.
; * See "Using the Network File System Commands" in the TCP/IP
; * User’s Guide for more information on how extension values are
; * used.
; ***********************************************************************

PCNFSD yes
VMfiletype *BIN translate=no lines=none
VMfiletype *EXE translate=translate-yesh lines=nl
VMfiletype *EXEEX translate=translate-yesh lines=nl
VMfiletype *EDIT translate=translate-yesh lines=nl
VMfiletype *AM translate=translate-yesh lines=nl
VMfiletype *ANSERV translate=translate-yesh lines=nl
VMfiletype *ANN translate=translate-yesh lines=nl
VMfiletype *ANNOUNCE translate=translate-yesh lines=nl
VMfiletype *APP translate=translate-yesh lines=nl
VMfiletype *APPEND translate=translate-yesh lines=nl
VMfiletype *ASC translate=translate-yesh lines=nl
VMfiletype *ASCI translate=translate-yesh lines=nl
VMfiletype *ASCI translate=translate-yesh lines=nl
VMfiletype *ASM translate=translate-yesh lines=nl
VMfiletype *ASM3705 translate=translate-yesh lines=nl
VMfiletype *ASSEMBLE translate=translate-yesh lines=nl
VMfiletype *AVL translate=translate-yesh lines=nl
VMfiletype *AVAIL translate=translate-yesh lines=nl
<table>
<thead>
<tr>
<th>VMfiletype</th>
<th>translate-yes</th>
<th>lines=nl</th>
</tr>
</thead>
<tbody>
<tr>
<td>A37</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>BASDATA</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>BASIC</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>BKS</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>BIGHELP</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>C</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>C++</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>CAT</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>CATALOG</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>CNTRL</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>COB</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>COBOL</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>COPY</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>CPP</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>DIRECT</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>DLCS</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>DOC</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>DOCUMENT</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>ERSERV</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>EXC</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>EXEC</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>FFT</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>FOR</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>FORM</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>FORTRAN</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>FREEFORT</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>GCS</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>GROUP</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>H</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>HPP</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>H++</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>JOB</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>LISTING</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>LXX</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>MAC</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>MACLIB</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>MACRO</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>MAK</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>MAKE</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>MK</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>MEMBER</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>MIMMO</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>MODULE</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>NAM</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>NAMES</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>NETLOG</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>NCXX</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>NOT</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>NOTE</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>NOTEBOOK</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>OPS*</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>OPT</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>OPTIONS</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>PACKAGE</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>PASCAL</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>PKG</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>PLAS</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>PLAS</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>PILOPT</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>PLS</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>PVT</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>REXX</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>RGJ</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>SCR</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>SCRIPT</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>STY</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>STYLE</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>TEXT</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>TEXTXXXX</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>TXT</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>TXTXXXX</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>UPDATE</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>UPDT</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>VMT</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>VSBASIC</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>VSBDATA</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
<tr>
<td>XED</td>
<td>translate-yes</td>
<td>lines=nl</td>
</tr>
</tbody>
</table>
E.2.3 SMTP CONFIG file

The SMTP configuration file defines how the SMTP server is to operate, and what services it provides.

```plaintext
;***********************************************************************
; Name of File: SMTP CONFIG
; The SMTP CONFIG file is used to specify runtime options and data
; to the SMTP virtual machine.
; Syntax Rules for the SMTP configuration file:
; (a) All characters to the right of and including a ';' will be
; treated as a comment.
; (b) Blanks and <end-of-line> are used to delimit tokens.
; See the TCP/IP for VM: Planning and Customization manual for
; a complete explanation of all the statements.
;***********************************************************************

; Defaults which normally aren't changed:

; Name of File: SMTP CONFIG
; The SMTP CONFIG file is used to specify runtime options and data
; to the SMTP virtual machine.
; Syntax Rules for the SMTP configuration file:
; (a) All characters to the right of and including a ';' will be
; treated as a comment.
; (b) Blanks and <end-of-line> are used to delimit tokens.
; See the TCP/IP for VM: Planning and Customization manual for
; a complete explanation of all the statements.

; The PORT statement causes SMTP to listen on the specified
; port_number
; PORT 25 ; Port to accept incoming mail on

; The BADSPOOLFILEID statement specifies the user ID on the local
; system where SMTP transfers unreadable spool files and looping
; mail.
; BADSPOOLFILEID TCPMAINT

; The INACTIVE statement specifies the number of seconds of inactivity
; after which SMTP considers a connection to be broken.
; INACTIVE 180

; The FINISHOPEN statement specifies the number of seconds SMTP waits
; while trying to establish a connection to a foreign site. After
; the specified number of seconds, SMTP aborts the connection.
; FINISHOPEN 120

; The RETRYAGE statement specifies the number of days after which
; SMTP returns mail as undeliverable.
; RETRYAGE 3

; The WARNINGAGE statement specifies the number of days after which
; a copy of the mail is returned to the sender.
; WARNINGAGE 1 ; warn sender that the mail has been undeliverable
; for 1 day, but that attempts to deliver the
; mail will continue for another 2 days.

; The RETRYINT statement specifies the number of minutes SMTP should
```
; wait between attempts to deliver mail to a TCP/IP host that is
; unavailable.
;
; RETRYINT 20
;
;******************************************************************************
; The MAXMAILBYTES statement specifies the maximum size, in bytes,
; of mail that is accepted over a TCP/IP connection.
; MAXMAILBYTES 5242880
;
;******************************************************************************
; The RESOLVERRETRYINT statement specifies the number of minutes
; SMTP waits between attempts to resolve domain names.
; RESOLVERRETRYINT 20
;
;******************************************************************************
; The RCPTRESPONSEDELAY statement specifies the amount of time the
; SMTP server delays responding to the RCPT commands from the
; sender SMTP, while it is waiting for a domain name resolution.
; RCPTRESPONSEDELAY 60
;
;******************************************************************************
; The TEMPERRORRETRIES statement specifies the number of times SMTP
; tries to deliver mail to a host with a temporary problem.
; TEMPERRORRETRIES 0 ; How many times to retry temporary delivery
; errors. The default, 0, means retry for
; RETRYAGE days; otherwise the mail is returned
; after this number of deliver attempts.
;
;******************************************************************************
; The LOG statement specifies that SMTP is to log all SMTP traffic.
; The SPOOL and DISK options indicate where the information is to
; be logged ($POOL indicates console).
;
; LOG DISK
;
;******************************************************************************
; ALTTCPHOSTNAME is used to specify an alternative fully qualified host
; name by which SMTP will know the local host. Mail sent to users at
; <hostname> are treated as if they were local users.
; ALTTCPHOSTNAME <hostname>
;
;******************************************************************************
; The POSTMASTER statement specifies the mailbox(es) where mail
; addressed to postmaster is spooled. Multiple POSTMASTER statements
; may be specified when not running in SECURE mode.
; POSTMASTER TCPMAINT
; POSTMASTER RSCUser@RSCNodc
; POSTMASTER SMTPUser@SMTPNodc
;
;******************************************************************************
; Use the SMSGAUTHLIST statement to specify the addresses of local
; users authorized to issue privileged SMTP SMSG commands.
; SMSGAUTHLIST
; TCPMAINT
; OPERATOR LocalUser
ENDSMSGAUTHLIST
;
;
;******************************************************************************
; Use the ONDISKFULL statement to specify a set of CP commands for
; SMTP to execute when specified SMTP 191 disk full thresholds are
; crossed.
The OUTBOUNDOPENLIMIT statement specifies the maximum number of simultaneous TCP connections over which SMTP will actively deliver mail.

OUTBOUNDOPENLIMIT 30

The OUTBOUNDOPENLIMIT statement specifies the maximum number of simultaneous TCP connections over which SMTP will actively deliver mail.

Configuration for a typical RSCS to TCP/IP mail gateway.

GATEWAY
RSCHDOMAIN bitnet ; Accept mail from and deliver mail to RSCS hosts
LOCALFORMAT NETDATA ; Local recipients receive mail in Netdata format
RSCSFORMAT NETDATA ; RSCS recipients receive mail in Netdata format

The OUTBOUNDOPENLIMIT statement specifies the maximum number of simultaneous TCP connections over which SMTP will actively deliver mail.

OUTBOUNDOPENLIMIT 30

Use the ALT RSCSDOMAIN to specify an alternate RSCS domain name.

This can be useful when the RSCS network is known by multiple domain names, such as "vnet" and "vnet.ibm.com".

ALT RSCSDOMAIN vnet

The REWRITE822HEADER statement specifies whether or not SMTP will rewrite the RFC822 headers on all mail passing from RSCS to TCP through the mail gateway. The SMTP RULES file specifies how the server is to rewrite the headers.

REWRITE822HEADER YES NOPRINT

Use the MAILER option if you run with the Columbia Mailer. Specify PUNCH and NOSOURCEROUTES for operation with the Columbia Mailer. The old FOLDnoSOURCEroute parameter is equivalent to specifying PUNCH and NOSOURCEROUTES.

MAILER MUSER@MNODE PUNCH NOSOURCEROUTES LOCAL RSCS UNKNOWN

The IPMAILERADDRESS statement specifies the address on which SMTP queues mail when it cannot resolve the recipients address - see the Planning and Customization manual for further details.

IPMAILERADDRESS <ip_addr>

Note: You cannot specify the UNKNOWN option of MAILER at the same time as IPMAILERADDRESS. This causes a fatal SMTP configuration message.

The RESTRICT statement specifies addresses of users who cannot utilize SMTP services.

RESTRICT RETURN ; return mail from restricted users
charming@ourvm.our.edu ; Don’t accept any mail from Prince Charming
charming@OURVMX ; via RSCS or TCP network.
charming@ourvm* ; This line takes place of previous two lines!
@castle ; Don’t accept mail from anyone at host castle
; TCP/IP Solutions for VM/ESA
;**************************************************************
; Use the SECURE statement if this SMTP machine is to run as an SMTP
; to RSCS Secure Gateway. Only users in the data file SMTP SECTABLE
; will be allowed to send mail, all other mail will be returned or
; rejected. Note that the file SECURITY MEMO will be sent to RSCS
; users that are not authorized to use the gateway.
; SECURE
;**************************************************************
; Use the DBCS statement if this SMTP machine is to perform DBCS
; code conversion on the mail. Consult the Planning and Customization
; manual for the necessary parameters.
; DBCS <parameters>
;**************************************************************
; Use the TRACE statement to turn on different types of tracing
; Trace All       ; Turns on all types of Tracing
; Trace CodeFlow  ; Turns on Code Flow Tracing
; Trace Conn      ; Turns on Connection Activity Tracing
; Trace Debug     ; Turns on Command and Reply Tracing
; Trace Notice    ; Turns on TCP/IP Notification Tracing
; Trace Resolver  ; Turns on Resolver Tracing
; Trace SPL       ; Turns on *SPL IUCV Tracing
;**************************************************************

E.2.4 UFTD CONFIG

The UFTD server configuration file is used to specify runtime options and data to
the UFTD virtual machine.

;*******************************************************************************
; Name of File:     UFTD CONFIG
;*******************************************************************************
; The UFTD CONFIG file is used to specify runtime options and data
; to the UFTD virtual machine.
;*******************************************************************************
; Syntax Rules for the UFTD configuration file:
; (a) All characters to the right of and including a ';' will be
;     treated as a comment.
; (b) Blanks and <end-of-line> are used to delimit tokens.
; See the TCP/IP for VM: Planning and Customization manual for
; a complete explanation of all the statements.
;*******************************************************************************
; Defaults which normally aren’t changed:
;*******************************************************************************
; The PORT statement causes UFTD to listen on the specified
; port_number
; PORT 608          ; Port to listen for contact on
;*******************************************************************************
; The IDENTIFY statement specifies whether the server should attempt
; to identify the connecting client via the identify protocol, RFC 1413.
; IDENTIFY YES      ; attempt to identify the client
;*******************************************************************************
The MAXFILEBYTES statement specifies the maximum size, in bytes, of a file that is accepted over a TCP connection.

```
MAXFILEBYTES * ; default to no limit
```

The NSLOOKUP statement specifies whether resolution of the client connection address should be done.

```
NSLOOKUP YES ; perform client address resolution
```

The TRACE statement specifies what types of tracing should be turned on at server initialization time.

```
TRACE ALL ; all tracing
TRACE CODEFLOW ; trace code paths
TRACE CONN ; trace connectivity (socket) information
TRACE IDENTIFY ; trace identifier activity
TRACE LINK ; trace buffers received and sent
TRACE NSLOOKUP ; trace address resolution
TRACE VERBOSE ; trace codeflow, conn and line
```

The TRANSLATE statement specifies whether a particular or tailored SBCS or DBCS translation table file should be used by the server.

```
TRANSLATE STANDARD ; specifies STANDARD TCPXLBIN should be used
```

The UFTCMDS EXIT statement specifies which UFT protocol commands should be processed by an administrator provided exit exec.

```
UFTCMDS EXIT UFTCMDX FOR ALL ; specifies UFTCMDX exec should be called for all allowed commands
```

E.2.5 ROUTED CONFIG

The ROUTED CONFIG file is used to specify the type of RIP packets.

```
; Name of File: ROUTED CONFIG
; The ROUTED CONFIG is used to specify the type of RIP packets that will be received or sent (if any) and a RIP2 authentication password.
; Syntax Rules for the ROUTED CONFIG:
; (a) All characters to the right of and including a ';' will be treated as a comment.
; (b) Blanks and <end-of-line> are used to delimit tokens.
; See the TCP/IP for VM: Planning and Customization manual for a complete explanation of all the statements.

; Defaults have been pre-configured.

; RIP_SUPPLY_CONTROL specifies one of the following options on a router-wide basis:
; 1) RIP1 - Unicast/Broadcast RIP Version 1 packets (Default)
; 2) RIP2B - Unicast/Broadcast RIP Version 2 packets (See description)
; 3) RIP2M - Unicast RIP Version 2/Broadcast Version 1 packets (Migration)
```
3) **RIP2** - Unicast RIP Version 2 packets
4) **NONE** - Disables sending RIP packets

Note: RouteD supports receiving multicasted RIP Version 2 packets only. Because VM/ESA TCP/IP Function Level 320 does not support multicasting, RIP Version 2 packets are unicast.

**RIP2** RIP Version 2 packets are unicast over point-to-point only; no RIP packets will be sent over multicast-incapable interfaces. If a point-to-point BSDROUTINGPARMS statement does not contain a destination address, no RIP Version 2 packets will be sent over that interface.

**RIP2M** RIP Version 2 packets are unicast over point-to-point interfaces and RIP Version 1 packets are broadcasted over broadcast-capable interfaces.

**RIP2B** RIP Version 2 packets are unicast over point-to-point interfaces and are broadcasted over broadcast-capable interfaces. Care must be taken when using this option since host route misinterpretations by adjacent routers running RIP Version 1 can occur. RIP2B may become obsolete in a future release.

**RIP_SUPPLY_CONTROL**: RIP1

**RIP_RECEIVE_CONTROL** specifies one of the following options on a router-wide basis:

1) **RIP1** - Receive RIP Version 1 packets only
2) **RIP2** - Receive RIP Version 2 packets only
3) **ANY** - Receive any RIP Version 1 and 2 packets (Default)
4) **NONE** - Disables receiving RIP packets

**RIP_RECEIVE_CONTROL**: ANY

**RIP2_AUTHENTICATION_KEY** specifies a plain text password containing up to 16 characters. The key is used on a server-wide basis and can contain mixed case and blank characters. The key will be used to authenticate RIP Version 2 packets and be included in the broadcasts for authentication by adjacent routers running RIP Version 2. For maximum security, set **RIP_SUPPLY_CONTROL** and **RIP_RECEIVE_CONTROL** to **RIP2**. This will discard RIP1 and unauthenticated RIP2 packets.

**RIP2_AUTHENTICATION_KEY**: 

---

TCP/IP Solutions for VM/ESA
Appendix F. Special notices

This publication is intended to discuss TCP/IP for VM/ESA Function Level 320 help customers, business partners and IBMer to plan, install and manage a VM/ESA TCP/IP network. The information in this publication is not intended as the specification of any programming interfaces that are provided by TCP/IP for VM/ESA Function Level 320. See the PUBLICATIONS section of the IBM Programming Announcement for TCP/IP for VM/ESA Function Level 320 for more information about what publications are considered to be product documentation.

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

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

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

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

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

Any performance data contained in this document was determined in a controlled environment, and therefore, the results that may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their specific environment.
Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

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

AFP
AnyNet
APPN
AT
CICS
CT
eNetwork
IBM Global Network
IMS
Language Environment
NetView
Nways
OfficeVision/VM
Operating System/2
OS/390
RACF
RS/6000
SP
System/390
VSE/ESA
XT
400

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc. in the United States and/or other countries.

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

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

PC Direct is a trademark of Ziff Communications Company in the United States and/or other countries and is used by IBM Corporation under license.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States and/or other countries. (For a complete list of Intel trademarks see www.intel.com/tradmarkx.htm)

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

SET 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.
Appendix G. Related publications

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

G.1 IBM Redbooks publications

For information on ordering these ITSO publications see “How to get IBM Redbooks” on page 231.

- TCP/IP Tutorial and Technical Overview, GG24-3376
- The Basics of IP Network Design, SG24-2580
- Web-Enabling VM Resources, SG24-5347
- Web Server Solutions for VM/ESA, SG24-4874
- Porting UNIX Applications to OpenEdition for VM/ESA, SG24-5458

G.2 IBM Redbooks collections

Redbooks are also available on the following CD-ROMs. Click the CD-ROMs button at http://www.redbooks.ibm.com/ for information about all the CD-ROMs offered, updates and formats.

<table>
<thead>
<tr>
<th>CD-ROM Title</th>
<th>Collection Kit Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>System/390 Redbooks Collection</td>
<td>SK2T-2177</td>
</tr>
<tr>
<td>Networking and Systems Management Redbooks Collection</td>
<td>SK2T-6022</td>
</tr>
<tr>
<td>Transaction Processing and Data Management Redbooks Collection</td>
<td>SK2T-8038</td>
</tr>
<tr>
<td>Lotus Redbooks Collection</td>
<td>SK2T-8039</td>
</tr>
<tr>
<td>Tivoli Redbooks Collection</td>
<td>SK2T-8044</td>
</tr>
<tr>
<td>AS/400 Redbooks Collection</td>
<td>SK2T-8049</td>
</tr>
<tr>
<td>Netfinity Hardware and Software Redbooks Collection</td>
<td>SK2T-8046</td>
</tr>
<tr>
<td>RS/6000 Redbooks Collection (BkMgr)</td>
<td>SK2T-8040</td>
</tr>
<tr>
<td>RS/6000 Redbooks Collection (PDF Format)</td>
<td>SK2T-8043</td>
</tr>
<tr>
<td>Application Development Redbooks Collection</td>
<td>SK2T-8037</td>
</tr>
<tr>
<td>IBM Enterprise Storage and Systems Management Solutions</td>
<td>SK3T-3694</td>
</tr>
</tbody>
</table>

G.3 Other resources

These publications are also relevant as further information sources:

- TCP/IP Function Level 320 Messages and Codes, GC24-5850
- TCP/IP Function Level 320 Planning and Customization, SC24-5847
- TCP/IP Function Level 320 User’s Guide, SC24-5848
- TCP/IP Function Level 320 Programmer’s Reference, SC24-5849
- Program Directory for TCP/IP Feature for VM/ESA and Features Function Level 320 Program Number 5654-030, GI10-4695 (Available with the product)
- VM RSCS Exit Customization, SH24-5222
- VM RSCS Operation and Use, SH24-5220
• VM RSCS Diagnosis Reference, SC24-5881
• VM RSCS General Information, GH24-5218
• VM RSCS Planning and Installation, SH24-5219
• VM RSCS Program Directory, GI10-4665
• PCL5e/PostScript Technical Reference, S544-5344
• Technical Reference Bundle, by Hewlett-Packard, Inc. part number 5021-0377 (Available from Hewlett-Packard)
• VM/ESA: CMS Application Development Reference, SC24-5762
• VM/ESA: CMS Command Reference, SC24-5776
• VM/ESA: CP Programming Services, SC24-5760
• Reusable Server Kernel Programmer’s Guide and Reference, SC24-5852

G.4 Referenced Web sites

These Web sites are also relevant as further information sources:

http://www.isoc.org/internet-history/
http://www.ietf.org/rfc.html
http://rs.internic.net/
http://www.ibm.com/javainfo/
http://www.java.sun.com/
http://www.socks.nec.com
http://www.printers.ibm.com/R5PSC.NSF/Web/nprefm
http://www.interactive.hp.com/cgi-bin/cspt/ljxxxx/dydncwrep.pl?lid=general&fid=bpl102705
http://www.netinst.com/
http://www.ibm.com/printers/
http://www.utexas.edu/academic/otl/software/lpr/
http://www.vm.ibm.com/nfs
http://www.ibm.com/s390/vm/vmse/
gopher://p370.bcs.bc.ca/11/vmtools
http://www.beyond-software.com/
http://www.macro4.com/
http://www.netcraft.com/survey
http://www.ibm.com/redbooks/
http://www.apache.org/
http://pucc.princeton.edu/~neale/vmoe.html
http://www.ibm.com/s390/vm/vsk/
http://www.ibm.com/s390/vm/download/
http://www.beyond-software.com/Products/EWeb/Webshare/webshare.html
http://www.vm.sterling.com/
http://www.ibi.com/
http://www.vm-resources.ca/
http://www.ibm.com/s390/vm/related/ovvm/
http://www.MailBookSoftware.com/
How to get IBM Redbooks

This section explains how both customers and IBM employees can find out about IBM redbooks, redpieces, and CD-ROMs. A form for ordering books and CD-ROMs by fax or e-mail is also provided.

  
  Search for, view, download, or order hardcopy/CD-ROM redbooks from the redbooks Web site. Also read redpieces and download additional materials (code samples or diskette/CD-ROM images) from this redbooks site.

  Redpieces are redbooks in progress; not all redbooks become redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

- **E-mail Orders**
  
  Send orders by e-mail including information from the redbooks fax order form to:

<table>
<thead>
<tr>
<th>In United States</th>
<th>e-mail address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outside North America</td>
<td><a href="mailto:usb6fpl@ibmmail.com">usb6fpl@ibmmail.com</a></td>
</tr>
</tbody>
</table>


- **Telephone Orders**
  
<table>
<thead>
<tr>
<th>In United States</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outside North America</td>
</tr>
</tbody>
</table>

  United States (toll free)  1-800-879-2755
  Canada (toll free)  1-800-IBM-4YOU
  Outside North America  Country coordinator phone number is in the “How to Order” section at this site:


- **Fax Orders**
  
<table>
<thead>
<tr>
<th>In United States</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outside North America</td>
</tr>
</tbody>
</table>

  United States (toll free)  1-800-445-9269
  Canada  1-403-267-4455
  Outside North America  Fax phone number is in the “How to Order” section at this site:


This information was current at the time of publication, but is continually subject to change. The latest information may be found at the IBM redbooks Web site.

---

**IBM Intranet for Employees**

IBM Redbook Fax Order Form

Please send me the following:

<table>
<thead>
<tr>
<th>Title</th>
<th>Order Number</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

First name  Last name

Company

Address

City  Postal code  Country

Telephone number  Telefax number  VAT number

☐ Invoice to customer number

☐ Credit card number

Credit card expiration date  Card issued to  Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM</td>
<td>International Business Machines Corporation</td>
</tr>
<tr>
<td>ITSO</td>
<td>International Technical Support Organization</td>
</tr>
<tr>
<td>AAA</td>
<td>Authentication, Authorization and Accounting</td>
</tr>
<tr>
<td>AAL</td>
<td>ATM Adaptation Layer</td>
</tr>
<tr>
<td>AFS</td>
<td>Andrews File System</td>
</tr>
<tr>
<td>AH</td>
<td>Authentication Header</td>
</tr>
<tr>
<td>AIX</td>
<td>Advanced Interactive Executive Operating System</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>APPN</td>
<td>Advanced Peer-to-Peer Networking</td>
</tr>
<tr>
<td>ARP</td>
<td>Address Resolution Protocol</td>
</tr>
<tr>
<td>ARPA</td>
<td>Advanced Research Projects Agency</td>
</tr>
<tr>
<td>AS</td>
<td>Autonomous System</td>
</tr>
<tr>
<td>ASCII</td>
<td>American Standard Code for Information Interchange</td>
</tr>
<tr>
<td>ASN.1</td>
<td>Abstract Syntax Notation 1</td>
</tr>
<tr>
<td>AS/400</td>
<td>Application System/400</td>
</tr>
<tr>
<td>ATM</td>
<td>Asynchronous Transfer Mode</td>
</tr>
<tr>
<td>BGP</td>
<td>Border Gateway Protocol</td>
</tr>
<tr>
<td>BIND</td>
<td>Berkeley Internet NameDomain</td>
</tr>
<tr>
<td>BNF</td>
<td>Backus-Naur Form</td>
</tr>
<tr>
<td>BRI</td>
<td>Basic Rate Interface</td>
</tr>
<tr>
<td>BSD</td>
<td>Berkeley Software Distribution</td>
</tr>
<tr>
<td>CA</td>
<td>Certification Authority</td>
</tr>
<tr>
<td>CBC</td>
<td>Cipher Block Chaining</td>
</tr>
<tr>
<td>CCITT</td>
<td>Comité Consultatif International Télégraphique et Téléphonique (now ITU-T)</td>
</tr>
<tr>
<td>CDMF</td>
<td>Commercial Data Masking Facility</td>
</tr>
<tr>
<td>CERN</td>
<td>Conseil Européen pour la Recherche Nucléaire</td>
</tr>
<tr>
<td>CGI</td>
<td>Common Gateway Interface</td>
</tr>
<tr>
<td>CHAP</td>
<td>Challenge Handshake Authentication Protocol</td>
</tr>
<tr>
<td>CICS</td>
<td>Customer Information Control System</td>
</tr>
<tr>
<td>CIDR</td>
<td>Classless Inter-Domain Routing</td>
</tr>
<tr>
<td>CIX</td>
<td>Commercial Internet Exchange</td>
</tr>
<tr>
<td>CLNP</td>
<td>Connectionless Network Protocol</td>
</tr>
<tr>
<td>CORBA</td>
<td>Common Object Request Broker Architecture</td>
</tr>
<tr>
<td>COS</td>
<td>Class of Service</td>
</tr>
<tr>
<td>CPCS</td>
<td>Common Part Convergence Sublayer</td>
</tr>
<tr>
<td>CPU</td>
<td>Central Processing Unit</td>
</tr>
<tr>
<td>CSMA/CD</td>
<td>Carrier Sense Multiple Access with Collision Detection</td>
</tr>
<tr>
<td>DARPA</td>
<td>Defense Advanced Research Projects Agency</td>
</tr>
<tr>
<td>DCE</td>
<td>Distributed Computing Environment</td>
</tr>
<tr>
<td>DCE</td>
<td>Data Circuit-terminating Equipment</td>
</tr>
<tr>
<td>DDN</td>
<td>Defense Data Network</td>
</tr>
<tr>
<td>DDNS</td>
<td>Dynamic Domain Name System</td>
</tr>
<tr>
<td>DEN</td>
<td>Directory-Enabled Networking</td>
</tr>
<tr>
<td>DES</td>
<td>Digital Encryption Standard</td>
</tr>
<tr>
<td>DFS</td>
<td>Distributed File Service</td>
</tr>
<tr>
<td>DHCP</td>
<td>Dynamic Host Configuration Protocol</td>
</tr>
<tr>
<td>DLC</td>
<td>Data Link Control</td>
</tr>
<tr>
<td>DLCI</td>
<td>Data Link Connection Identifier</td>
</tr>
<tr>
<td>DLL</td>
<td>Dynamic Link Library</td>
</tr>
<tr>
<td>DLSw</td>
<td>Data Link Switching</td>
</tr>
<tr>
<td>DLUR</td>
<td>Dependent LU Requester</td>
</tr>
<tr>
<td>DLUS</td>
<td>Dependent LU Server</td>
</tr>
<tr>
<td>DME</td>
<td>Distributed Management Environment</td>
</tr>
<tr>
<td>DMI</td>
<td>Desktop Management Interface</td>
</tr>
<tr>
<td>DMTF</td>
<td>Desktop Management Task Force</td>
</tr>
<tr>
<td>DMZ</td>
<td>Demilitarized Zone</td>
</tr>
<tr>
<td>DNS</td>
<td>Domain Name Server</td>
</tr>
<tr>
<td>DOD</td>
<td>U.S. Department of Defense</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DOI</td>
<td>Domain of Interpretation</td>
</tr>
<tr>
<td>DOS</td>
<td>Disk Operating System</td>
</tr>
<tr>
<td>DSA</td>
<td>Digital Signature Algorithm</td>
</tr>
<tr>
<td>DSAP</td>
<td>Destination Service Access Point</td>
</tr>
<tr>
<td>DSS</td>
<td>Digital Signature Standard</td>
</tr>
<tr>
<td>DTE</td>
<td>Data Terminal Equipment</td>
</tr>
<tr>
<td>DTP</td>
<td>Data Transfer Process</td>
</tr>
<tr>
<td>DVMRP</td>
<td>Distance Vector Multicast Routing Protocol</td>
</tr>
<tr>
<td>EBDC</td>
<td>Extended Binary Communication Data Interchange Code</td>
</tr>
<tr>
<td>EGP</td>
<td>Exterior Gateway Protocol</td>
</tr>
<tr>
<td>ESCON</td>
<td>Enterprise Systems Connection</td>
</tr>
<tr>
<td>ESP</td>
<td>Encapsulating Security Payload</td>
</tr>
<tr>
<td>FDDI</td>
<td>Fiber Distributed Data Interface</td>
</tr>
<tr>
<td>FQDN</td>
<td>Fully Qualified Domain Name</td>
</tr>
<tr>
<td>FR</td>
<td>Frame Relay</td>
</tr>
<tr>
<td>FTP</td>
<td>File Transfer Protocol</td>
</tr>
<tr>
<td>GGP</td>
<td>Gateway-to-Gateway Protocol</td>
</tr>
<tr>
<td>GMT</td>
<td>Greenwich Mean Time</td>
</tr>
<tr>
<td>GSM</td>
<td>Group Special Mobile</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HDLC</td>
<td>High-level Data Link Control</td>
</tr>
<tr>
<td>HMAC</td>
<td>Hashed Message Authentication Code</td>
</tr>
<tr>
<td>HPR</td>
<td>High Performance Routing</td>
</tr>
<tr>
<td>HTML</td>
<td>Hypertext Markup Language</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>IAB</td>
<td>Internet Activities Board</td>
</tr>
<tr>
<td>IAC</td>
<td>Interpret As Command</td>
</tr>
<tr>
<td>IANA</td>
<td>Internet Assigned Numbers Authority</td>
</tr>
<tr>
<td>IBM</td>
<td>International Business Machines Corporation</td>
</tr>
<tr>
<td>ICMP</td>
<td>Internet Control Message Protocol</td>
</tr>
<tr>
<td>ICSS</td>
<td>Internet Connection Secure Server</td>
</tr>
<tr>
<td>ICV</td>
<td>Integrity Check Value</td>
</tr>
<tr>
<td>IDEA</td>
<td>International Data Encryption Algorithm</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>JVM</td>
<td>Java Virtual Machine</td>
</tr>
<tr>
<td>JPEG</td>
<td>Joint Photographic Experts Group</td>
</tr>
<tr>
<td>LAC</td>
<td>L2TP Access Concentrator</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>LAPB</td>
<td>Link Access Protocol Balanced</td>
</tr>
<tr>
<td>LCP</td>
<td>Link Control Protocol</td>
</tr>
<tr>
<td>LDAP</td>
<td>Lightweight Directory Access Protocol</td>
</tr>
<tr>
<td>LE</td>
<td>LAN Emulation (ATM)</td>
</tr>
<tr>
<td>LLC</td>
<td>Logical Link Layer</td>
</tr>
<tr>
<td>LNS</td>
<td>L2TP Network Server</td>
</tr>
<tr>
<td>LPD</td>
<td>Line Printer Daemon</td>
</tr>
<tr>
<td>LPR</td>
<td>Line Printer Requester</td>
</tr>
<tr>
<td>LSAP</td>
<td>Link Service Access Point</td>
</tr>
<tr>
<td>L2F</td>
<td>Layer 2 Forwarding</td>
</tr>
<tr>
<td>L2TP</td>
<td>Layer 2 Tunneling Protocol</td>
</tr>
<tr>
<td>MAC</td>
<td>Message Authentication Code</td>
</tr>
<tr>
<td>MAC</td>
<td>Medium Access Control</td>
</tr>
<tr>
<td>MD2</td>
<td>RSA Message Digest 2 Algorithm</td>
</tr>
<tr>
<td>MD5</td>
<td>RSA Message Digest 5 Algorithm</td>
</tr>
<tr>
<td>MIB</td>
<td>Management Information Base</td>
</tr>
<tr>
<td>MILNET</td>
<td>Military Network</td>
</tr>
<tr>
<td>MIME</td>
<td>Multipurpose Internet Mail Extensions</td>
</tr>
<tr>
<td>MLD</td>
<td>Multicast Listener Discovery</td>
</tr>
<tr>
<td>MOSPF</td>
<td>Multicast Open Shortest Path First</td>
</tr>
<tr>
<td>MPC</td>
<td>Multi-Path Channel</td>
</tr>
<tr>
<td>MPEG</td>
<td>Moving Pictures Experts Group</td>
</tr>
<tr>
<td>MPLS</td>
<td>Multiprotocol Label Switching</td>
</tr>
<tr>
<td>MPOA</td>
<td>Multiprotocol over ATM</td>
</tr>
<tr>
<td>MPTN</td>
<td>Multiprotocol Transport Network</td>
</tr>
<tr>
<td>MS-CHAP</td>
<td>Microsoft Challenge Handshake Authentication</td>
</tr>
<tr>
<td>MTA</td>
<td>Message Transfer Agent</td>
</tr>
<tr>
<td>MTU</td>
<td>Maximum Transmission Unit</td>
</tr>
<tr>
<td>MVS</td>
<td>Multiple Virtual Storage Operating System</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>PAP</td>
<td>Password Authentication Protocol</td>
</tr>
<tr>
<td>PDU</td>
<td>Protocol Data Unit</td>
</tr>
<tr>
<td>PGP</td>
<td>Pretty Good Privacy</td>
</tr>
<tr>
<td>PI</td>
<td>Protocol Interpreter</td>
</tr>
<tr>
<td>PIM</td>
<td>Protocol Independent Multicast</td>
</tr>
<tr>
<td>PKCS</td>
<td>Public Key Cryptosystem</td>
</tr>
<tr>
<td>PKI</td>
<td>Public Key Infrastructure</td>
</tr>
<tr>
<td>PNNI</td>
<td>Private Network-to-Network Interface</td>
</tr>
<tr>
<td>POP</td>
<td>Post Office Protocol</td>
</tr>
<tr>
<td>POP</td>
<td>Point-of-Presence</td>
</tr>
<tr>
<td>PPP</td>
<td>Point-to-Point Protocol</td>
</tr>
<tr>
<td>PPTP</td>
<td>Point-to-Point Tunneling Protocol</td>
</tr>
<tr>
<td>PRI</td>
<td>Primary Rate Interface</td>
</tr>
<tr>
<td>PSDN</td>
<td>Packet Switching Data Network</td>
</tr>
<tr>
<td>PSTN</td>
<td>Public Switched Telephone Network</td>
</tr>
<tr>
<td>PVC</td>
<td>Permanent Virtual Circuit</td>
</tr>
<tr>
<td>QLLC</td>
<td>Qualified Logical Link Control</td>
</tr>
<tr>
<td>QOS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>RACF</td>
<td>Resource Access Control Facility</td>
</tr>
<tr>
<td>RADIUS</td>
<td>Remote Authentication Dial-In User Service</td>
</tr>
<tr>
<td>RAM</td>
<td>Random Access Memory</td>
</tr>
<tr>
<td>RARP</td>
<td>Reverse Address Resolution Protocol</td>
</tr>
<tr>
<td>RAS</td>
<td>Remote Access Service</td>
</tr>
<tr>
<td>RC2</td>
<td>RSA Rivest Cipher 2 Algorithm</td>
</tr>
<tr>
<td>RC4</td>
<td>RSA Rivest Cipher 4 Algorithm</td>
</tr>
<tr>
<td>REXEC</td>
<td>Remote Execution Command Protocol</td>
</tr>
<tr>
<td>RFC</td>
<td>Request for Comments</td>
</tr>
<tr>
<td>RIP</td>
<td>Routing Information Protocol</td>
</tr>
<tr>
<td>RIPE</td>
<td>Réseaux IP Européens</td>
</tr>
<tr>
<td>RISC</td>
<td>Reduced Instruction-Set Computer</td>
</tr>
<tr>
<td>ROM</td>
<td>Read-only Memory</td>
</tr>
<tr>
<td>RPC</td>
<td>Remote Procedure Call</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>TFTP</td>
<td>Trivial File Transfer Protocol</td>
</tr>
<tr>
<td>TLPB</td>
<td>Transport-Layer Protocol Boundary</td>
</tr>
<tr>
<td>TLS</td>
<td>Transport Layer Security</td>
</tr>
<tr>
<td>TOS</td>
<td>Type of Service</td>
</tr>
<tr>
<td>TRD</td>
<td>Transit Routing Domain</td>
</tr>
<tr>
<td>TTL</td>
<td>Time to Live</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>UID</td>
<td>Unique Identifier</td>
</tr>
<tr>
<td>URI</td>
<td>Uniform Resource Identifier</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator</td>
</tr>
<tr>
<td>UT</td>
<td>Universal Time</td>
</tr>
<tr>
<td>VC</td>
<td>Virtual Circuit</td>
</tr>
<tr>
<td>VM</td>
<td>Virtual Machine Operating System</td>
</tr>
<tr>
<td>VPN</td>
<td>Virtual Private Network</td>
</tr>
<tr>
<td>VRML</td>
<td>Virtual Reality Modeling Language</td>
</tr>
<tr>
<td>VRRP</td>
<td>Virtual Router Redundancy Protocol</td>
</tr>
<tr>
<td>VTAM</td>
<td>Virtual Telecommunications Access Method</td>
</tr>
<tr>
<td>WAN</td>
<td>Wide Area Network</td>
</tr>
<tr>
<td>WWW</td>
<td>World Wide Web</td>
</tr>
<tr>
<td>XDR</td>
<td>External Data Representation</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
<tr>
<td>X11</td>
<td>X Window System Version 11</td>
</tr>
<tr>
<td>X.25</td>
<td>CCITT Packet Switching Standard</td>
</tr>
<tr>
<td>X.400</td>
<td>CCITT and ISO Message-handling Service Standard</td>
</tr>
<tr>
<td>X.500</td>
<td>ITU and ISO Directory Service Standard</td>
</tr>
<tr>
<td>X.509</td>
<td>ITU and ISO Digital Certificate Standard</td>
</tr>
<tr>
<td>3DES</td>
<td>Triple Digital Encryption Standard</td>
</tr>
</tbody>
</table>
Index

Numerics
155 ATM 11
7-bit ASCII 188
8BITMIME 2

A
Abstract Syntax Notation 1 182
address 73
Address Resolution Protocol (ARP) 13, 175
AFP printer drivers for Windows 170
Apache 194
APPC/VM 55
AppleTalk 24, 26
Applets 48
application 171
environments 187
ARP
cache 175
request 175
ARPPNET 7
ASCII 187
ATM 27
ATM switch 27
ATMARPSERVER 1
AUTOLOG 171

B
BM OpenEdition 187
BOOTP 13, 174
BOOTPD
virtual machine 174
Bootstrap protocol (BOOTP) 41
bridges 25
bridging 21
BSDROUTINGPARMS 180
Byte File System 175

C
caching-only server 177
channel-attached routers 86
Channel-to-Channel support (CTC) 12
circuit switching 24
class A addresses 15
class B addresses 15
class C addresses 15
class C networks 83
class D addresses 15
CLIST 182
CMS SENDFILE command 184
CMS support for the Euro 188
Common Link Access to Workstation 12
Configuration files 209
DTCPARMS 209
HOSTS LOCAL 214
NSMAIN DATA 216
PROFILE TCP 209
ROUTED CONFIG 223
SMTP CONFIG 219
TCPIP DATA 219
UFTD CONFIG 222
VMNFS CONFIG 217
CTC 72
CTCA 12

D
daemon 61
DATAGRamfwd 77
DB2 177
DECnet 26
define a zone 177
DHCP 13
dynamic allocation 42
manual allocation 42
DHCPACK 173
DHCPD
server 171
DHCPDECLINE 173
DHCPNAK 173
DHCPOFFER 172
DHCPRELEASE 173
DHCPREQUEST 172
discontiguous subnets 69
DISCOVER 172
distance vector algorithm 67, 74
Distributed Program Interface 182
DNS 36
domain name resolution process 178
Domain Name Server 176
Domain Name Server (DNS) 9
DOMAINLOOKUP 1
DOMAINSEARCH 1
dotted decimal host address 5
DPI 182
DTCPARMS 105, 174
Dynamic Host Configuration Protocol (DHCP) 41
dynamically 207

E
EBCDIC 187
Electronic mail 115
Enterprise printing 145
concepts 146
configuration of RSCS 156
configuration of TCP/IP 156
configuring Windows 149
LPRXFORM CONFIG 163
printing from VM/ESA 168
printing from Windows 169
required documentation 148
setting up the LPD server in RSCS 157
setting up the LPR function in RSCS 160
setting up the RSCSDNS server 156
EnterpriseWeb/VM 195
ENTR 11
ETC BOOTPTAB 174
ETC DHCPTAB 171
ETC GATEWAYS 180
ETHERNET/IEEE 22
Euro-ready
  translation table 189

F
Fast Ethernet 11
fault-tolerant network connections 78
FDDI 11
File Transfer Protocol (FTP) 117
  filters
    block filter 96
    forward filter 94
    noforwarding filter 94
    noreceive filter 96
    passive filter 97
    receive filter 95
    ripoff filter 97
    supply off filter 97
Firewalls 50
  proxy servers 51
  screening filters 51
  SOCKS servers 51
FTP 34, 36, 117
  ANONYMOUS 133
  CHKIPADR EXEC 117
  command 126
  concepts 117
  configuration 117
  DTCPARMS 119
  FTP within an EXEC 133
  FTPEXIT 123
  GET 131
  MGET 131
  MPUT 131
  multiple servers 124
  PROFILE TCPIP 119
  PUT 131
  security 121
  server 1
  sessions 126
  SMSG interface 122
  support for the Euro 188
  FTPSERVE virtual machines 57
  fully-qualified host name 5

H
Hardware Address Resolution 13
HOME statement 3
Host addressing strategies 82
host name resolution 176
host routes 82
host takeover 99
HOSTS LOCAL 111, 209
HTTP 46, 47
  hub 24
HYPERchannel 11

I
IAB 8
IBM 2216 92
  RIP-In-Metric 92
  RIP-Out-Metric 92
IBM 3088 12
IBM 3745/46 11
IBM 9221 Integrated Communication Processor 10
IBM Open Systems Adapter 2 11
ICMP 21
IGNORERedirect 77
IMAP4 115
INTERALCLIENTPARMS statements 1
  interface 74
  interface metric customization 90
  interface metrics 90
  interior gateway protocols 66
  International Standard Organization 182
  Internet Activities Board (IAB) 8
  Internet message access protocol (IMAP) 40
  Internet Protocol (IP) 14
  Internet security 49
  Internetwork layer 10
IP
  address 171
    Authentication Header (AH) 52
    datagram 18
    Encapsulated Security Payload (ESP) 52
    Security Architecture (IPSec) 52
  IPX 8, 24
  ISOC 8
  IUCV 55, 72, 84

J
JavaScript 49

K
Key management 52
kill routes 86

L
LAN 22, 26
  bridges 23
  extenders 23
  media-access methods 22

G
GATEWAY 21
  gateway 74
GATEWAYS file 75
Gopher 46
switch 28
switches 23
topology 23
LCS1 87
LCS2 87
Lightweight Directory Access Protocol (LDAP) 42
local printers 147
Logical Device Service Facility (LDSF) 56
Logical Link Control (LLC) 27
loopback 16
LPD 36
configuration 190
LPD-type link 4
server 2
LPR 36
LPR-type links 4
searches 190
TCPXLBIN 190
LPR-LPD- and TCPASCII-type 4
LPR/LPD support for the Euro 189
LPRXFORM 5, 163

M
MailBook 200
MAINT 207
Management Information Base 182
mapping 38
Media Access Control (MAC) 27
metric 74
MIB 182
MIME 115
MORETRACE statement 2
MOUNT request 189
Multiaccess broadcast networks 13
Multicasting 1
Multipurpose Internet Mail Extensions (MIME) 39

N
NAMESRV virtual machine 176
Native Asynchronous Transfer Mode 1
Netbios 26
NETSTAT 64
network 62
  address 171
  configuration 174
  interface 62
  interface layer 10
  parameters 174
  protocols 32
Network File System (NFS) 41, 135
Network News 46
Network Station 174
NFS 135
  concepts 135
  creating alias 140
  DTCPARMS 136
  enabling 135
  mounting file systems 142
  operations 140
PROFILE TCPIP 136
security 139
SMSG interface 138
support for the Euro 189
Version 3 2
VMNFS CONFIG 137
NFSNET 7
NJE 184
NSMAIN
  command 176
  DATA configuration file 176

O
OBEYFILE 109, 185
OfficeVision/VM (OV/VM) 199
Open Shortest Path First (OSPF) 44, 67
OPENVM
  GETBFS 188
  PUTBFS 188
OSA-2
  adapter 101
  configuration 102
OSPF 73, 87

P
P735FALT 207
packet switching 25
PARM statement 184
path determination 29
PCL 164
ping 21
point-to-point 63
  link 24
  networks 13
POP3 115
port 32
portmap 40
POSIX
  translate tables 5
Post office protocol (POP) 39
PostScript 166
primary interface 85, 89, 92
primary name server 177
PRIMARYINTERFACE 21
print server consolidation 145
printer queue name 5
private address 71, 73
PRODUCT ENABLE 207
PROFILE TCPIP 209
Protocol Data Unit 182
random access memory 175
RARP 13
redundant 84
remote bridges 26
remote daemon 4
Remote Execution virtual machine (REXEC) 185
remote login 35
remote Network Job Entry (NJE) printers 147
Remote Procedure Call (RPC) 40
remote SHell protocol 185
Request For Comments (RFC) 8, 203
resolvers 179
REXEC 36
REXED
  principle 185
  server 185
REXECXIT EXEC 185
RFC 8, 175
ring topology 23
RIP 67
  database 73
  limitations 68
  packets 223
  router 179
  Version 1 179
  Version 2 179
RIP-1 message format 67
RIP-2 67, 70, 71, 84
  authentication 70
  message format 71
  multicast 70
  next hop 70
  route tag 70
  routing control 70
  sample configuration 71
  supernetting support 70
  variable subnetting 70
Route Killing 85
route killing 86
route recovery 88
RouteD 83, 93
  configuration 75
  explicit routes 75
  normal operation 74
  PROFILE TCPIP 77
  server 179
  server characteristics 77
  server configuration file 75
  trace 88
router 61
routing 21, 29, 30, 61
  algorithm 64
  bandwidth 31
  daemons 61
  DATAGRamfwd 77
  definition 67
  delay 31
  direct delivery 61
  distance vector algorithm 74
  dynamic 30, 66, 80
  gateway 61
  GATEWAYS file 75
  IGNORERedirect 77
  indirect delivery 61
  load 32
  metrics 31, 63
multipath 31
path length 31
reliability 31
Route Killing 85
RouteD normal operation 74
routed server configuration file 75
single-path 31
static 30, 65
table management 65
tables 62, 65
TCPIP.PROFILE 77
terminology 61
VARSUBNETTING 77
Routing Information Protocol 67, 179
  database 73
  definition 67
  discontiguous subnets 69
  limitations 61, 68
  RIP-2 70, 71
Routing Information Protocol (RIP) 43
RSCS Data Interchange Manager 200
RSCS LPR/LPD support for the Euro 190
RSH 36, 185
RSH commands 186
RSK-GA from IBM 194
S
SBCS translation tables 187
secondary interface 88
secondary network interface 84
Secure Sockets Layer (SSL) 50
separator page setting 5
service Daemons 171
servlets 48
SET PRODUCT command 208
Shared File System 3
Simple Mail Transfer Protocol (SMTP) 39, 115
Simple Network Management Protocol (SNMP) 42, 182
SMTP
  configuration 219
  Operation 115
  server 219
  server configuration 116
  support for the Euro 189
SNA/APPN 11
SNALINK virtual machine 57
SNMP
  agent 182
  daemon 182
  Query Engine 182
SNMPD
  virtual machine 182
SNMPIUCV task 183
SNMQE
  virtual machine 183
sockets 32
  interface 33
SOURCEROUTES 2
SOURCEVIPA 21, 81
SSL
  authentication 50
  integrity 50
  privacy 50
stack failures 99
star topology 23
subnet mask 73
subnetwork 62
switches 25
switching 21
SYSTEM DTCPARMS 106

T
TCP/IP
  applications 179
  CMS Servers 58
  Configuration files 105
  configuration files 209
  GCS servers 58
  hosts 181
  network 179
  overview 7
  PROFILE 107
  protocol 35
  routing protocols and techniques 179
  server initialization 58
  server startup 59
  stack 173
  starting 112
  stopping 112
  support for the euro 187
  TCPASCII-type 4
  virtual machine 56, 182
TCP/IP Function Level 310 1
TCP/IP Function Level 320 1
TCP/IP Solutions 191
  Mail servers 198
  Web browsers 191
  Web servers 193
TCPIP DATA 109, 179
  ATSIGN 110
  DOMAINLOOKUP 110
  DOMAINORIGIN 110
  DOMAINESEARCH 110
  HOSTNAME 110
  NSINTERADDR 110
  RESOLVETIMEOUT 110
TCPIP PROFILE
  Device 108
  Gateways 108
  Home 108
  Link 108
  Start 109
  Translate 109
  TCPIP virtual machine 56
  TCPIP.PROFILE 77
  TCPIXLBIN translate table 4
  TELNET 113
  concepts 113
  implementation 113
  operations 113
  printer sessions 4
  printing 114
  remote login 35
  security 114
  server 3
  TCPIXLBIN 113
  terminal emulation 35
TFTP 36, 117, 174
  clients 175
  daemon 174, 175
TFTPD
  PERMLIST 176
  server 175
  virtual machine 176
The Internet Architecture Board (IAB) 8
The Internet Assigned Numbers Authority (IANA) 8
The Internet Engineering Task Force (IETF) 8
The Internet Research Task Force (IRTF) 8
time 74
time-to-live 178
TN3270 36, 113
TN3270E 114
TN3270E-Type link 4
trace
  output 2
  Traceroute 21
  traffic splitting 90, 92
Transmission Control Protocol (TCP) 34
transport layer 10
tree topology 23

U
UDP
  datagram 174
  port number 174
UFT
  client 184
  server 184
UFTD-type link driver 184
UFTXIN 184
UFTXOUT 184
Universal File Transfer Daemon 184
Unsolicited File Transfer 5
User Datagram Protocol (UDP) 33
using exits 107

V
VARSUBNETTING 77
VCTC 12
VIP A 72, 81, 82, 83, 84, 91, 93, 99, 100, 181
Virtual IP Addressing 179
  alternate interface 84
  benefits 79
  block filter 97
  block filters 96
  configuration 20
definition 78
fault-tolerant network 100
fault-tolerant network connections 78
forward filter 94
forward filters 94
gateways options 74
host routes 82
host takeover 99
interface metric customization 90
noforwarding filter 95
noreceive filters 96
OSA-2 adapter 101
OSA-2 ports 101
passive filter 97
passive filters 97
primary interface 84
receive filters 95
ripoff filters 97
route filters 94
route killing 86
route recovery time 87
session traffic splitting 99
SOURCEVIPA 81
supply off filter 97
traffic splitting 90
VIPA address with shared OSA-2 adapter 101
Virtual Machine Communication Facility 55
virtual private networks 52
Virtual Router Redundancy Protocol (VRRP) 44
VM Mail Server 199
VM Special Message Facility 1
VM TCP/IP support 187
VM/ESA HOME IP addresses 181
VM:Webgateway 196
VMCF 55
VMLIB CSLLIB 188
VMNFS
  CONFIG 217
  TCPXLBIN 189
VMSES/E 207
VRRP
  introduction 44
  overview 44
VTAM virtual machine 57

W
Web390 197
Webshare 194
what is new in TCP/IP 1
Wide Area Network (WAN) 24
Winsock 33

X
X Window system 41
X-client 41
xlate 189
XNS 26
X-server 41
IBM Redbooks evaluation

TCP/IP Solutions for VM/ESA
SG24-5459-00

Your feedback is very important to help us maintain the quality of ITSO redbooks. Please complete this questionnaire and return it using one of the following methods:

• Use the online evaluation form found at http://www.redbooks.ibm.com/
• Fax this form to: USA International Access Code + 1 914 432 8264
• Send your comments in an Internet note to redbook@us.ibm.com

Which of the following best describes you?
_ Customer _ Business Partner _ Solution Developer _ IBM employee _ None of the above

Please rate your overall satisfaction with this book using the scale:
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction

Please answer the following questions:

Was this redbook published in time for your needs?   Yes___ No___

If no, please explain:

What other redbooks would you like to see published?

Comments/Suggestions: (THANK YOU FOR YOUR FEEDBACK!)

© Copyright IBM Corp. 1999 245